CS570

Analysis of Algorithms

Spring 2017

Exam I

Name: _____

Student ID: _____

Email Address:_____

_____**Check if DEN Student**

	Maximum	Received

Problem 1	20	
Problem 2	15	
Problem 3	10	
Problem 4	10	
Problem 5	12	
Problem 6	10	
Problem 7	13	
Problem 8	10	
Total	100	

Instructions:

1. This is a 2-hr exam. Closed book and notes
2. If a description to an algorithm or a proof is required please limit your description or proof to within 150 words, preferably not exceeding the space allotted for that question.
3. No space other than the pages in the exam booklet will be scanned for grading.

4. If you require an additional page for a question, you can use the extra page provided within this booklet. However please indicate clearly that you are continuing the solution on the additional page.

20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE]
If f(n) = O(g(n)), then g(n) = Ω (f(n)).

[TRUE]
In the interval scheduling problem, if all intervals are of equal size, a greedy algorithm based on earliest start time will always select the maximum number of compatible intervals.

[FALSE]
There are at most 2 distinct solutions to the stable matching problem: one that is preferred by men and one that is preferred by women.

[FALSE]
The divide and conquer algorithm to solve the closest pair of points in 2D runs in O(n log n). But if the two lists of points, one sorted by X-coordinate and the other sorted by Y-coordinate are given to us as input, the rest of the algorithm (skipping the sorting steps) runs in O(n) time.

[TRUE].
In a graph, if one raises the lengths of all edges to the power 3, the minimum spanning tree will stay the same.

[TRUE]
The first edge added by Kruskal's algorithm can be the last edge added by Prim's algorithm.

[TRUE]
The shortest path of a graph could change if the weight of each edge is increased by an identical number.

[TRUE]
The shortest path in a weighted DAG can be found in linear time.

[FALSE]
If an operation takes O(1) amortized time, then that operation takes O(1) worst case time.

[FALSE]
In Binomial heaps, the decrease-key operation takes O(1) worst case time.

15 pts

You are given a set S of n points, labeled 1 to n, on a line. You are also given a set of k intervals $I_1, \ldots, I_k$, where each interval I_i, is of the form $[s_i, e_i]$, $1 \leq s_i \leq e_i \leq m$. Present an efficient algorithm to find a smallest subset X Í S of points such that each interval contains at least one point from X. Prove that your solution is optimal.

Solution: Sort the intervals in increasing order of ei. Select the first point as the right end-point of the first interval in this order. Remove all intervals which intersect with this point, and repeat. Proof of correctness is similar to the interval scheduling problem discussed in class. If the algorithm picks points p1 < p2 < · · · < pk, and optimum solution picks points q1 < q2 < · · · , qs, then prove by induction that pi $\geq$ qi, and so k $\leq$ s.

10 pts

Arrange the following functions in increasing order of growth rate with g(n) following f(n) in your list if and only if *f(n) = O(g(n))*

$\log n^n$, n^2, $n^{\log n}$, $n \log n \log \log n$, $2^{\log n}$, $n^{\sqrt{2}}$

Solution: $2^{\log n}$, $\log n^n$, $n \log n \log \log n$, $n^{\log n}$, $n^{\sqrt{2}}$, n^2

2) 10 pts
An ***ordered stack*** is a singly linked list data structure that stores a sequence of items in increasing order. The head of the list always contains the smallest item in the list. The ordered stack supports the following two operations. POP() deletes and returns

the head (NULL if there are no elements in the list). PUSH(x) removes all items smaller than x from the beginning of the list, adds x and then adds back all previously removed items. PUSH and POP can only access items in the list starting with the head. What would be the amortized cost of PUSH operation, if we start with an empty list?? Use the aggregate method.

Solution: The worst sequence of operations is pushing in order. Assume we push 1,2,3, …, n, starting with an empty list. The first push costs 1. The second push costs 3 (pop, push(2), push(1)). The last push costs 2n-1 (n-1 pops followed by push(n), followed by n-1 pushes). The total cost is given by

$1 + 3 + 5 + \ldots + (2n-1) = O(n^2)$
The amortized cost is O(n).

3) 12 pts
   a) 5 pts
      You are given a weighted graph G, two designated vertices s and t, and a number W. Your goal is to find a path from s to t in which every edge weight is at least W.

Describe an efficient algorithm to solve this problem and show its complexity. Your algorithm should work even if the edge weights are negative.

Solution: Run BFS, ignoring any edges of weight less than W. This will take O(V + E) time. Or Remove all edges with weight less than W and run BFS from s.

b) 7 pts

You are given a weighted graph G, two designated vertices s and t. Your goal is to find a path from s to t in which the minimum edge weight is maximized i.e. if there are two paths with weights 10->1->10 and 2->2->2 then the second path is considered better since the minimum weight (2) is greater than the minimum weight of the first (1). Describe an efficient algorithm to solve this problem and show its complexity. You may use the algorithm from Part (a). For full credit, your algorithm must run in
O((V + E) log E).

Solution: Sort all edges $e_1, e_2, \ldots, e_m$. Then run the Part a) algorithm with $W = e_k$, testing for each W whether you can find a path in the graph using only edges of weight greater than W. The largest W is the solution. How to choose W among $e_1, e_2, \ldots, e_m$? We do it in a binary search fashion. Runtime O((V + E) log E).
If we go through all edges, then O((V + E) E).
We could give slightly more credit to the better solution. Maybe 7 points vs 5 pts.

[redacted]10 pts

Given an instance of the original Stable Marriage problem with n couples (so that every man ranks every woman and vice versa), where there is a man M who is last on each woman's list and there is a woman W who is last on every man's list. Prove or

disprove the following statement: if we run the Gale-Shapley algorithm on this instance, then M and W will be paired with each other.

Solution: True. Note that no man will propose to W unless he has been rejected by all n-1 other women. Let X be the first man to propose to W. At the time this happens, all the other women must be engaged to men they prefer to X. This can only happen if X is M, as otherwise M would have been preferred to X by some woman. So M proposes to W, she accepts and then everyone is engaged and the algorithm stops

4) 13 pts
Suppose we define a new kind of directed graph where we have (positive) weights on the vertices and not the edges. If the length of a path is defined by the total weight of

all nodes on the path, describe an algorithm that finds the shortest path between two given points within this graph.

Solution:

Remove the vertex weight by slicing every vertex a into two vertices a1 and a2 with an edge from a1 to a2 with the given weight for original vertex. All edges going into a will go into a1 and all edges going out of a will go out of a2. Then, to find shortest path from x to y in the original graph, we can run Dijkstra's to find the shortest path from x1 to y2. (Suppose vertex x is sliced into x1, x2, and y is sliced into y1 and y2)

10 pts

Use the Master Theorem to solve each of the following recurrences by giving tight Θ-notation bounds, or describe why the Master Theorem does not apply to the particular recurrence formula(s).

(a) $T(n) = 2^n T(n/2) + n^n$

Solution:2^n is not constant, so master theorem does not apply

(b) $T(n) = 16T(n/4) + n!$ **Solution**: $\Theta(n!)$

(c) $T(n) = 4T(n/2) + c\,n$ **Solution**: $\Theta(n^2)$

(d) $T(n) = 2T(n/2) + n \log n$ **Solution**: $\Theta(n \log^2 n)$

(e) $T(n) = 64T(n/8) - n^2 \log n$ **Solution**: -n^2 log n is not asymptotically positive function, so master theorem does not apply

Additional Space

Additional Space