Eric Yang

304263623

CS 180

Dis 1B

Homework 7

1. a. Suppose there is a source node connected to several other nodes by 1 edge each. The node with the lowest edge weight must have shortest path with 1 edge from source node since no other path can possibly cycle back to this node to create a shortest path otherwise there would be a negative cycle which is a contradiction.

b. Suppose the algorithm is run on a directed graph with no negative cycle. The longest possible path without a cycle can have V-1 edges so the edges will be scanned V-1 times to find the shortest path for each node. Suppose once we perform the nth round and a distance is updated, then a path of length V edges is found which can only happen when there is a negative cycle which is a contradiction.

1. We can choose a vertex and label that s. Then try every other vertex as t and run the Ford Fulkerson algorithm for each t. Record each min cut result in a table. Output the lowest cost solution in table.
2. Suppose we have a graph with a vertex v. Then we create a new graph and change v into two vertices v1 and v2 which are connected by an edge and let all incoming edges connect to v1 and outgoing edges connect to v2. We also assume all other edge weights are infinite. This new graph must have a minimum s-t cut that contains only new edges which means this min cut can be reflected as a set of vertices in the original graph. The total capacity of the vertices equals the min edge capacity in the new graph. The maximum flow of the original graph = the maximum flow of the new graph. Thus, the maximum s-t flow = min total capacity of vertices whose removal separates s and t.
3. Create an undirected graph with each application as a vertex. Each pair of vertices has an edge between them if there exists an x value associated with them and the capacity of the edge is equal to the x value. Next we create a new vertex called t and create edges from each application vertex (except 1) to t with associated capacity b. To find the set of applications where the benefit minus expense is maxed, we simply find the min cut with source node = application 1 vertex, and sink node = t