Summer 2013 Clyde Kruskal

# CMSC 351: Practice Questions for Final Exam

These are practice problems for the upcoming final exam. You will be given a sheet of notes for the exam. Also, go over your homework assignments. **Warning:** This does not necessarily reflect the length, difficulty, or coverage of the actual exam.

**Problem 1.** Show that

$$\frac{1}{2} \leq \sum_{j=1}^{\infty} \frac{1}{j2^j} \leq 1 \qquad \text{and} \qquad 1 \leq \sum_{j=1}^{\infty} \frac{1}{j^2} \leq 2$$

**Problem 2.** Let $A[1,..,n]$ be an array of $n$ numbers (some positive and some negative).

(a) Give an algorithm to find which *three* numbers have sum closest to zero. Make your algorithm as efficient as possible. Write it in pseudo-code.

(b) Analyze its running time.

**Problem 3.** Assume that you developed an algorithm to find the (index of the) $n/3$ smallest element of a list of $n$ elements in $2n$ comparisons.

(a) Using the algorithm (as a black box), give an algorithm, efficient in the worst case, to find the $k$th smallest element of a list.

(b) Write down a recurrence for (a bound on) the number of comparisons it executes in the worst case.

(c) Solve the recurrence (using constructive induction). Find the high order term exactly (but you do not need any low order terms).

(d) Using the (black box) algorithm for finding the $n/3$ smallest element and using the ideas and results of Parts (a), (b), and (c), give an efficient algorithm to find (the index of) two elements, the $k_1$th smallest and the $k_2$ smallest (for inputs $k_1$ and $k_2$). The algorithm description can be very high level and brief.

(e) How many comparisons does it use? Find the high order term exactly (but you do not need any low order terms). Give a brief justification.

**Problem 4.**

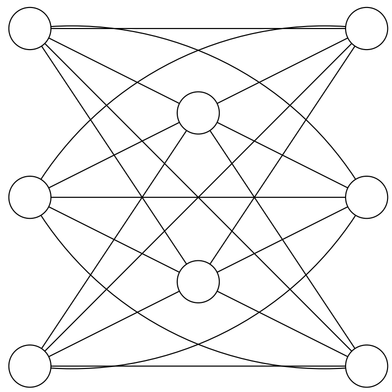
(a) Give a lower bound for sorting $n$ integers in range $1, \ldots, k$ using a *comparison-based algorithm.* You may assume $k \leq n$. [We know that counting sort can solve this problem in time $\Theta(n + k)$, but it uses basic operations other than comparisons.]

(b) Give an efficient comparison-based algorithm for sorting $n$ integers in range $1, \ldots, k$.

(c) Analyze your algorithm.

(d) Compare the upper and lower bounds.

**Problem 5.** A graph is tripartite if the vertices can be partitioned into three sets so that there are no edges internal to any set. The *complete* tripartite graph, $K(a, b, c)$, has three sets of vertices with sizes $a$, $b$, and $c$ and all possible edges between each pair of sets of vertices. $K(3, 2, 3)$ is pictured below. A *Hamiltonian* cycle in a graph is a cycle that traverses every vertex exactly once.



(a) For which values of $n$ does $K(1, 1, n)$ have a Hamiltonian cycle. Justify your answer.

(b) For which values of $n$ does $K(1, n, n)$ have a Hamiltonian cycle. Justify your answer.

(c) For which values of $n$ does $K(n, n, n)$ have a Hamiltonian cycle. Justify your answer.

**Problem 6.** Let $G = (V, E)$ be an undirected graph. A *triangle* is a set of three vertices such that each pair has an edge.

(a) Give an efficient algorithm to find all of the triangles in a graph.

(b) How fast is your algorithm?

**Problem 7.** This problem is more open-ended than you would see on an exam: If you do not know how to play Sudoku, look it up. Normally, Sudoku is played on a $9 \times 9$ grid.

(a) Generalize Sudoku to larger grids.

(b) State the (generalized) Sudoku game as a decision problem.

(c) Show that (generalized) Sudoku is in NP.

(d) Show that if you can solve the decision version of Sudoku in polynomial time, you can solve a Soloku puzzle in polynomial time.

**Problem 8.** For each pair of expressions $(A, B)$ below, indicate whether $A$ is $O$, $o$, $\Omega$, $\omega$, or $\Theta$ of $B$. Note that zero, one or more of these relations may hold for a given pair; list all correct ones.

| | $A$ | $B$ |
|---|---|---|
| $(a)$ | $n^{100}$ | $2^n$ |
| $(b)$ | $\sqrt{n}$ | $(\log n)^{12}$ |
| $(c)$ | $\sqrt{n}$ | $n^{\cos(\pi n/8)}$ |
| $(d)$ | $100^n$ | $10^n$ |
| $(e)$ | $n^{\log n}$ | $(\log n)^n$ |
| $(f)$ | $\log(n!)$ | $n \log n$ |