Evan Zhang | z5383657

Question 3

3.1

For an edge to be not useful, it will not be part of any shortest path from any vertex u to any vertex v. Therefore, we will need to find all shortest paths (including equal weighted ones) and set all edges that form a shortest path as being part of the useful set. Any edges excluded will be not useful. Use an array with indexes corresponding to vertex number as representing whether a number is in the 'useful' set or not, with either a 1 or a 0, for being in or not, respectively.

Perform the Floyd-Warshall algorithm on the graph, taking $O(n^3)$ time total as in the lectures. Set the edges in the final shortest path as useful for every shortest path found.

Recurrence:

Recurrence: for all $1 \leq i, j, k \leq n$,

$$\text{opt}(i,j,k) = \min(\text{opt}(i,j,k-1), \text{opt}(i,k,k-1) + \text{opt}(k,j,k-1)).$$

If there are ties, that is if opt(i , j , k −1) is equal to opt(i , k, k −1)+opt(k, j , k −1), set the predecessors as both paths.

Base cases:

Base cases:

$$\text{opt}(i,j,0) = \begin{cases} 0 & \text{if } i = j \\ w(i,j) & \text{if } (v_i, v_j) \in E \\ \infty & \text{otherwise.} \end{cases}.$$

If there are multiple w(i,j) that are equal, add them all to a list representing predecessors for opt(i,j,0).

Time complexity:

The Floyd-Warshall algorithm takes $O(n^3)$ time, as detailed in the lectures. The additional representations of predecessors will take O(1) time for each additional edge, not increasing time complexity bound. same-length shortest paths will take O(1) time for each

3.2

For an edge to be very useful, it will need to be a part of every shortest path from some vertex u to some vertex v. Therefore, if a path from u to v only has 1 shortest path, all edges along that path will be useful. Else if there are multiple shortest paths, all common edges along those paths will be very useful.

Repeat the algorithm in 3.1 except for the following modifications:

- If there are no ties (i.e. there is only one shortest path), add all edges on that path to the 'very useful' set.
- If there are ties, add all edges to the 'useful' set if it is the first shortest path to be looked at. From then on, when setting edges as 'useful', check if they are already in the 'useful' set for that path. If they are not, do not add them. If they are, increase their value in the 'useful' array by 1. Once all edges have been processed for each shortest path, add each edge that has a value equal to the amount of shortest paths in the 'useful' set to the 'very useful' set.

Once all shortest paths existing in the graph have been viewed, the remaining edges in the 'very useful' set are the 'very useful' edges. This is correct as all common edges in all shortest paths between all vertices has been added to the 'very useful' set, without duplicates.

Time complexity:

The algorithm is completed in $O(n^3)$ time as the Floyd-Warshall algorithm takes $O(n^3)$ time and for each shortest path