CS 170 Homework 6

Due **Saturday 3/8/2025, at 10:00 pm (grace period until 11:59pm)**

1 Study Group (1 point)

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

2 Happy Trees (13 points)

Let $G = (V, E)$ be an undirected graph with edge weights $\{w_e\}_{e \in E}$ **that are all distinct**. Call an edge $e \in E$ *happy* if all cycles containing e have an edge with weight greater than w_e.

(a) (5 points) We will show that the MST of G is unique, and in particular, is made up of all the happy edges of G.

(i) Given an MST T, prove that all its edges are happy edges.

(ii) Given an MST T, prove that all happy edges in G are in T.

(b) (3 points) True or False: The minimum spanning tree of G includes the minimum-weight edge in every cycle in G. If true, write a proof. If false, draw a counterexample with at most 5 vertices, and label (i) the MST T and (ii) a cycle in which the minimum-weight edge is not included in T.

(c) (5 points) Suppose we compute the MST T on G, and then afterwards realized that we had the incorrect weight w_e for an edge $e \in T$ and that its true weight is $w'_e > w_e$. You should assume this new w'_e is distinct from all the edge weights in $\{w_e\}_{e \in E}$.

(i) Give an algorithm for computing the MST T' under the true weights by only removing at most one edge and adding at most one edge to T.

(ii) Prove that your algorithm is correct.

Hint: First show that all other edges in G that were previously happy will still be happy.

3 Firefighters (11 points)

PNPLand is made of n cities that are numbered from $0, 1, \ldots, n-1$, which are connected by two-way roads. You are given a matrix D such that, for each pair of cities (a, b), $D[a][b]$ is the distance of the shortest path between a and b. The distances $D[a][b]$ are metric, so you can assume that the triangle inequality always holds: $0 \leq D[a][b] \leq D[a][c] + D[c][b]$ for all a, b, c.

We want to pick s distinct cities and build fire stations there. For each city without a fire station, the response time for that city is given by distance to the nearest fire station. We define the response time for a city with a fire station to be 0. Let R be the maximum response time among all cities. We want to create an assignment of fire stations to cities such that R is as small as possible.

In this problem, we'll analyze an efficient greedy approach to solving this problem that will not always give the optimal solution, but that we can prove is a "good enough" approximation.

We can formalize the problem as follows: Suppose the optimal assignment of fire stations to cities produces response time R_{opt}.

(a) (2 points) Consider the following algorithm: given positive integers n, s and the 2D matrix D as input, start by building the first fire station at an arbitrary city, e.g. at city 1. Then, for the remaining $s-1$ fire stations, repeatedly find the city with the largest response time and build a fire station there.

Analyze the runtime of this greedy algorithm.

(b) (9 points) Prove that this greedy algorithm achieves an approximation factor of 2, i.e. that it always achieves a response time of $R_g \leq 2 \cdot R_{\text{opt}}$. You may devise your own proof or follow the outline below:

Let the fire stations found by the greedy solution be at $g_1, \ldots, g_s$ and let the fire stations of some optimal solution be at $\omega_1, \ldots, \omega_s$. Assume for the sake of contradiction that $R_g > 2 \cdot R_{\text{opt}}$, i.e. there exists some city v^* whose response time is greater than $2R_{\text{opt}}$.

(i) Show for any i, j that $D[g_i][g_j] > 2 \cdot R_{\text{opt}}$.

(ii) Show that for any i, there is at most one j such that $D[\omega_i][g_j] \leq R_{\text{opt}}$.

(iii) Show that for any i, there is at least one j such that $D[\omega_i][g_j] \leq R_{\text{opt}}$. Conclude that for any i, there is **exactly** one j such that $D[\omega_i][g_j] \leq R_{\text{opt}}$.

(iv) Now, consider the city v^* defined above. Say it is served by some fire station ω_i in the optimal solution (that is, $D[\omega_i][v^*] \leq R_{\text{opt}}$). Use the previous parts to show that there must exist a g_j that is at most distance $2 \cdot R_{\text{opt}}$ away from it.

(v) Conclude that the greedy algorithm indeed achieves a response time of $R_g \leq 2 \cdot R_{\text{opt}}$.

4 Cooper's Pipe (8 points)

Cooper has a copper pipe of length n inches and an array of nonnegative integers that contains prices of all pieces of size at most n. He wants to find the maximum value he can make by cutting up the pipe and selling the pieces. For example, if the length of the pipe is 8 and the values of different pieces are given as following, then the maximum obtainable value is 22 (by cutting in two pieces of lengths 2 and 6).

length	1	2	3	4	5	6	7	8
price	1	5	8	9	10	17	17	20

Give an efficient dynamic programming algorithm so Cooper can find the maximum obtainable value given any pipe length and set of prices.

(a) (6 points) Describe your algorithm.

(b) (2 points) Analyze the runtime of your algorithm.

5 Arbitrage (11 points)

Shortest-path algorithms can also be applied to currency trading. Suppose we have n currencies $C = \{c_1, c_2, \ldots, c_n\}$: e.g., dollars, Euros, bitcoins, dogecoins, etc. For any pair of currencies c_i, c_j, there is an exchange rate $r_{i,j}$: you can buy $r_{i,j}$ units of currency c_j at the price of one unit of currency c_i. Assume that $r_{i,i} = 1$ and $r_{i,j} \geq 0$ for all i, j.

The Foreign Exchange Market Organization (FEMO) has hired Oski, a CS170 alumnus, to make sure that it is not possible to generate a profit through a cycle of exchanges; that is, for any currency $i \in C$, it is not possible to start with one unit of currency i, perform a series of exchanges, and end with more than one unit of currency i. (That is called *arbitrage*.)

More precisely, arbitrage is possible when there is a sequence of currencies $c_{i_1}, \ldots, c_{i_k}$ such that $r_{i_1,i_2} \cdot r_{i_2,i_3} \cdot \cdots \cdot r_{i_{k-1},i_k} \cdot r_{i_k,i_1} > 1$. This means that by starting with one unit of currency c_{i_1} and then successively converting it to currencies $c_{i_2}, c_{i_3}, \ldots, c_{i_k}$ and finally back to c_{i_1}, you would end up with more than one unit of currency c_{i_1}. Such anomalies last only a fraction of a minute on the currency exchange, but they provide an opportunity for profit.

We say that a set of exchange rates is arbitrage-free when there is no such sequence, i.e. it is not possible to profit by a series of exchanges.

(a) (7 points) Give an efficient algorithm for the following problem: given a set of exchange rates $(r_{i,j})_{i,j \in n}$ which is *arbitrage-free*, and two specific currencies a, b, find the most profitable sequence of currency exchanges for converting currency a into currency b. That is, if you have a fixed amount of currency a, output a sequence of exchanges that gets you the maximum amount of currency b.

Hint: $\log(xy) = \log(x) + \log(y)$.

(i) Describe your algorithm.

(ii) Prove the correctness of your algorithm.

(iii) Analyze the runtime of your algorithm.

(b) (4 points) Oski is fed up with manually checking exchange rates and has asked you for help to write a computer program to do his job for him. Give an efficient algorithm for detecting the possibility of arbitrage.

(i) Describe your algorithm.

(ii) Analyze the runtime of your algorithm.

