Isaac Abboudi

Find Minyan Write-Up

February 1 2023

My solution models the problem as a graph, as graphs are the building blocks of computer science. Nodes are created implicitly when nCities is defined, and edges are saved as directedEdge objects. When calling addHighway, 2 directedEdge objects are instantiated, one in each direction, and added to the digraph's adj[].

I used Dijkstra's SPT algorithm as a base for solving the problem. The algorithm does not take other necessary 'middle-man nodes' into account. I figured I could split the journey into 2 parts: first, from the source to any city with a minyan, and second, from each minyan city to the goal.

Assuming there is a path from S to G that goes through a middle city C, that we want to find, it can be no greater than the sum of the shortest path from S to C, and the shortest path from C to G. Therefore,using Dijkstra initially to find the shortest path from city 1 to all cities marked by hasMinyan, we save each value as the first half of any journey to any minyan. From there, we iterate over all the minyanim calling Dijkstra's algorithm and find the distance to Goal from each minyan. The path with the lowest cost is found easily and from there we can compare it to all other values to find how many paths are of the same minimum value.

The factors we need to consider are V - number of nodes, E - number of edges, and the source node + number of minyan nodes we can call M. The time complexity of Dijkstra is O(M * (V + E*log(V))) but since we have to call it for all M's so we have to add in M^2. I believe the order of growth comes out to O(M^2 + M * (V + E*log(V))).