CSE317 Design & Analysis of Algorithms

Final Examination

Max Marks: 55 Time Allowed: 3 hours

Answer all questions. Use seperate answer sheet.
Please give clear and rigorous answers. Be to the point. Show your work.

Name: _____ **ERP:** _____

Question 1: Asymptotic notations and Graphs ..... *6 marks*

(a) [3 marks] For each pair of expressions $T(n)$ and $f(n)$ below, indicate whether $T(n)$ is O, Ω, or Θ of $f(n)$.

$T(n)$	$f(n)$	$T \stackrel{?}{=} O(f)$	$T \stackrel{?}{=} \Omega(f)$	$T \stackrel{?}{=} \Theta(f)$
10^{1000}	1			
$(4n^2 - 12n + 9)/(2n - 3)$	n			
$n(n+1)/2$	n^3			
$3n^2 + 8n \log n$	$n/100$			

(b) [3 marks] Design a linear-time algorithm which, given an undirected graph G and a particular edge e in it, determines whether G has a cycle containing e.

Question 2: Divide & Conquer ..... *7 marks*

(a) [4 marks] You are given a sorted array of numbers where every value except one appears exactly twice; the remaining value appears only once. Design an efficient algorithm for finding which value appears only once.

Here are some example inputs to the problem:

1 1 2 2 3 4 4 5 5 6 6 7 7 8 8

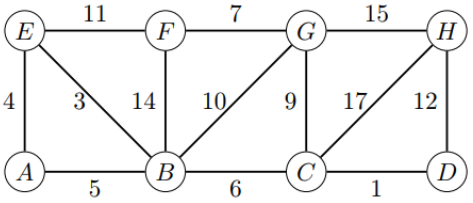
10 10 17 17 18 18 19 19 21 21 23

1 3 3 5 5 7 7 8 8 9 9 10 10

(b) [3 marks] The sequences $X_1, X_2, \ldots, X_l$, are sorted sequences such that $\sum_{i=1}^{n} |X_i| = n$. Show how to merge these l sequences in time $O(n \log l)$.

Question 3: MSTs ..... *8 marks*

(a) [3 marks] Consider the weighted graph below.



i. Run Prim's algorithm starting from vertex A. Write the edges in the order which they are added to the minimum spanning tree.

ii. Run Kruskal's algorithm starting from vertex A. Write the edges in the order which they are added to the minimum spanning tree.

(b) [5 marks] You are given an edge-weighted undirected graph, using the adjacency list representation, together with the list of edges in its minimum spanning tree (MST). Describe an efficient algorithm for updating the MST, when each of the following operations is performed on the graph. Give the running time of your algorithm as a function of V and/or E. Assume that common graph operations (e.g., DFS, BFS, finding a cycle, etc.) are available to you, and don't describe how to re-implement them.

i. Update the MST when the weight of an edge that ***was not*** part of the MST is ***decreased***.

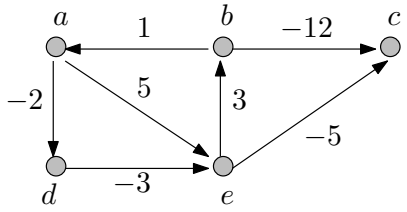
ii. Update the MST when the weight of an edge that ***was*** part of the MST is ***decreased***.

iii. Update the MST when the weight of an edge that ***was not*** part of the MST is ***increased***.

iv. Update the MST when the weight of an edge that ***was*** part of the MST is ***increased***.

Question 4: Shortest Paths ..... *7 marks*

(a) [3 marks] Run the Bellman-Ford shortest path algorithm on the following graph, starting from vertex a. Show the intermediate distances in a table after each iteration of the outer loop of the algorithm.



(b) [4 marks] *Generalized shortest-paths problem.* In Internet routing, there are delays on lines but also, more significantly, delays at routers. This motivates a generalized shortest-paths problem. Suppose that in addition to having edge lengths $\{l_e \mid e \in E\}$, a graph also has *vertex costs* $\{c_v \mid v \in V\}$. Now define the cost of a path to be the sum of its edge lengths, *plus* the costs of all vertices on the path (including the endpoints). Give an efficient algorithm for the following problem.

Input: A directed graph $G = (V, E)$; positive edge lengths l_e and positive vertex costs c_v; a starting vertex $s \in V$.

Output: An array cost[·] such that for every vertex u, cost[u] is the least cost of any path from s to u (i.e., the cost of the cheapest path), under the definition above.

Notice that cost[s] = c_s .

