University of Washington
Department of Computer Science and Engineering
CSE 421, Winter 2019

October 21, 2019

Midterm Exam, Wednesday, February 13, 2019

NAME: ____________________

Instructions:

- Closed book, closed notes, no calculators
- Time limit: 50 minutes
- Answer the problems on the exam paper.
- If you need extra space use the back of a page
- Problems are not of equal difficulty, if you get stuck on a problem, move on.

1	/10
2	/10
3	/30
4	/10
5	/20
Total	/80

Problem 1. Stable Marriage (10 points):

Show that the Gale-Shipley Stable Marriage algorithm can take $\Theta(n^2)$ steps with appropriate choice of preference lists. Give preference lists and an ordering of the proposals that require $\Theta(n^2)$ steps. Explain why your example achieves the bound.

Hint: This can be done with all of the M's having the same preference lists, and all of the W's having the same preference lists.

Problem 2. Big Oh (10 points):

Let q, r, and s be positive constants. Prove that $qn^2 + rn + s$ is $O(n^2)$ using the formal definition of $O(\cdot)$.

Big $O(\cdot)$ definition: $f(n)$ is $O(g(n))$ if there exists $c > 0$ and $n_0 \geq 0$ such that for all $n \geq n_0$, $f(n) \leq cg(n)$.

Problem 3. True or False (30 points):

Determine if the following statements are true or false. Provide a short justification for each answer.

a) *True or false:* If G is a directed graph on n vertices where every vertex has out degree at least two, then G has a cycle. Justify your answer.

b) *True or false:* If G is a directed graph on n vertices with at least $2n$ edges, then G has a cycle. Justify your answer.

c) *True or false:* If G is a directed graph on n vertices, with distinct vertices r and s, where there is a path from r to every vertex in the graph, and there is a path from s to every vertex in the graph, then there is a cycle in the graph. Justify your answer.

d) *True or false:* If G is an undirected graph with edge weights, and edge e has weight strictly greater than any other edge in the graph, then e cannot be in a minimum spanning tree for G. Justify your answer.

e) *True or false:* If G is an undirected graph with edge weights, and edge e has weight strictly less than any other edge in the graph, then e must be in every minimum spanning tree for G. Justify your answer.

f) *True or false:* If G is a undirected graph on n vertices with more than $n/2$ connected components, then at least one of the connected components is an isolated vertex. Justify your answer.

Problem 4. Minimum Weight Branching (10 points):

A branching is a rooted subtree in a directed graph where there is a path from the root r to every vertex in the graph. The *minimum branching problem* is: given a directed graph with weights on the edges and a specified vertex r, find a branching of minimum weight rooted at r.

Show that Dijkstra's shortest paths algorithm *does not* solve this problem. Specifically, give a graph where the shortest paths found by Dijkstra's algorithm do not form a minimum weight branching.

Problem 5. One-Two Knapsack Problem (20 points):

The *Knapsack Problem* is: Given a collection of items $I = \{i_1, \ldots, i_n\}$ and an integer K where each item i_j has a weight w_j and a value v_j, find a subset of the items with weight at most K which maximizes the total value of the set. More formally, we want to find a subset $S \subseteq I$ such that $\sum_{i_k \in S} w_k \leq K$ and $\sum_{i_k \in S} v_k$ is as large as possible.

We define the *density* of d_j of item i_j to be $d_j = v_j/w_j$. A natural greedy algorithm for the knapsack problem is to consider the items in order of decreasing density, and place each item into the knapsack if there is still sufficient space for the item.

For this problem, we restrict the weights of the items to be either 1 or 2. For convenience, we assume the capacity K of the knapsack is an even number.

a) Given an example that shows that the greedy algorithm based on sorting items by density does not necessarily give an optimal solution, even if the weights are restricted to 1 and 2.

b) Describe an efficient algorithm that finds an optimal solution to the knapsack problem when the weights are restricted to 1 and 2.

c) Provide a justification that your algorithm is correct.