Isaac Abboudi

WeAreAllConnected write-up

March 23 2023

The algorithm models the problem as a weighted graph where the nodes represent the cities and the edges represent the distances between them. The graph is represented using an adjacency matrix or list, and the distances between any two nodes are recorded in a distance matrix. The algorithm uses the Floyd-Warshall algorithm to compute the shortest distance between all pairs of nodes in the graph. This algorithm has a time complexity of O(N^3), where N is the number of nodes in the graph.

A brute force algorithm for solving the problem would be to iterate over all possible new edges that can be added to the graph and compute the shortest path between all pairs of nodes for each possible new edge. Using Dijkstra's algorithm, the time complexity would be O(V^4)

I believe the better algorithm is to use the Floyd-Warshall algorithm to compute the shortest distance between all pairs of nodes in the original graph, and then iterate through all possible new edges and compute the shortest path between all pairs of nodes using the distance matrix. The new edge that results in the smallest increase in the total distance is the chosen edge. This algorithm has a time complexity of O(N^3), which is the same as the Floyd-Warshall algorithm. However, it is more efficient than the brute force algorithm because it does not require computing the shortest path for all pairs of nodes for every possible new edge. Instead, it uses the precomputed distance matrix to quickly compute the shortest path for each new edge. Also, if the new edge would replace an existing edge and the new duration is greater than or equal to the original duration, then that edge does not help us and is not considered.