Data structures 2

Shortest paths Algorithms

Mohamed Elnady Mohamed Gomaa	20011513
Mohamed Aly Hassan Mahmoud	20011662
Abd ElRahman Ramadan Mohamed Moussa	20010809
Khaled Mohamed Mohamed Ahmed Soliman	20010528

Overview

In this Assignment we implemented 3 shortest paths algorithms which are

1- **Dijkstra Algorithm**

2- **Bellman-Ford Algorithm**

3- **Floyd-Warshall Algorithm**

Implementation details

Graph is implemented as Adjacency matrix (2D array), where a[i][i] is the value of weight of edge that connects between node i and node j.

So the space complexity of the graph is worst case including adding or removing an edge O(V^2).

Adding edge or find and existing edge has complexity time O(1) to access a[i][j].

Time and space complexity for implemented algorithms

1. **Dijkstra Algorithm**

   It finds shortest paths from the source to all other nodes in the graph.

   as it is implemented in adjacency matrix it has time complexity O(V^2), where V is the number of vertices in graph and this because for each vertex V it is needed to relax all its connected edges and this takes time O(V) , However the time complexity can be reduced using an adjacency list or priority queue .

   Space Complexity is O(V^2) as the graph is implemented as 2D matrix, However two extra arrays are used to keep track of visited vertices and distance of each vertex but they have size of V , so space complexity is still O(V^2).

2. **Bellman-Ford Algorithm**

   It finds shortest paths from a single source vertex to all of the other vertices in a weighted directed graph, Unlike Dijkstra it can deal with negative edge weights and detects negative cycles existence.

   The time complexity in an average case we relax E edges for V-1 times so it is O(VE), However in the worst case the graph is complete and has E(E-1)/2 edges and relax them all (V-1) time , So it is O(V^3).

Space Complexity is O(V^2) as the graph is implemented as 2D matrix, However two extra array are used to keep track of distance of each vertex but they have size of V , so space complexity is still O(V^2).

3. **Floyd Warshall Algorithm**

   It is used to find the shortest path between all vertices in the weighted graph. This algorithm works with both directed and undirected graphs but it does not work along with the graph with negative cycles.

   The time complexity is O(V^3), where V is the number of vertices in a graph.

   The space complexity is O(V^2), where it returns a 2D array containing the shortest paths of all pair nodes from i to j.

Notes:

1. To find all pairs node shortest path using dijkstra we will need to find dijkstra for all node (V) time which will increase time complexity to O(V^3) and space complexity to O(V^2) , but it is still can't compute graphs with negative weights.
2. To find all pairs node shortest path using Bellman Ford we will need to find bellman for all node (V) time which will increase time complexity to O(V^4), and it can detect negative cycles and compute graphs with negative weights.

Conclusions:

- Dijkstra is the best to find the shortest path from a single node to all other nodes that the graph has all edges weighted by positive weights.
- Bellman is the best to find the shortest path from a single node to all other nodes no matter whether edge weights are positive or negative.
- Floyd warshall is the best to find all pair nodes shortest path.

Time comparisons

1. the mean time to get the shortest path between 2 specific nodes

Graph	Dijkstra (nano. s)	Bellman (nano. s)	Floyd (nano. s)
Graph 1 (6 nodes)	8600	11600	24300
Graph 2 (4 nodes with +ve cycle)	9900	7200	13800
Graph 3 (7 nodes) -ve weight	N/A	16000	31700
Graph 4 (5 nodes)	5500	6500	18600
Graph 5 (8 nodes)	12000	15600	33100
Graph 6 (7 node) -ve weight	N/A	13400	27400
Mean	9000	11716.667	24816.667

2. the time to get the shortest paths between all pairs of nodes

Graph	Dijkstra (nano. s)	Bellman (nano. s)	Floyd (nano. s)
Graph 1 (6 nodes)	266000	212700	74600
Graph 2 (4 nodes with +ve cycle)	208300	155100	79100
Graph 3 (5 nodes)	200800	186700	84400
Graph 4 (8 nodes)	313100	342400	89500
Graph 5 (6 nodes) -ve weight	N/A	254700	88600
Graph 6 (7 node) -ve weight	N/A	380900	109000
Mean	247050	255416.67	87533.33