Name: Taylor Heilman

CS 275: Spring 2016

1.

Let T be the tree obtained from $G.BFS(v)$ (using the BFS algorithm on graph G with v as the root).

Suppose there is a vertex $w \in V(G)$ such that the shortest distance from v to w is of length n.

Let n = 1:

Then w is adjacent to v, so $w \in C(v)$ and therefore a (v, w) path of length 1 exists in T.

Assume $n = k$ and the minimum length of some $(v, w) path \in G$ is equal to the minimum length of the $(v, w) path \in T$

Let $n = (k + 1)$

Then there exists some vertex $y \in G$ such that the shortest (v, y) path in G is of length (k+1).

According to out assumption, the following is true:

- Suppose some vertex, $e \in T$ exists with the distance of length 1 away from the root v. Following the BFS algorithm, e gets an edge connecting itself to v. Thus a v, e path has been created and has length 1.
- Suppose some vertex, $t \in T$ exists with the distance of length 2 away from v. Following the BFS algorithm, t gets an edge connecting itself to e. Thus a v, t path has been created and has length 2.
- We continue this procedure until we get to vertices with distances of length k away from the root. Hence at least one vertex exists such that there is path of minimum length k that connects v to some vertex $z \in T$.

Now, since vertex w has a vw path of length k+1 in G, there exists a path of length k to some vertex, r adjacent to w in G. By creating an edge connecting r, w a v, w path is created and has length $k + 1$ in T.

Therefore distance is preserved when using BFS.

2.

(WTS: $e_T(v) \leq e_G(v)$)

Suppose there exists some graph T which is a spanning tree of graph G.

Hence, $V(T) = V(G)$ but $|E(T)| \leq |E(G)|$.

So T, the spanning tree of G, has at most the same amount of edges as G ($E(T) = E(G)$), or less edges than G.

Notice the $e_G(v)$ is simply MAX $d_G(v, w)$ where $w \in V(G)$.

Similarly, $d_G(v, w)$ is simply the shortest path between v and w.

A (v, w) path exists in T, and finding the path between $(v, w) \in T$ creates 2 possible cases.

Case 1: The path between $(v, w) \in T$ is the same as the path between $(v, w) \in G$.

In this case $d_G(v, w) = d_T(v, w)$ since they are the same path.

Case 2: The path between $(v, w) \in T$ is different than the path between $(v, w) \in G$.

In this case there exists some vertex x which creates vx and xw paths $\in T$ which do not exist in the (v, w) path $\in G$.

By the Triangle inequality, the distance of the vw path $\in T$ containing the vertex x is $\geq$ to the distance of the (v, w) path $\in G$ which does not contain x.

Therefore $e_T(v) \leq e_G(v)$.

3.

1. Let visited = [v] (v is the root)

2.Let C = N(v) (adjacent vertices to v)

3. Let labels = [v] ('labels' is a list where the index of each vertex is its label number), p(v) = v, b* = v (keeps track of current vertex we are branching from), i = 1.

4. For each $w \in N(b*)$ append w to labels.

5. Delete b* from all adjacency lists , remove b* from C and append b* to visited. Define a new b* to be the vertex in C such that $l[x]$ is minimum. If C contains a vertex that is also in labels (this checks if a vertex adjacent to the current b* has already been labeled by a previous b*, which means a cycle has been found), return True. Else return to step 4. If every vertex of G is contained in visited return false.