UC Berkeley—CS 170 | Final Exam
Lecturer: Tom Henzinger | December 17, 2002

Final Exam for CS 170

This exam consists of five problems, each worth 80 points, for a total of 400 points. Each problem has four parts (a)–(d). Please read all problems before you attempt solving any of them.

For full credit, you need to justify each answer, but try to be succinct in your justifications. If you are asked to provide an algorithm, write both (1) a brief prose description and (2) a more precise pseudo-code description. In pseudo-code, you may use any operations that are straight-forward to implement, such as $\max\{A[i] \mid 1 \leq i \leq n\}$ to find the maximum of the array $A[1..n]$.

Please write your solutions in the space provided in this booklet and put your student ID on every page. Clearly box and cross out rough work that does not form a part of your solutions.

Good luck!

Name:

Student ID:

Section Number or TA Name:

Problem	1	2	3	4	5	Total
Points	80	80	80	80	80	400
Score						

Problem 1. [Algorithm Analysis] (80 points)

(a) Let n be a power of 2. The maximum entry of an $n \times n$ matrix can be found by finding and comparing the maximum entries of the four $\frac{n}{2} \times \frac{n}{2}$ matrices that are the "quadrants" of the original matrix. What is the running time of this recursive scheme? Use big-O notation.

(b) Let n be a multiple of 10. An array of length n contains only 0s and 1s. The array is 0-*dominant* if it contains at least 90% 0s (and at most 10% 1s); the array is 1-*dominant* if it contains at least 90% 1s. We know that the array is either 0-dominant or 1-dominant, but not which of the two. How many array elements do we need to look at in order to decide conclusively if the array is 0-dominant or 1-dominant?

(c) This is a continuation of part (b). Again we know that the array is k-dominant, and again we do not know if $k = 0$ or $k = 1$. An algorithm chooses three array elements uniformly at random (the same element may be chosen repeatedly), and claims the array is k-dominant if two or more of the chosen elements are k. What is the worst-case probability that the algorithm gives a wrong answer?

(d) A sequence of 2^n operations is performed on a data structure. The i-th operation costs i if i is a power of 2, and 1 otherwise. What is the amortized cost per operation? Give either an exact answer or use big-O notation.

Problem 2. [Graph Algorithms] (80 points)

(a) An undirected graph is *unicyclic* if it is connected and contains exactly one cycle. Write an algorithm that checks if a given undirected graph is unicyclic. Provide a justification why your algorithm works. What is the asymptotic running time of your algorithm?

(b) Modify your algorithm from part (a) so that, if the input graph is unicyclic, the algorithm marks the nodes on the cycle (and leaves the other nodes unmarked). Your algorithm should run in linear time.

(c) How quickly can you find a minimum spanning tree of a unicyclic graph with positive edge weights? Use big-O notation; you need not give a pseudo-code algorithm, only a prose justification of your answer.

(d) How quickly can you find a minimum cut of a unicyclic graph with positive edge weights? Use big-O notation; you need not give a pseudo-code algorithm, only a prose justification of your answer.

Problem 3. [Optimization I] (80 points)

Suppose you have m dollars to invest in any of n enterprises. Assume that all investments must be integer amounts. The $m \times n$ matrix *Return* shows the expected returns for individual investments: $Return[i, j]$ is the expected return on an investment of i dollars in enterprise j (you may assume that the columns of *Return* are nondecreasing). You are asked to determine the maximum possible expected return for investing m dollars.

For example, consider the input matrix

$$Return = \begin{pmatrix} 1 & 2 \\ 2 & 2 \\ 3 & 2 \end{pmatrix}$$

and $m = 3$. As $Return[3, 1] = 3$ and $Return[3, 2] = 2$, the expected return on investing all 3 dollars in the first enterprise is 3, while the expected return on investing them in the second enterprise is only 2. The maximum possible expected return on 3 dollars is 4, which is achieved by investing 2 dollars in the first enterprise and 1 dollar in the second enterprise.

(a) There is a greedy strategy that allocates dollar by dollar. For the sample matrix from above, the greedy strategy invests the first dollar in the second enterprise for a maximum return of 2; the second dollar is invested in the first enterprise to achieve a maximum additional return of 1; and the third dollar is invested in the first enterprise to achieve a maximum additional return of 1. What is the asymptotic running time of implementing this greedy scheme (as a function of m and n)? Use big-O notation; you need not give the algorithm in pseudo-code.

