Assignment 8 Problem 2: Approximating the Minimum Weight Connected Subgraph

Construct an complete graph namely H on the vertex set S. While constructing, for each edge uv, we search it in E(G). If it is in E(G), then the weight of such edge should be the exactly the same as that in G. If not, contains some information on that edge. The information should be the shortest uv-path in G and the edges it used. And set the weight of uv-path in G to be the weight of that 'edge'.

Then we apply Kruskal's algorithm to find the minimum spanning tree of the complete graph H. During the algorithm, we need to track the edges that is already used. For those edges originally not in E(G), we need to add the path (stated previously) that is recorded in the 'edge' to the final answer. Also note that when adding edge, we need to check whether the edge is already in the answer, if yes then just ignore the adding step of such edge.

Once we find the minimum spanning tree of H, we also constructed an connected subgraph which satisfies the requirements.

The time used in this algorithm: to construct the complete graph, we need $O(n^2 m \log n)$ to search all shortest path and record them. For Kruskal's algorithm we need $O(m \log n)$ to find the minimum spanning tree. To record the answer, for each adding edge, we need $O(m)$ time to check. The total adding edge operation is in $O\big((n-1)n^2\big) \in O(n^3)$. Therefore, the total running time is polynomial.

For the approximation part, the reason why the result get by the algorithm is not optimal is because the optimal solution may use some edges that can also be used by other paths rather than the shortest path. If this happens, then the length of the adding path is strictly less than the path that in the optimal solution. Which means that the additional parts can never exceed the weight of the optimal answer. Therefore, the algorithm is 2-approximation.