PICT, PUNE

Assignment 8

Name : Bhavana Bafna
Class : SE 09
Batch : E 09
Roll no. : 23107

Title : Shortest Path Finding

Aim : To implement shortest path using Dijikstras algorithm.

Problem statement : Represent a graph of city using adjacency matrix / adjacency list. Nodes should represent the distance between them. Find the shortest path using Dijikstras' algo from single source to all destination.

Theory :

What is shortest path?
In graph theory the shortest path is the path between 2 vertices as such that the sum of the weight of its edge is minimized. The problem of finding the shortest path in a graph from one vertex to another. Shortest can be least number of edges least total weight, etc.

Various algorithm to find shortest path :
1) Dijikstras algorithm.
2) Bellman-ford algorithm.
3) Floyd-Warshall algorithm
4) Johnson algorithm.
5) Viterli algorithm.

PICT, PUNE

- Greedy approach -

An algorithm is designed to achieve optimum solution for a given problem. In greedy algorithm approach, decisions are made from given solution domain. As being greedy the closet solution that seems to provide an optimizes solution is chosen.

Greedy algorithm builds up solution piece by ~~path~~ piece always choosing the next piece that offers the most obvious and immediate benefits.

- Dijikstras algorithm -

It is an algorithm for finding the shortest path between nodes in graph. The algorithm creates a ~~tree~~ tree of shortest path from starting vertex (source) to all other points in graph Dijikstras algorithm finds the shortest path till from a single source node by building a set of nodes that have minimum distance from source.

Real time uses of Dijikstras Algorithm:

1) Social Networking appliance.
2) Telephone Network.
3) Digital Mapping service in Google meyp.
4) IP routing to find open shortest path first.
5) Fighting agenda.

⊛ Algorithm for Dijikstras single source to multiple destination:

PICT, PUNE

Procedure Dijikstras:
//src is source vertex
for i = 0 to v
// find initial distance
if weight [src][i] != 0
dis [i] = weight [src][i]
else
dis [i] = 32767
path [i] = src
visited [i] = 0
End for
// take source as current vertex and make it visited
current = src
visited [src] = 1
// repeat for all vertices
for j = 0 to v - 2
min dist = 32767
// find min. dist from current to all other
for i = 0 to v
if visited [i] != 0 & dist [i] < min. dist.
mindist = dis [i]
current = i
End for
// make current as visited
visited [current] = 1
// find shortest path from current.
for i = 0 to v
if visited [i] = 0 and (dis [current] + weight [current][i] (dis [i])).
dis [i] = dis [current] + weight [current][i]

```
path [i] = current
End for
End for

// display shortest path.
    for i = 0 to v
        if i ≠ src
            print i, dis[i]
            int j = i
            do
                j = path [j]
                print path [j]
                while j ≠ src
            End for
```

END.

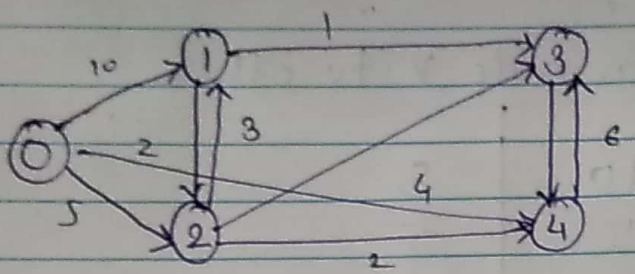
Validation:

1) No. of vertices & edges are positive
2) Weight of an edge is not negative.
3) Remove parallel edge at higher cost.

Test cases:

PICT, PUNE

1) For directed Graph:



Node	From node V_0 to other
V_1	10
V_2	5
V_3	∞
V_4	∞
best	

2) Find shortest path to neck 0:

Node 2 selected.

Node	From node V_0 to other
V_1	10
V_2	5
V_3	∞
V_4	∞
best	V_2

PICT, PUNE

3) Recalculate path to other node (Node V_4):

Node	From node V to other	
V_1	10	8
V_2	5	5
V_3	∞	14
V_4	∞	7
best	V_2	V_4

4) Find shortest path to Node 0, Node V selected:

Node	From node V_0 to other		
V_1	10	8	8
V_2	5	5	5
V_3	∞	14	13
V_4	∞	7	7
best	V_2	V_4	V_1

5) Find shortest path to node 0, node 3 selected:

Node	From node V_0 to other			
V_1	10	8	8	8
V_2	5	5	5	5
V_3	∞	14	13	9
V_4	∞	7	7	7
best	V_2	V_4	V_1	V_3

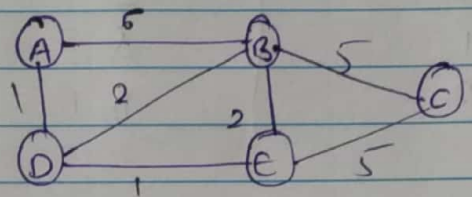
PICT, PUNE

∴ Shortest path can be calculated as

$V_0 \rightarrow V_2 \rightarrow V_4 \rightarrow V_1 \rightarrow V_3$

Cost = 11

2) For Undirected graph:



From current vertex = A, examine unvisited neighbours:

Vertex	shortest dist	Path
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Current vertex = D Examine unvisited neighbour at D:

Vertex	Shortest dist	Path
A	6	
B	3	D
C	∞	
D	1	A
E	2	D

PICT, PUNE

Current Vertex = E ∴

Vertex	Shortest dist	Path
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Current vertex = B

Vertex	Shortest dist	Path
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Path :

A → D → E → B → C

Cost = 9

PICT, PUNE

Conclusion:

We successfully implemented graph and shortest path using Dijikstras algorithm.

Time complexity = $O(E \log V)$

Space complexity = $O(V)$