COL351: Analysis and Design of Algorithms

Tutorial Sheet - 2

August 20, 2022

Question 1

(a) Let T be a spanning tree of $G = (V, E)$ and $e_0 = (x, y)$ be any edge not lying in T. For any $a, b \in V$, let $\textsc{path}(a, b, T)$ denote the unique path from a to b in T. Then prove that on replacing any edge lying on $\textsc{path}(x, y, T)$ with e_0 we get another spanning tree of G.

(b) Prove the correctness of the following algorithm to compute MST of a weighted graph $G = (V, E, wt)$.

```
Initialize T to be any arbitrary spanning tree of G.
while there exits e_0 = (x, y) ∉ T and e ∈ PATH(x, y, T) satisfying wt(e_0) ⪇ wt(e) do
    Replace e with e_0 in T;
end
Output T.
```

Algorithm 1: MST($G = (V, E, wt)$)

(c) Consider a weighted graph $G = (V, E, wt)$. Prove that an edge $e_0 = (x, y) \in E$ doesn't lie in any MST of G if and only if there exists a path connecting x and y consisting only of edges whose weight is strictly less than $wt(e_0)$.

Question 2 Let G be an undirected graph with n vertices and m edges. A vertex x in G is said to be *cut-vertex* if there are vertices u, v different from x, such that u and v are disconnected in $G \setminus x$. Let T be a DFS tree of G.

i) Show that a leaf node of T cannot be a cut-vertex.

ii) Show that root of T is cut-vertex iff it has at least two children.

iii) Prove that an internal node x is cut vertex iff it has a child, say y, in DFS tree T satisfying $High\text{-}point(y) \geq Level(x)$.

iv) Devise an $O(n + m)$ time algorithm to find all cut-vertices of a graph G.

Question 3 You are starting a start-up that needs to obtain licenses for n different pieces of cryptographic software. Due to regulations, you can only obtain these licenses at the rate of at most one per month.

Each license is currently selling for a price of 100 INR.

However, they are all becoming more expensive according to exponential growth curves: in particular, the cost of license j increases by a factor of $r_j > 1$ each month, where r_j is a given parameter. This means that if license j is purchased after t months from now, it will cost $(100 \cdot r_j^t)$. Further, all the price growth rates are distinct, that is $r_i \neq r_j$ for licenses $i \neq j$ even though they start at the same price of 100 INR.

Give an algorithm that takes the n rates of price growth $r_1, r_2, \ldots, r_n$ and computes an order in which to buy the licenses so that the total amount of money spent is minimized. The running time of the algorithm should be polynomial in n.

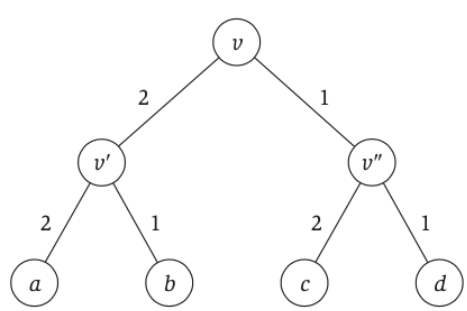
Question 4 There is a complete binary tree with n leaves representing a circuit. Here n is a power of two. Each edge e of the tree has an associated delay $d_e \geq 0$. The root generates a clock signal which is propagated along the edges to the leaves. The time taken for a message to reach from the root to a given leaf is the sum of the delays of all the edges on the path from the root to the leaf.

Now, we want all the leaves to be completely synchronized, i.e. they receive the signal at the same time. To make this happen, we will have to increase the delay of certain edges, so that all root-to-leaf paths have the same delay. Our goal is to achieve this synchronization in a way that keeps the total delay enhancement in the circuit to be minimum.

(a) Design a greedy step that transforms the given instance to a smaller instance of the same problem.

(b) Next establish a relation between optimal solution of given instance and the optimal solution of smaller instance.

(c) Design an $O(n)$ time greedy algorithm based upon (a) and (b), and prove the correctness.



Question 5 Recall that prefix-free encoding is a way of encoding symbols such that no code word is a prefix (initial segment) of another code.

(a) Compute the best prefix-free encoding for an n-length message with 6 characters and Frequency vector $F = (45, 13, 12, 16, 9, 5)$.

(b) Suppose you aim to compress a file with 16-bit characters such that the maximum character frequency is strictly less than twice the minimum character frequency. Prove that the compression obtained by Huffman encoding, in this case, is same as that of the ordinary fixed-length encoding.

(c) Present an $O(n \log n)$ time implementation for Huffman encoding. You can assume that all the frequencies lie in the range $[1, cn]$, for some constant $c > 0$.