Maze Simulation

Prateek Chaudhury 2019CS10384
Gunjan Kumar 2019CS10353

July 2021

1 Problem Description

There is a policeman who has to patrol the area after every hour in the night. Every time, the policeman starts patrolling from the police station. He visits all the localities and then after visiting all places, he returns back. All the places are connected by one way streets with each other. There are many possible such paths for policeman to visit all the localities. Out of all possible paths, he wants to chose the path which requires minimum distance to be covered. As the policeman cannot figure this out by himself, we have to help him find out the required path.

2 Simulation Using Maze

We can change the above problem to a graph problem which can further be mapped to travelling in a maze. Police Station and all the localities are vertices of graph where the edges between them represent the length of one-way routes between the localities. There can be many walks in the graph which starts from Police Station, then visits all other nodes and ends at the Police Station. Out of all such walks, we have to find the walk which is of the minimum length.

Further, the problem can be simulated using a maze. Each cell of maze which is not block represents the non-residential areas where police man need not go. Every other cell represents the nodes which has to be visited. Let's place the police man at the top left corner. Each cell will contain a certain number which represents the distance of the cell to the neighbouring cell using the one-way streets. We have to find a walk that starts from the top left corner, then visit all the cells of the maze and them, return back to the top-right corner by covering the minimum possible distance.

3 Approach 1 : Brute Force

Step 1 : In this approach, we will find all the possible permutations of nodes. Now, each permutation will represent one way of visiting all the nodes of the graph. Cost for each permutation will be equal to the summation of shortest paths between two consecutive vertices in the permutation.

Step 2 : We will use Floyd - Warshall algorithm to pre-process the graph and store the shortest distance between any two pair of nodes.

Step 2 : The shortest path visiting all the vertices of the graph will be minimum of all the possible permutations.

Feasibility :

The time complexity of the Floyd-Warshall algorithm = $O(N^3)$
The number of all different permutations of N nodes = $O(N!)$
Hence, total complexity will be $O(V^3 + V \times V!)$ which is not feasible as the factorial grows very fast.

4 Optimised Approach : Dijkstra Algorithm

5 Polynomial Time Complexity Not Achievable

6 Approximation Algorithm

Since, polynomial time complexity of the problem has not been discovered and most likely don't exist. We will try to find a path which has sum near to the minimum path possible and it is feasible even for higher number of nodes.

6.1 Algorithm

Step 1 : Starting from the Police Station as the root, find the Minimum Spanning Tree of the Graph using Prim's algorithm.

Step 2 : Find the list of vertices, ordered according to when they are first visited in a preorder tree traversal of Minimum spanning Tree.

Step 3 : The approximated path is the hamiltonian cycle of the list of the vertices found in the above step.

6.2 Correctness of Algorithm

The path returned by the above algorithm clearly visits all the vertices of the graph. Now, we need to show that the length of the path return by the above algorithm is close to the length of the actual such path.

Claim : The length of the path produced by the above algorithm is never more than twice the length of the path of best possible output.

Proof : Let the length of any path (P) is L(P). Let's name the minimum spanning tree as T and The path returned by above algorithm as H. Assume that the best possible path is B.

Now, the length of the path B is never less than the length of the MST, T. This is because MST is the minimum cost tree that connects all vertices. The path B visits all nodes so, its length has to be greater than the length of B.

Now, the length of the path H is never less than the length of the MST, T.

Hence, length of the path H is less than twice the length of the path B.

6.3 Run Time Analysis

Complexity of Algorithm = **Prim's Algorithm** Complexity + **Tree Traversal**

Even with the implementation of MST-PRIM with using priority queue, the running time complexity of the above is proposed Algorithm = $O(N^2)+O(N) =$ $\mathbf{O}(N^2)$

If we implement the Prim's algorithm using binary min heap **priority queue**, then the time complexity of the algorithm will drop to = $O(NlogN) + O(N) =$ **O(NlogN)**

Feasibility : This approach is feasible and will run for even very large number of nodes upto millions. The $O(N^2)$ implementation of the approach can run within seconds for 10^4 nodes while the $O(NlogN)$ implementation of the approach can run within seconds for 10^7 nodes.

6.4 Implementation Plan