assignment 3

Q1 Find shortest distance graph & shortest path between all pairs of nodes in a weighted directed graph using floyd warshall algorithm.

Algorithm

Algorithm - set path(i) (source node) and label (i, -)

$D_0 = [w_{ij}]$ $n \times n$

$D_k[i,j] = \min(D_{k-1}[i,j], D_{k-1}[i,k] + D_{k-1}[k,j])$

where D_k

Analysis of runtime

In the above algorithm the first for loop is used to create the adjacency matrix for the graph in $\Theta(n^2)$ time.

There are three nested for loops which are now to run from 1 to n resulting of this algorithm is $O(n^3)$

The information about the path can also be maintained by creating one more matrix

Example:

Initially adjacency matrix is as given by

$$\begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix}$$

Now, there are four matrix in the graph we need to create three more matrix D_1, D_2, D_3 where D_1 is a matrix which compute the distance between all pairs of nodes with 1 as the intermediate vertex.

$$D_0 = \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix} \quad D_1[i,j] = \min(D_0[i,j], D_0[i,1] + D_0[1,j])$$

$D_1[1,1] = \min(D_0[1,1], D_0[1,1] + D_0[1,1])$

$= \min(0, +\infty)$

$= 0$

$D_1[1,2] = \min(D_0[1,2], D_0[1,1] + D_0[1,2])$

$= \min(3, +\infty)$

$= 3$

$D_1[1,3] = \min(D_0[1,3], D_0[1,1] + D_0[1,3])$

$= \min(\infty, +\infty)$

$= \infty$

$D_1[1,4] = \min(D_0[1,4], D_0[1,1] + D_0[1,4])$

$= \min(7, +\infty)$

$= 7$

$D_1[2,1] = \min(D_0[2,1], D_0[2,1] + D_0[1,1])$

$= \min(8, +\infty)$

$= 8$

$D_1[2,2] = \min(D_0[2,2], D_0[2,1] + D_0[1,2])$

$= \min(0, +\infty)$

$= 0$

$D_1[2,3] = \min(D_0[2,3], D_0[2,1] + D_0[1,3])$

$= \min(2, +\infty)$

$= 2$

$D_1[2,4] = \min(D_0[2,4], D_0[2,1] + D_0[1,4])$

$= \min(\infty, +\infty)$

$= \infty$

Similarly D_2 and D_3 can be computed.

$$D_3 = \begin{bmatrix} 0 & 3 & 3 & 7 \\ 8 & 0 & 2 & 10 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & 5 & 0 \end{bmatrix}$$

The final matrix which is by the last iteration the shortest matrix which contains the all pairs shortest path values.

Q2 Bellman Ford algorithm

Given graph

1st Step

2nd Step

3rd Step

final graph

Q3 Dijkstra's Algorithm

Step 1 - Start from 0

Step 2 - Add C

Step 3 - Add D

Step 4 - Add B

Step 5 - Add A

Q4 MST

A MST is a connected undirected, weighted graph is a subset of the edges that connects all vertices together that has the minimum possible total edge weight among all possible spanning trees.

a) Kruskal Algorithm - Greedy Approach

Time Complexity MST has edges this version of Kruskal's algorithm runtime is $O(E \log E)$ where E is number edges in G.

Steps: a) Find all vertices and edges

b) Add all edge starting from smallest edge, consider this growing MST as a root node.

c) Repeat until all vertices are included.

Sub Problem Sort edges lexicographically by ascending (priority queue) and use adjacency list.

Time Complexity $O(E \log V)$

b) Prim's Algorithm - Greedy Approach

Step: Find all edges Greedy approach and choose the minimum weight edge from the set which connect vertices.

Step 2: Find all edges to non-connecting vertices of existing MST.

Step 3: Repeat until MST has $(V-1)$ edges.

Sub Problem Use Binary Heap (Priority Queue) for distance values.

Time Complexity $O(E \log V)$

Q5 DYNAMIC

Analysis	Fibonacci Algorithm	Kruskal's Algorithm
Complexity	Fibonacci MST Runtime is $O(E + V \log V)$ where E is number of edges and V is number of vertices.	Kruskal's algorithm runtime is $O(E \log E)$
Sub problem	Priority Queue	Binary Heap
Time Complexity	$O(E + V \log V)$	$O(E \log E)$

Kruskal's Implementation Binary Heap gives better performance than Fibonacci Heap.

Sub Problem Complexity: $O(E \log E)$ Priority queue, adjacency list, using Binary Heap

Q6

T = STARTCATCATCAT