COMPSCI330 Design and Analysis of Algorithms
Practice Final

Guidelines

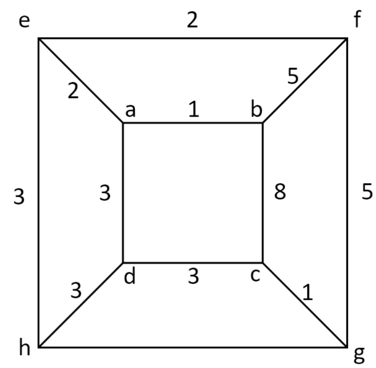
- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure and analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice. If the running time of your algorithm is worse than the suggested running time, you might receive partial credits.

Name: ____________________________ Duke ID: ____________________________

Problem 1 (Warm-up)**.** (20 points) (a) (10 points) Solve the following recurrence relation:

$$T(n) = 3T(n/2) + 4T(n/4) + n^2.$$

(b) (10 points) Run Dijkstra on the following graph starting at vertex a. List the vertices visited (in the order that they are visited) and the shortest path distance from a to each vertex.



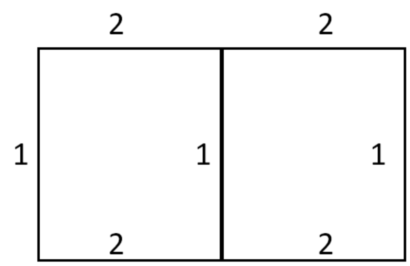
Problem 2 (Double Knapsack)**.** (20 points) There are n items with weights $w_1, w_2, ..., w_n \geq 0$ and value $v_1, v_2, ..., v_n \geq 0$ (weights w_i's are integers). We would like to put these items into two knapsacks with capacities m_1, m_2 (both m_1, m_2 are integers) respectively. The goal is to maximize the total value for all items in the knapsacks, while not exceeding the capacity limit. The items are not divisible (meaning the item can only be put as a whole in one of the knapsacks).

(a)(6 points) Describe the state for a dynamic programming algorithm to solve the problem. You need to define both the notation for the states and give an English explanation for what an arbitrary state means (such as $a[i, j]$ is the maximum of ......).

(b)(8 points) Give the transition function and the base cases for the states you defined.

(c) (6 points) Design the algorithm and analyze its running time. Your algorithm should run in time $O(nm_1m_2)$.

Problem 3 (Graph Components)**.** Given a connected, undirected graph G with n vertices and m edges. Suppose the edges have weights $w(u, v) \geq 0$, and we are also given an integer $1 \leq k \leq n$. The goal is to select a subset of edges so that if all the other edges are removed, the graph has exactly k components. We also want to minimize the total weight of the edges selected. As an example, see the figure below: if $k = 3$ we should select the 3 vertical edges and the total cost is 3.

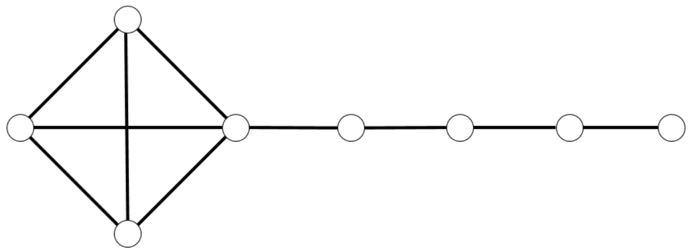


(a) (12 points) Design an algorithm for this problem and analyze its running time. Your algorithm should run in $O(m \log n)$ time.
(Hint: When $k = 1$, this problem is just the minimum spanning tree problem. Recall the algorithms for MST.)

(b) (8 points) Prove the correctness of your algorithm.

Problem 4 (Basketball Traffic)**.** (20 points) Suppose there is a road network represented as a directed graph. Each edge has a capacity $c_{i,j}$ which shows the number of cars it can transport within 1 unit of time. Suppose there are three types of vertices in the graph - parking lots, neighborhoods and intermediate points. There is now a basketball game. For a neighborhood vertex i, there are a_i people that wants to go to the game. All of these people need to go to one of the parking lots. Parking lot j has capacity b_j. Assume driving is actually very fast (that is, does not take any time) as long as the roads are not congested, write a linear program whose optimal solution gives the shortest amount of time needed for all the people to reach the parking lots.

Problem 5 (KITE)**.** (20 points) We say a graph is a kite of size t (for an even number t) if it has t vertices $v_1, v_2, ..., v_t$. For the first $t/2$ vertices, there are edges between any pair of them; for the last $t/2$ vertices, the only edges they have are (v_i, v_{i+1}) where $i = t/2, t/2+1, ..., t-1$. See the figure below for a kite of size 8.



In the KITE problem, we are given a graph G and a number t, and we want to decide if there are t vertices in the graph G that forms a kite of size t. (The set of t vertices form a kite, if after removing all the other vertices, the remaining graph is a kite of size t.)

We are going to show the KITE problem is NP-complete. Recall that in the CLIQUE problem, we are given a graph G and a number k, and the goal is to decide whether there are k vertices that are pairwisely connected with each other. CLIQUE problem is NP-complete.

(a) (3 points) To show KITE is NP-hard based on the fact that CLIQUE is NP-complete, what is the correct direction of reduction?

(b) (5 points) Prove that KITE is in NP.

(c) (12 points) Do a reduction (related to the CLIQUE problem) to show KITE is NP-hard.