Recitation 2

February 27, 2013

Problem

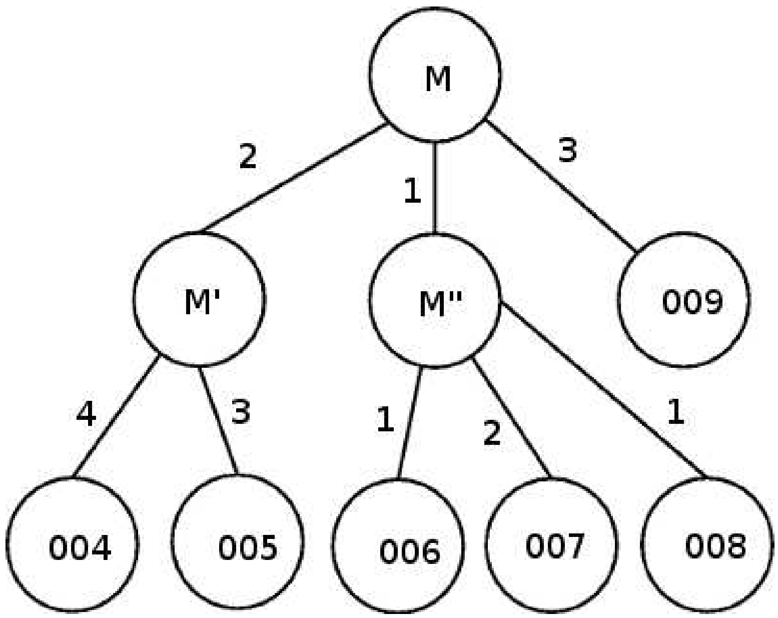

Figure 1: One of the British Secret Service Super-Secure Communication Networks.

You have just been hired by the British Secret Service (BSS) and your job is to synchronize the efforts of many **"00"** agents around the globe. Your communication network is a rooted tree $T = (V, E)$ comprised of vertices V and edges E with a root vertex $M \in V$ (See the figure for an example). This network stretches vast distances and the communication time across each edge varies depending on the edge. Then the time for a message to reach an agent x is the sum of communication times across all edges on the path from the root M to the leaf x. Your job is to send messages via this communication network so that all the messages are received by the agents simultaneously. Because the communication times vary from edge to edge, adding a delay d_e to some edges may be necessary to ensure that the agents receive messages at the same time (i.e. that the communication time from the root to each leaf is the same). However, the IT department at BSS does not like these delays. They have requested you add as little delay as possible (i.e. minimize $\sum_{e \in E} d_e$).

To summarize, given a tree T and communication times t_e, you must find an assignment of delays d_e minimizing $\sum_{e \in E} d_e$ and such that the communication times on all root-leaf paths are equal.

1. An assignment of d_e is said to be *feasible* if all the agents receive a message at the same time. Assuming the tree from the figure with edge times as denoted, give two feasible delay assignments to the edges, one which is not optimal and one which is optimal.
2. Design an algorithm for assigning the optimal edge delays.
3. Prove the correctness of your algorithm.
4. Prove a running time of $O(n)$.

Solution

1

Left to the reader.[1]

2 The Algorithm

We are given a tree $T = \langle V, E\rangle$, with a root $m \in V$ and edge lengths L_e for $e \in E$. Let C_v be the set of children of any vertex $v \in V$ (note that if these are not given, they can be calculated in linear time using depth first search). Let m_v be a value for each vertex $v \in V$. It will be assigned by the algorithm and store maximum communication time to a leaf of v's subtree. The following algorithm assigns the minimum delay values d_e to each edge $e \in E$ when executed on the root node, i.e. Assign-Delay($\langle V, E\rangle, m$) . Note, for clarity, any edge $e \in E$ may be represented by a tuple $\langle v, w\rangle$ where $v, w \in V$.

