**CSE 551 – Assignment 4**

Submitted By:

**Ankit Sharma**

1219472813

ashar263@asu.edu

**Solution for Question (1)**

* Does the minimum spanning tree change?

**No**, Minimum spanning tree (MST) will not change even if each edge weight in a graph is increased by 1.

*Proof*:

Assume graph G has ‘n’ vertices.

Then, all spanning trees of G has edges.

Since weight of each edge is increased by 1 as provided in question. Therefore,

weight of all spanning trees will be increased by

Since weight of all spanning trees is increased by same value, then **MST in previous graph G will also be MST in new graph**.

* Does the shortest paths change?

**Yes**, the shortest paths change even if each edge weight in a graph is increased by 1.

*For ex*: Please consider below graph –

A picture containing text, whiteboard

Description automatically generated

Shortest path from to is:

and cost,

New graph after increasing each edge weight by 1

A white board with blue writing

Description automatically generated with low confidence

Shortest path now from to will be:

and cost,

As has

Hence proved, that **shortest paths between nodes can change**.

**Solution for Question (2)**

This algorithm will not always return minimum node cover set.

*Proof*: Let’s take following graph as example –

A picture containing text, whiteboard

Description automatically generated

According to the given greedy COVER algorithm, every time we select a vertex of maximum degree.

Suppose is selected first amongst and. Then the resultant graph after removing the vertex from set V and its corresponding edges from set E will look like –

Diagram

Description automatically generated

In next two iterations, following greedy approach, if and then are selected. The graph would look like –

A picture containing text

Description automatically generated

Finally,

Then the size of node cover set would be

But the minimum node cover for this graph is and , as shown in below:

Diagram

Description automatically generated

Hence proved that the greedy algorithm **will not always return minimum node cover set**.

**Solution for Question (3)**

We can use Ford-Fulkerson algorithm to compute the maximum flow from Vancouver to Winnipeg.

First, we initialize each edge with a flow , where means all edges in the graph.

A picture containing text, watch

Description automatically generated

Let, **first augmented path**

A picture containing text, watch

Description automatically generated

Now,

Since all are forward labels in the augmented path. The augmented flow will be:

A picture containing text, watch

Description automatically generated

Let, **second augmented path**

A picture containing text, watch

Description automatically generated

Now,

Since all are forward labels in the augmented path. The augmented flow will be:

A picture containing text, watch

Description automatically generated

Let, **third augmented path**

A picture containing text, watch

Description automatically generated

Now,

Since all are forward labels in the augmented path. The augmented flow will be:

A picture containing text, watch

Description automatically generated

Now, there are no possible augmented paths. So, the maximum flow from Vancouver to Winnipeg is:

We can also see that this value of

A picture containing text, watch

Description automatically generated

, where and



**Solution for Question (4)**

Let’s consider a network, given below:



Now, there are two possible cuts:

*Case 1*:

and

*Case 2*:

and



In the network above, Max Flow from to

We need to check if if is defined as-

**(a)**

Using this formula-

For case 1, capacity of the cut is

For case 2, capacity of the cut is

Now,

Hence, the **above definition of C(S:S’) is** **FALSE** and cannot be used to calculate Maximum flow.

**(b)**

Using this formula-

For case 1, capacity of the cut is

For case 2, capacity of the cut is

Now,

Hence, the above definition of C(S:S’) is **FALSE** and cannot be used to calculate Maximum flow.

**(c)**

We know that Total Flow from (S:S’) is-

As discussed in class,

Total Flow,

=>

We also know that , So-

=>

=>

Now , So-

=>

From the above relation we can see that value of cannot be higher than minimum value of . Therefore-

=>

=>

=>

Hence, the **above definition of C(S:S’) is** **TRUE** and can be used to calculate Maximum flow. We can also check validity of definition in the above network.

For case 1, capacity of the cut is

For case 2, capacity of the cut is

Capacity of =

**(d)**

Using this formula-

For case 1, capacity of the cut is

For case 2, capacity of the cut is

Now,

Hence, the above **definition of C(S:S’) is** **FALSE** and cannot be used to calculate Maximum flow.