Question 5: Dynamic Programming ..... *9 marks*

(a) [4 marks] In this question we will compute edit-distance between the strings $X =$ *your-firstname* and $Y =$ *lastname*.

i. Write the recurrence for computing the optimal cost of a problem given the optimal solution of relevant subproblems. How many subproblems we get?

ii. Fill in the appropriate table using the recurrence from previous part. (If your first-name or last-name have more than 5 letters then consider the first 5 letters only).

(b) [5 marks] A pharmacist has W pills and n empty bottles. Let $\{p_1, p_2, \ldots, p_n\}$ denote the number of pills that each bottle can hold.

i. Describe a greedy algorithm, which, given W and $\{p_1, p_2, \ldots, p_n\}$, determines the fewest number of bottles needed to store the pills. Prove that your algorithm is correct (that is, prove that the first bottle chosen by your algorithm will be in some optimal solution).

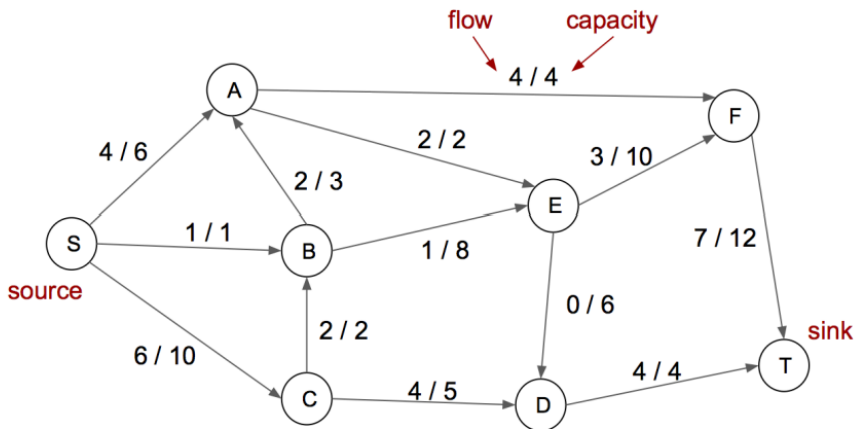
ii. Consider a variation when each bottle also has an associated cost c_i, and you want to minimize the total cost of the bottles used to store all the pills.
Let $MinPill(i, j)$ be the minimum cost obtainable when storing j pills using bottles among 1 through i. Give a recurrence expression for $MinPill(i, j)$. We shall do the base case for you:

- $MinPill(i, 0) = 0$ for all i
- $MinPill(0, j) = \infty$ for $j > 0$

iii. Describe briefly how you would design an algorithm for it using dynamic programming and analyse its running time.

Question 6: Network Flows ..... *6 marks*

Consider the following flow network and feasible flow f from the source vertex S to the sink vertex T.



(a) [1 mark] What is the size of the flow f?

(b) [1 mark] Draw residual graph G_f.

(c) [2 marks] Perform one iteration of the Ford-Fulkerson algorithm.

i. List the sequence of vertices on the augmenting path, in order from S to T.

ii. Update flow f along the augmenting path found above.

(d) [1 mark] What is the size of the maximum flow?

(e) [1 mark] List all vertices on the sink (T) side of the minimum cut.

Question 7: Short questions ..... *12 marks*

True/False/Fill in the blanks. Briefly justify your answer. Each part carry 1 mark except the last four which carry 1.5 marks each.

(a) Given n integers $a_1, \ldots, a_n$, the third smallest number among them can be computed in $O(n)$ time.

(b) It is known that there can be no polynomial-time algorithm for a NP-complete problem.

(c) The adjacency matrix representation is usually preferred over adjacency lists, especially for storing sparse graphs compactly.

(d) Dijkstra's algorithm can find shortest paths in a directed graph with negative weights, but no negative cycles.

(e) A graph flow is a max flow if and only if there exists no cut with the same capacity as the flow's value.

(f) The choice of which augmenting paths to consider first in the Ford-Fulkerson algorithm doesnt impact the number of paths that need to be considered.

(g) Given a directed graph G, consider forming a graph G' as follows. Each vertex $u' \in G'$ represents a strongly connected component (SCC) of G. There is an edge (u', v') in G' if there is an edge in G from the SCC corresponding to u to the SCC corresponding to v. Then G' is a directed acyclic graph.

(h) Consider two positively weighted graphs $G = (V, E, w)$ and $G' = (V, E, w')$ with the same vertex-set V and edge-set E such that, for any edge $e \in E$, we have $w'(e) = w(e)^2$. Then for any two vertices $u, v \in V$, any shortest path between u and v in G' is also a shortest path in G.

(i) The best case running time of Quicksort is $\Theta($_____$)$.

(j) Assume you have n positive integers in the range 1 through k. Counting Sort sorts the n integers in $O($_____$)$ time using $O($_____$)$ additional space.