CSE 101: Homework #4

Due on May 9, 2024 at 23:59pm

Professor Jones

Ray Tsai
A16848188

Problem 1

You are given a connected graph G with n vertices and with positive edge weights $w : E \to \mathbb{R}^+$. You wish to find a connected subgraph of G with $n - 1$ vertices that has minimum total cost.

Proof. □

Problem 2

Suppose you are managing a computer network of n computing sites. Between some pairs of computing sites, there is a link that has a positive initialization time. When you turn on the entire network, all links start initializing at the same time. The whole network is not operational until all links have been initialized. You wish to remove links so that the network stays connected yet you have minimized the maximum initializion time.

The network is given to you as a connected undirected graph with positive edge weights. You can assume that $|E| = O(|V|)$.

Design an algorithm that achieves this goal.

Proof. Consider running Kruskal's algorithm and return the edge set X.

We already know Kruskal's algorithm takes $O(|E| \log |V|)$ time.

We now give a justification for the correctness of the algorithm. Kruskal's algorithm furnishes a connected network, so it remains to show that X is optimal. Let G_n be the graph formed by the edge set at the end of the nth iteration, and let A_n be the graph formed by the edges considered in the first n iterations. We show that A_n is disconnected if G_n is disconnected. Suppose for the sake of contradiction that A_n is connected but not G_n. Notice that G_n is a subgraph of A_n. Consider the set of edges S in A_n but not in G_n. For all $e \in S$, e does not connect any components in G_n, otherwise e would have been added to G_n by the algorithm. But then adding all edges in S to G_n yields a connencted graph A_n, contradiction.

Now suppose for sake of contradiction that there exists a network of a smaller initialization time. That is, there exists an edge set E which forms a connect graph and $\max E < \max X$. Let H be the graph formed by all edges at most $\max E$. We know $H = A_n$ for some n and H is connected. By our result above, $G_n \subset A_n = H$ is connected. But then $\max X = \max E(G_n) \leq \max E$, contradiction. □

Problem 3

You are given two sets of n points each on the number line: $(A[1], \ldots, A[n])$ and $(B[1], \ldots, B[n])$ (all values are different and each list is sorted in increasing order.)

You wish to pair up the points (one from the first list and one from the second list):

$$[(A[1], B[i_1]), (A[2], B[i_2]), \ldots, (A[n], B[i_n])]$$

(such that each point is in exactly one pair.)

You wish to pair them up in such a way to minimize:

$$\sum_{k=1}^{n} |A[k] - B[i_k]|$$

(a) Describe this problem as we have done in class in terms of:

- **Input:** Arrays A, B of size n in increasing order and all values are distinct.
- **Solution Format:** $[(A[1], B[i_1]), (A[2], B[i_2]), \ldots, (A[n], B[i_n])]$.
- **Constraints:** $i_\alpha \neq i_\beta$ if $\alpha \neq \beta$.
- **Objective:** Minimize cost value $\sum_{k=1}^{n} |A[k] - B[i_k]|$.

(b) **Candidate Greedy Strategy I:** Find the pair $A[i], B[j]$ that is the closest (with the smallest overall $|A[i] - B[j]|$ distance) and pair $(A[i], B[j])$ (break ties by choosing the smaller value of $A[i]$.) Remove $A[i]$ from the first list and remove $B[j]$ from the second list and repeat on the remaining points until all points are paired.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. Consider $A = [1, 4]$, $B = [-3, 2]$. The algorithm would first pair up 1 and 2, then 4 and -3. Hence, the algorithm would return $x = [(1, 2), (4, -3)]$, which has a cost value of 8. But then there exists a pairing $y = [(1, -3), (4, 2)]$, which has a cost value of 6. Hence, this strategy is not optimal. □

(c) **Candidate Greedy Strategy II:**

Pair up $(A[1], B[1]), (A[2], B[2]), \ldots, (A[n], B[n])$.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. This algorithm is correct. Let σ be a non-trivial permutation of B. There exists some i such that $\sigma(i) \neq i$. Let k be the smallest such index. We know $\sigma(k) > k$. Let $l = \sigma^{-1}(k)$. Since k is the smallest index which deviates, l must be larger than k.

We now show that

$$|A[k] - B[l]| + |A[l] - B[l]| \geq |A[k] - B[k]| + |A[k] - B[k]| \tag{1}$$

by considering the following 6 possible cases:

Case 1: $A[k] \leq A[l] \leq B[k] \leq B[l]$.

$$(B[l] - A[k]) + (B[k] - A[l]) = (B[k] - A[k]) + (B[l] - A[l]).$$

Case 2: $A[k] \leq B[k] \leq A[l] \leq B[l]$.

$$\begin{aligned}(B[l] - A[k]) + (A[l] - B[k]) &= (B[k] - A[k]) + (B[l] - A[l]) + 2(A[l] - B[k]) \\ &\geq (B[k] - A[k]) + (B[l] - A[l])\end{aligned}$$