(b) Give an example that shows that the greedy strategy of part (a) does not necessarily find the maximum possible expected return.

(c) Show that the greedy strategy of part (a) does not have a fixed approximation ratio; that is, for every $r > 0$, there is an example with optimum solution s, whereas the answer of the greedy algorithm is less than $\frac{s}{r}$.

(d) Write a dynamic programming algorithm that solves the problem. Explicitly give the recursive relationship between the optimal solutions for subproblems that your algorithm exploits. What is the asymptotic running time of your algorithm? Use big-O notation.

Problem 4. [Optimization II] (80 points)

Let C be a set of cities, and let L be a set of possible facility locations. Let l_i be the cost of building a facility at location $i \in L$, and let $c_{j,i}$ be the cost of connecting city $j \in C$ to a facility at location $i \in L$. The *facility location problem* is to find a subset $F \subseteq L$ of locations at which facilities should be be built, and a function $f\colon C \to F$ that assigns each city to a facility (several cities may be assigned to the same facility) such that the total cost $\Sigma_{i \in F}\, l_i + \Sigma_{j \in C}\, c_{j,f(j)}$ of building facilities and connecting cities to facilities is minimized.

(a) Define the corresponding decision problem FACILITY-LOCATION.

(b) Prove that FACILITY-LOCATION is NP-hard by reduction from SET-COVER. Recall that the SET-COVER problem asks, given a universe U of elements, a set $T = \{S_1, S_2, \ldots, S_m\}$ of subsets $S_i \subseteq U$, and a positive integer k, if there is a set $C \subseteq T$ such that (1) C contains at most k of the subsets in T, and (2) the union of all subsets in C is equal to U. Have the elements of U correspond to cities.

(c) Formulate the facility location problem (optimization version) as an integer linear program. An *integer linear program* is a linear program with the additional constraints that all solution values must be integers. Explain what each variable means.

(d) Is it possible to formulate the facility location problem as a polynomial-size linear program without the constraints that all solution values be integers?

Problem 5. [NP-Completeness] (80 points)

The SPECIFIED-ENDPOINTS HAM-PATH problem asks, given a directed graph G and two nodes v and w, if G contains a Hamiltonian path from v to w. You may assume that SPECIFIED-ENDPOINTS HAM-PATH is NP-complete.

(a) The UNSPECIFIED-ENDPOINTS HAM-PATH problem asks, given a directed graph G, if G contains a Hamiltonian path that starts in any node and ends in any node. Prove that UNSPECIFIED-ENDPOINTS HAM-PATH is NP-complete.

If $G = (V, E)$ is a directed graph, then the *transitive closure* $G^* = (V, E^*)$ is the graph with the same nodes as G and the following edges: $e = (u, v) \in E^*$ iff either $u = v$, or there is a path from u to v in G.

(b) What is the transitive closure G^* of a directed graph G that contains a Hamiltonian cycle?

(c) What is the transitive closure G^* of a directed graph G that contains a node $w \in V$ such that for all nodes $v \in V$, both $(v, w) \in E$ and $(w, v) \in E$?

The TRANSITIVE-SKELETON problem asks, given a directed graph $G = (V, E)$ and an integer k, if there is a subset $E' \subseteq E$ of the edges such that $|E'| \leq k$, and the two graphs G and $G' = (V, E')$ have the same transitive closure. If the answer is Yes, then the graph G' is called a *transitive skeleton* of G.

Consider the following polynomial-time reduction from UNSPECIFIED-ENDPOINTS HAM-PATH to TRANSITIVE-SKELETON: given a directed graph $G = (V, E)$, let $H = (W, F)$ be the directed graph with the nodes $W = V \cup \{w\}$ (where w is a new node, not in V), and with the edges $F = E \cup \{(v, w) \mid v \in V\} \cup \{(w, v) \mid v \in V\}$. Let $k = n + 1$, where n is the number of nodes in V.

(d) Prove that G has a Hamiltonian path (with any endpoints) if and only if H has a transitive skeleton with at most k edges.