CSCI-GA.1170-001/002 Fundamental Algorithms	December 3, 2015

Solutions to Problem 1 of Homework 10 (8 points)

Name: GOWTHAM GOLI (N17656180) *Due: Tuesday, December 1*

You are given a directed graph $G = (V, E)$ on n nodes and m edges, where the node set $V = A \cup B$ consists of two disjoint subsets A and B of sizes n_1 and n_2 (so $n = n_1 + n_2$). Nodes in A are "healthy", while nodes in B are "infected". Given a source $s \in A$, your goal is to compute the shortest distance from s to every other healthy node which can pass through *at most one* infected node (i.e., if the path from s to v contains at most one infected u, this is OK, but if it contains two or more, this path is not allowed when computing the shortest distance).

Define the following *directed* graph $G' = (V', E')$ on $2n_1 + n_2$ nodes. The vertex set of V' of G' is $V' = A_1 \cup B \cup A_2$, where A_1 and A_2 are two copies of healthy nodes A. Two nodes in A_1 are connected in G' if and only if they are connected in G, and the same between two nodes in A_2. The nodes in B are more interesting. For every original incoming edge $(a, b) \in E$, where $a \in A$ and $b \in B$, we will put an edge (a_1, b) in E', where a_1 is the copy of a in A_1. Similarly, for every original outgoing edge $(b, a) \in E$, where $a \in A$ and $b \in B$, we will put an edge (b, a_2) in E', where a_2 is the copy of a in A_2.

(a) (2 pts) Let n', m' be the number of vertices and edges in G'. Show that $n' \leq 2n$ and $m' \leq 2m$.

Solution:

$n' = 2n_1 + n_2 = n + n_1 < n + n = 2n \implies n' < 2n$

Let number of edges within A be m_1 and the number of edges within B be m_2 and the number of edges between A and B be m_3. Hence $m = m_1 + m_2 + m_3$.

$\therefore |E'| = m' = m_1 + m_1 + m_3 < m_1 + m_1 + m_3 + (m_2 + m_2 + m_3) = 2m \implies m' < 2m$ $\square$

(b) (4 pts) Recall our original problem of computing the required shortest distance in G from s to every other healthy node $a \in A$ which can pass through *at most one* infected node $b \in B$. Call this distance $a[dis]$. Let s_1 and s_2 be the two copies of s in G'. Using one "appropriate" BFS call on G', show how to compute the values $a[dis]$. Specifically, say what is the starting node (call it s') of your BFS call in G'. Also, after your BFS call computed shortest distances $v'.d$ from s' to v', for every $v' \in V'$, show how to compute the desired values $a[dis]$ for the problem at hand (i.e., write an explicit formula for $a[dis]$ using appropriate $v'.d$ values). Justify your algorithm.

Solution:

Let s' be s_1 i.e the copy of s in A_1 and now call BFS on s' in G'. This gives us the shortest distance from s' to every other node in G' i.e $v'.d$, for every $v' \in V'$.

However we are only interested in $v' \in A_1 \cup A_2$ i.e the copy of all the vertices of A. Consider some vertex $a \in A$. There can be multiple paths from s to a in G i.e the paths containing zero infected nodes or one infected nodes or two infected nodes etc.

Now $a[dis]$ is the shortest distance from s to a containing atmost one infected node. Therefore this can be broken down into two parts i.e the minimum of the shortest distance from s to a containing 0 infected nodes, let it be $a[dis_0]$ or the shortest distance from s to a containing 1 infected nodes, let it be $a[dis_1]$.

$$a[dis] = min\{a[dis_0], a[dis_1]\}$$

Let $a_1 \in A_1$ and $a_2 \in A_2$ be the copies of the vertex $a \in A$. Note that the shortest path from s' to a_1 has 0 number of infected nodes in it as there are no edges from the B to A_1 and the path from s_2 to a_2 has exactly 1 infected node as we need to visit exactly one node in B to reach any node in A_2. Therefore substitute these values of $a_1.d = a[dis_0]$ and $a_2.d = a[dis_1]$ in the above equation

$$a[dis] = min\{a_1.d, a_2.d\}$$

where a_1 and a_2 are the copies of a in A_1 and A_2 □

(c) (2 pts) Show that the running time of your procedure is $O(m + n)$.

Solution:

In the above algorithm we called BFS on s' in G'. This step takes $O(|V'| + |E'|) = O(m + n)$ time and then we take minimum of $a_1.d$ and $a_2.d$ for every $a \in A$. This step takes $O(|A|) = O(|V|) = O(n)$ time. Therefore the total running time of the algorithm is $O(m + n)$ □