CSC373 - Problem Set 2

Authors: Luke Bacchus, Naslin Rahman, Zhuoqian Li

February 2, 2021

Question 1

Question 2

a. Our goal is to find the easiest trip which is the trip with the minimum toughness. The toughness of a trip is the greatest degree of difficulty among all portages that trip. Originally, in Dijkstra's algorithmn, we have d(v) = the sum of the weights between nodes/lakes on the path to node v. However, the proposed modification is to assign d(v) as the max of the weights (aka "difficulties") of the portages between lakes.

   Below is the pseudocode:

```
function find(s, t, L, n):
    R = [s]
    d[s] = 0
    V = [1,2,...,n]

    for v = 1 to n:
        if v != s:
            if v in L[s]:
                v_weight = L[s][v][1]
                R.append(v)
                d[v] = v_weight
                p[v] = s
            else:
                R.append(inf)
                d[v] = inf
                p[v] = nil

    while R != V:
        not_R = V - R
        not_d = []

        for v in not_R:
            not_d[v] = d[v]

        u = not_d.get_index(min(not_d))
        R.append(u)

        for v = 1 to n:
            if v != u and v in not_R:
                if v in L[u]:
                    if max(d[u], L[u][v][1]) < d[v]: \\TODO:Check if this l
                        d(v) =  max(d[u], L[u][v][1])
                        p(v) = u

    return d(t)
```

TODO: Give, and briefly justify, the complexity of your modified algorithm The first for loop would have a runtime of c*n, where c is a constant. The while loop would take at most n iterations, while the first for loop within the while loop would take m runtime, where m ¡= n. The second for loop in the while loop would take at most n iterations. Thus the worst case runtime of the modified algorithm is $O(c+cn+m+n^2)$ and thus $O(n^2)$.

b. In Dijkstra's original algorithm, d(v) is a sum of the weights. It assumes that when a new a new node, u, is added to R, for all nodes, v, in V - R whose R-path contain u, $d(v) = min(d(u) + w_u v, d(v))$. This relies on non negative weights since it relies on the shortest path to v being either the original path s-¿ v or the path s-¿u + v. Negative weights would allow this assumption to be false since a negative weight connected to v elsewhere could lessen the weight of another path, not mentioned previously, making it possible to be the shortest path to v.

   This assumption is not necessary since our modification takes the max weight of all the portages on the path. Thus, negative weights would not impact the outcome.

Question 3