CS 170 DIS 10

Released on 2019-04-01

1 MST Mishap

You found the MST of a huge graph $G = (V, E)$, but afterwards you realized that you made a small mistake: the weight of a particular edge e is not $w(e)$, but instead $w'(e)$. Give a linear-time algorithm for finding the true MST. For each of the four cases, clearly describe your algorithm (with pseudocode if desired). Your algorithm must run in linear time.

(a) Claim : Let T be the original tree. There exists an MST T' such that T and T' differ on at most 2 edges.

(b) Case 1: e is in the current MST and $w'(e) < w(e)$.

(c) Case 2: e is in the current MST and $w'(e) > w(e)$.

(d) Case 3: e is not in the current MST and $w'(e) > w(e)$.

(e) Case 4: e is not in the current MST and $w'(e) < w(e)$.

2 Scheduling Meetings : Greedy and dynamic

You are the scheduling officer for the CS department. The department has one conference room, and every day you receive a list L of n requests for it, in no particular order: $L = (s_1, e_1), \ldots, (s_n, e_n)$. Each item in the list (s_i, e_i) is a pair of integers representing start and end times, respectively, with $s_i < e_i$. Your goal is to schedule as many meetings as possible in one day, with the constraint that meetings cannot conflict with each other (the intervals for the scheduled meetings must be disjoint). Assume for simplicity that all of these integers are distinct.

(a) You consider a greedy algorithm for this problem: *Choose the meeting that starts first, and delete all conflicting meetings; repeat.* Give a simple example with just three meetings showing that this cannot work.

(b) Now find an optimal greedy algorithm: given the list L as specified above, your algorithm should return a list of non-conflicting meetings of maximum length.

(c) Briefly justify the correctness of your algorithm.

(d) What is the running time of this algorithm, given the unsorted list of n meetings? Justify briefly.

(e) Now suppose each meeting also has a positive integer weight w_i, and you want to schedule the set of meetings that maximize the total weight. Your input is the list L of n requests, again in no particular order, with each element now formatted as (s_i, e_i, w_i). Since the greedy algorithm no longer works, you design a dynamic programming algorithm to return the maximum weight achievable. (*Note*, you do not need to return the set of meetings that achieve it).

First, define your subproblems in words:

For $i = 1, \ldots, n$ $W[i] =$ ____

(f) Give clear pseudocode for this problem. You do not need to justify your solution. What is the runtime of your algorithm?

3 Midterm Prep: Short Answer

Answer each question below *concisely*

(a) Suppose that in a weighted graph with negative weights we know that the shortest path from s to t has at most $\log|V|$ edges. Can we find this shortest path in $O(|V|^2 \log|V|)$ time?

(b) True/False : In a Huffman code, if there is a leaf of depth 1 and a leaf of depth 3, then there must also be a leaf of depth 2.

(c) True/False : In a Huffman code where all letter frequencies are distinct, the third least frequent letter has depth either equal to that of the two least frequent letters, or one less.

(d) True/False : If a linear program is unbounded, then its dual must be unbounded.

(e) True/False : If the capacities of a max flow problem are multiples of $\frac{1}{2}$, the value of the optimum flow will also be a multiple of $\frac{1}{2}$.

(f) True/False : If we add an integer $k > 0$ to all capacities of a max-flow problem, the max flow is also increased by exactly k.

(g) True/False : If we multiply all capacities of a max flow problem by an integer $k > 0$, the max flow is also multiplied by exactly k.

4 Edge Disjoint Paths

Given a directed graph $G = (V, E)$ and two vertices in it a and b, find out the maximum number of edge disjoint paths from a to b. Two paths are said to be edge disjoint if they don't share any edge.