Homework 5

Date: 2023-07-18 14:15
type: #Homework
Category: 📌cs 311 MOC

MSTs & Dijkstra's

Question 1

Let G be a graph and T be its MST, suppose we increase the weight of an edge $e \in T$ by a postive quantity δ. Give an algorithm that gets G, T, e and δ as input and returns a MST of the new graph. Prove the correctness of your algorithm and state the run-time.

```
MST(G,T,e,d) {
        remove edge e = (u,v)

        //after removing e, we have 2 subtrees, T1 and T2
        //traverse them to know which vertices are in each tree (T1 or T2)
        T1vertices = BFS(T, u)
        T2vertices = BFS(T, v)

        //now we traverse G and if any minimum edge connects T1 to T2 that is the
new MST
        Traverse edges of G:
                keep track of minimum edge e = (u,v)
                if e in T1vertices & T2vertices
                        new minimum edge = e

}
```

The algorithm starts by removing the edge that is increased by δ, because we are assuming that the increase no longer makes it the MST of graph G. Removing and edge $e = (u, v)$ creates 2 subtree rooted at either u or v respectively, so we traverse them with BFS to discover which vertices belong to which tree. once we know which vertices are where, we can traverse each edge of the original graph G, and we know that if we see a minimum edge that connects any vertice from subtree T1 to subtree T2, then that is the new edge that would replace the initial removed edge, thus forming the new MST.

Proof:
Lets assume that the original MST is still the MST after the edge e was increased by δ.

We continue to remove the edge $e = (u, v)$ and discover vertices of each subtree u and v. Continue to traverse each edge in graph G to see if there is a connection between the two subtrees: if the edge does connect them, then it is the new minimum, if not, continue

checking edges. Repeat until all edges have been visited. We check if our minimum edge weight equals the original edge we removed:

- If yes : then our assumption holds
- If not : then our assumption does not hold, which indicates that the original edge needs to be removed, which contradicts our assumption meaning that our algorithm will produce the desired result.

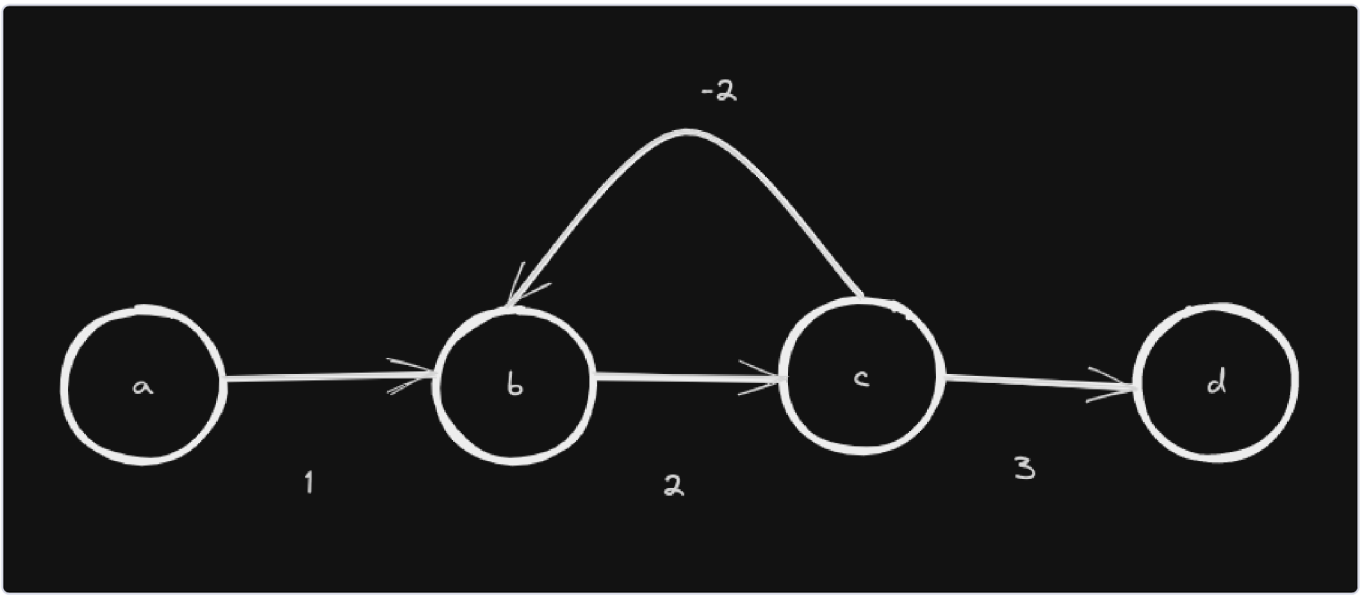
Runtime:

- We run BFS to discover all edges in each subtree, so 2 $O(E+V)$.
- Then we check each edge in the initial graph G, so thats $O(E)$.
- Added together we get $2O(V+2E)$ which can be simplified to just $O(V+E)$.

Question 2

Give an example of a directed edge weights on which Dijkstra's shortest path algorithm produces an incorrect answer. Draw the graph, identify the source vertex s and a vertex v for which the algorithm produces incorrect shortest path. Write the path obtained by Dijkstra's algorithm and the correct path.

A graph in which Dijkstra's algorithm produces an incorrect shortest path is when negative weights are introduced:



Here, Dijkstra's algorithm would perform the follow:

$$a \to b \to c \to b \to c \to d$$

Because it greedily checks whichever path is shortest to its current vertex, when going from $c \to d$, it sees 3, but going $c \to b$ is -2 which it thinks is shorter but actually goes backwards.

It would get $1+2-2+2+3$ = 6, and the actual shortest path is $a \to b \to c \to d$, $1+2+3$ which is also 6, so in theory it is correct, but in actuality it is not the shortest path.