Assign-Delay($\langle V, E\rangle, v$)

1 **if** $|C_v| = 0$
2   **then**
3         $m_v \leftarrow 0$
4   **else**
5   **for** $\langle v, w\rangle \in C_v$
6   **do** Assign-Delay($\langle V, E\rangle, w$)
7   **for** $\langle v, w\rangle \in C_v$
8   **do** $m_v \leftarrow \max(m_v, L_{\langle v,w\rangle} + m_w)$
9   **for** $\langle v, w\rangle \in C_v$
10   **do** $d_{\langle v,w\rangle} \leftarrow m_v - m_w - L_{\langle v,w\rangle}$

3 Correctness

Claim. Assign-Delay terminates.[2]
Proof: Each call of the Assign-Delay works on the child of some node. The input, $\langle V, E\rangle$, is assumed to be a tree, therefore at some point each call reaches a leaf and the clause on line 1 becomes true.

Claim. Assign-Delay satisfies the feasibility criteria that the distance from the root to all leafs is equal.
Proof: The approach will be by induction on the height of the tree (note that this precisely coincides with recursive call structure). The base case with no children is true because the communication time is trivially 0. Hence, we can assume that the distance from our children to their leaves is equal, and we need to show that the delays we assign will maintain equal communication to our children. More formally, assume we are at vertex v and the time from each of the children $w \in C_v$ to their leaves is m_w. We must show that, for some constant c, $L_{\langle v,w\rangle} + d_{\langle v,w\rangle} + m_w = c\ \forall w \in C_v$. From line 10 we know that $d_{\langle v,w\rangle} = m_v - m_w - L_{\langle v,w\rangle}$. Combining these facts we have that $c = m_v\ \forall w \in C_v$. Hence by design, the communication from v to any of its leaves is exactly m_v.

Claim. Assign-Delay satisfies the optimality criteria that $\sum_{e\in E} d_e$ is minimal.
Proof: The approach is by contradiction. Assume some alternate weighing d^* was better than the weighting produced by Assign-Delay, d. That is $\sum_{e\in E} d^*_e < \sum_{e\in E} d_e$. This implies that $\exists\ e \in E$ such that $d^*_e < d_e$. Lets consider the point at which Assign-Delay chooses d_e. The argument implies that,

$$d^*_{\langle v,w\rangle} < d_{\langle v,w\rangle} = m_v - m_w - L_{\langle v,w\rangle}$$

Because $L_{\langle v,w\rangle}$ is fixed, there are two possibilities: $m^*_v < m_v$ or $m^*_w < m_w$. The later case would imply yet another smaller edge case so without loss of generality we can assume $m^*_w = m_w$ and need only consider $m^*_v < m_v$. However, combing that fact with the definition of m_v from line 8, we have,

$$m^*_v < \max_{\langle v,w\rangle \in C_v} L_{\langle v,w\rangle} + m_w$$

[1] Making such a remark in an assignment is worth 0 points.

[2] Termination is often obvious and unnecessary to state explicitly. In the case of recursive algorithms a simple explanation is a nice touch.

This violates the feasibility criteria because at least one edge will be shorter than it needs to be to ensure the distance to the leafs are all equal. This implies the existence of d^* is a contradiction.

TODO: summary: satisfies blah blah blah

4 Runtime

Answer 1

The ASSIGN-DELAY method is simply a modified version of Depth First Search (DFS). At each node in the search a constant amount of work is performed (note the simple arithmetic in lines 3, 8, and 10 are all constant time). Hence, the DFS retains its running time of $O(|V| + |E|)$ because the input is a tree, and thus the running time reduces to $O(|V|)$.

Answer 2

Based on the termination argument it is clear that ASSIGN-DELAY executes once for each node in V. The non-recursive body of the method (the loops on lines 7 and 9) takes $2|C_v|$ time. Combining these facts we have,

$$\sum_{n \in V} 2|C_v|$$

The sum of all the children of V exactly covers E. Hence, $\sum_{n \in V} 2|C_v| = 2|E|$. Because this is a tree the running time can be restated as $O(|V|)$