Case 3: $A[k] \leq B[k] \leq B[l] \leq A[l]$.

$$\begin{aligned}(B[l] - A[k]) + (A[l] - B[k]) &= (B[k] - A[k]) + (A[l] - B[l]) + 2(A[l] - B[l]) \\ &\geq (B[k] - A[k]) + (B[l] - A[l])\end{aligned}$$

Case 4: $B[k] \leq A[k] \leq A[l] \leq B[l]$.

$$\begin{aligned}(B[l] - A[k]) + (A[l] - B[k]) &= (A[k] - B[k]) + (B[l] - A[l]) + 2(A[l] - A[k]) \\ &\geq (B[k] - A[k]) + (B[l] - A[l])\end{aligned}$$

Case 5: $B[k] \leq A[k] \leq B[l] \leq A[l]$.

$$\begin{aligned}(B[l] - A[k]) + (A[l] - B[k]) &= (A[k] - B[k]) + (A[l] - B[l]) + 2(B[l] - A[k]) \\ &\geq (B[k] - A[k]) + (B[l] - A[l])\end{aligned}$$

Case 6: $B[k] \leq B[l] \leq A[k] \leq A[l]$.

$$(A[k] - B[l]) + (A[l] - B[k]) = (A[k] - B[k]) + (A[l] - B[l])$$

Hence, by (1), we may achieve a lower objective value by swapping the value of k and l from σ. But then a lower objective value can be obtained whenever the permutation is non-trivial. Therefore, by repeating the swapping process above, the permutation will eventually become a trivial permutation, and thus

$$\sum_{k=1}^{n} |A[k] - B[k]| \leq \sum_{k=1}^{n} |A[k] - B[\sigma(k)]|.$$

□

(d) **Candidate Greedy Strategy III:** Let $B[j]$ be the closest point to $A[1]$ in the B list (break ties by choosing the lower B value). Pair up $(A[1], B[j])$ and remove $A[1]$ and $B[j]$ from the lists and continue with $A[2]$ until all points are paired.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. Consider $A = [1, 4]$, $B = [-3, 2]$. The algorithm would first pair up 1 and 2, then 4 and -3. Hence, the algorithm would return $x = [(1, 2), (4, -3)]$, which has a cost value of 8. But then there exists a pairing $y = [(1, -3), (4, 2)]$, which has a cost value of 6. Hence, this strategy is not optimal. □

Problem 4

Suppose you are driving along a road in an electric car. The battery of the electric car can bring you $x[0]$ miles. There are battery stations along the way at positive positions $D[1], \ldots D[n]$ (in sorted order.) Each battery station can *replace* your battery and give you a new battery that can bring you a certain number of miles. The distances of the batteries are given in the array $x[0], x[1], \ldots, x[n]$

You wish to start at position 0 with a full battery and end at position $D[n]$ replacing the fewest batteries.

(a) Describe this problem as we have done in class in terms of:

- **Input:** A sorted list of n battery station positions and a list of $n+1$ distances of the batteries.
- **Solution Format:** A set of indices $X \subseteq \{1, \ldots, n\}$.
- **Constraints:** For all $k \in X$,
$$\sum_{\substack{i \in X \\ i<k}} x[i] \geq D[k].$$

Objective: Minimize $|X|$.

(b) **Candidate Greedy Strategy I:** Travel to the farthest battery station without exceeding $x[0]$ miles. Replace the battery at that station and repeat the process starting from that station until you can reach position $D[n]$.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. Consider $D = [0, 1, 3, 6]$, $x = [3, 5, 1, 0]$. The strategy would traval to station 2 first and replace to a battery with distance $x[2] = 1$. But then the next station is 3 unit distances away, so the strategy fails to arrive at the destination. However, there exists solution $X = \{1, 3\}$ which could reach the destination, and thus the strategy is not optimal. □

(c) **Candidate Greedy Strategy II:**

Travel to the battery station with the largest $x[i]$ value without exceeding $x[0]$ miles. Replace the battery at that station and repeat the process starting from that station until you can reach position $D[n]$ and then go directly there.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. Consider $D = [0, 1, 2]$ and $x = [2, 1, 0]$. The strategy would first stop at station 1 then arrive at the destination. However, the initial battery $x[0]$ is sufficient to reach the destination, so no intermediate stops are required. Thus, this strategy is not optimal. □

(d) **Candidate Greedy Strategy III:**

Travel to the battery station with the largest $D[i] + x[i]$ value (in other words, the battery that can take you the farthest down the road.) Replace the battery at that station and repeat the process starting from that station until you can reach position $D[n]$ and then go directly there.

Either prove that this strategy always yields an optimal solution or give a counterexample to show that it is not always optimal.

Proof. Consider $D = [0, 1, 2]$ and $x = [2, 3, 0]$. Since $D[2] + x[2] < D[1] + x[1]$, the strategy would stop at station 1 first before arriving at station 2. But then the initial battery $x[0]$ is sufficient to reach the destination, so no intermediate stops are required. Thus, this strategy is not optimal. □