Midterm Exam — Introduction to Algorithms (CS 300)

April 26, 2012, 10:00 to 12:00

- Before you start: Write your name and student number on *every page* of your exam sheet.
- This is a closed book exam. You are not allowed to consult any book or notes.
- To ensure a quiet exam environment, we will not answer questions during the exam. If you think there is a mistake in the question, explain so on your answer sheet, and use common sense to answer the question.
- The questions have to be answered in *English.* Write clearly!

Problem 1: (32 pts) Each of these eight questions has one of the following five answers:

$$A : \Theta(1) \qquad B : \Theta(\log n) \qquad C : \Theta(n) \qquad D : \Theta(n \log n) \qquad E : \Theta(n^2)$$

Choose the correct answer for each question. (Correct answer 4 points, incorrect answer -2 points, and no points for no answer.)

(a) What is $\displaystyle\sum_{i=1}^{n} H_i$ (where H_i is the ith harmonic number, $H_i = \sum_{j=1}^{i} \frac{1}{j}$)?

(b) What is $\displaystyle\sum_{i=1}^{\log n} 2^i$?

(c) How many digits are required to write $n!$ in decimal?

(d) What is the solution to the recurrence $D(n) = 16D(n/4) + 2n \log n$?

(e) What is the solution to the recurrence $E(n) = E(n-2) + \log n$?

(f) What is the solution to the recurrence $F(n) = F(n-1) + 1/n^2$?

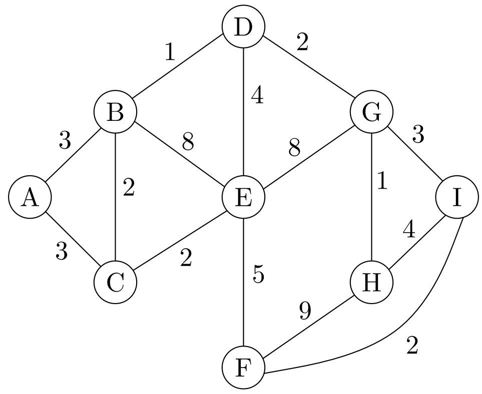
(g) What is $\displaystyle\sum_{i=1}^{n} c^i$, where $0 < c < 1$?

(h) In the worst case, how long does it take to solve the traveling salesman problem for 10,000,000,000 cities?

Problem 2: (18 pts) Let $G = (V, E)$ be the *complete undirected* graph on n vertices, where each edge has a *non-negative* edge weight. Given a source vertex $s \in V$, we wish to compute the length of the shortest path from s to every vertex $v \in V$. Explain how this can be done in time $\Theta(n^2)$ as easily as possible. (You do not have to explain algorithms we discussed in the class, but you need to justify the running time.)

You will receive 5 points if you write "I don't know" and nothing else.

Problem 3: (30 pts) You are given the following undirected graph G with weighted edges.



(a) Run Dijkstra's algorithm on G from source vertex A. When vertices have the same cost, choose them in alphabetical order. Only draw the final shortest-path tree for G. Label each vertex with the length of the shortest path from A. Also label the tree edges with the numbers 1 to 8 to indicate the order in which the edges are added by Dijkstra's algorithm.

(b) Run Prim's algorithm on G, starting from vertex A. When edges have the same weight, choose the edge with the alphabetically smallest endpoint first. Only draw the resulting minimum spanning tree for G, and label the tree edges with the numbers 1 to 8 in the order in which they are added by Prim's algorithm.

(c) Run Kruskal's algorithm on G. When edges have the same weight, choose again the edge with the alphabetically smallest endpoint first. Only draw the resulting minimum spanning tree for G, and label the tree edges with the numbers 1 to 8 in the order in which they are added by Kruskal's algorithm.

Problem 4: (20 pts) Let $G = (V, E)$ be an undirected weighted graph with n vertices and m edges. Consider a path between two vertices s and t in G. The *bottleneck length* of this path is the maximum weight of any edge in the path. The *bottleneck distance* between s and t is the minimum bottleneck length of *any* path from s to t. (If there are no paths from s to t, the bottleneck distance between s and t is ∞.)

For instance, in the graph G of problem 3, the bottleneck distance between A and G is 3, and the bottleneck distance between E and G is 2.

Describe an algorithm to compute the bottleneck distance for *every pair of vertices* of the graph G. The algorithm needs to output a list of $\binom{n}{2}$ bottleneck distances. Prove that your algorithm is correct and analyze its running time.

You will receive 5 points if you write "I don't know" and nothing else.