CSC373 Winter 2015 Problem Set # 2

Name: Weidong An
Student Number: 1000385095
UTOR email: weidong.an@mail.utoronto.ca
January 20, 2015

(a) No. Consider the following undirected graph (G, w, L):

$$G.V = \{v_1, v_2, v_3\}$$
$$G.E = \{e_1 = (v_1, v_2), e_2 = (v_1, v_3), e_3 = (v_2, v_3)\}$$
$$w(e_1) = w(e_2) = w(e_3) = 1$$
$$L = G.V = \{v_1, v_2, v_3\}$$

G has 3 vertices, so the spanning trees of G have 3 nodes. L contains all the vertices of G. Consider trees with 3 nodes. We can break them into two cases: the root has 2 children or the root has only 1 child. In the case that the root has 2 children, clearly the root can not be a leaf. In the case that the root has only 1 child, the child of the root must have 1 child, so this vertex can not be a leaf. Hence, in any spanning tree of G, there always exists a vertex which can not be a leaf.

(b) The pseudocode are as follows. The idea is fixing Kruskal's algorithm. But instead of sorting the edges based their weights straightly, the algorithm defines a new weight function w' and finds a subset E' of $G.E$ first. The algorithm checks each edge in $G.E$. If neither of the end points is in L, add the edge to E' and maintain the same weight. If one of the end points is in L and the other is not in L, add the edge to E' and increase the weight by *max-weight* defined in line 1 (This is to make the weights of all this kind of edges greater than the edges in the first case). Otherwise, the edge would not be added into E'. Finally, call Kruskal's algorithm on (V, E') and w'.

MST-With-Set-Of-Leaves(G, w, L)

```
1   max-weight = max{w(e)| for e ∈ G.E}
2   E' = ∅
3   for each edge (u, v) ∈ G.E
4       if u ∉ L and v ∉ L
5           w'((u, v)) = w((u, v))
6           E' = E' ∪ {(u, v)}
7       elseif (u ∈ L and v ∉ L) or (v ∈ L and u ∉ L)
8           w'((u, v)) = w((u, v)) + max-weight
9           E' = E' ∪ {(u, v)}
10  call Kruskal's algorithm on (G' = (V, E'), w')
```

(c) Comparing E' with E, we find that E' is obtained by cancelling all the edges connecting two vertices in L. Claim that the edges cancelled can not be in the required spanning trees. Suppose the edge (u, v) where $u, v \in L$ is in a required spanning tree. Then, u and v are leaves of the spanning tree (i.e they should both be connected to only one vertex.). Since u

and v are connected, u and v is not connected to other vertices contradicting the definition of spanning tree. (Or if u and v are the only two vertices in G, one of u and v cannot be a leaf.)

When Kruskal's algorithm is called, it sorts the edges into $e_1, ..., e_k, e_{k+1}, ..., e_m$ where $e_1, ..., e_k$ connects two vertices in $G.V - L$, $e_{k+1}, ..., e_m$connects two vertices one of which is in L and one of which is in $G.V - L$ and $w(e_1) \leq ... \leq w(e_k)$ and $w(e_{k+1}) \leq ... \leq w(e_m)$ by the above algorithm. First, it finds a minimum spanning tree of $(G.V - L, \{e_1, ..., e_k\})$. At this point, vertices in $G.V - L$ are in the same connected component. Adding edges from $e_{k+1}, ..., e_m$ will make every vertex in L a leaf. Kruskal's algorithm also ensures that edges selected from $e_{k+1}, ..., e_m$ has minimum total weight. Hence, the algorithm is correct.