Ex. No. 9c Dijkstra’s Shortest Path

Date:

**Aim :**

To find the shortest path for the given graph from a specified source to all other vertices using Dijkstra’s algorithm.

**Algorithm**

1. Start

2. Obtain no. of vertices and adjacency matrix for the given graph

3. Create cost matrix from adjacency matrix. C[i][j] is the cost of going from vertex i to vertex j. If there is no edge between vertices i and j then C[i][j] is infinity

4. Initialize visited[] to zero

5. Read source vertex and mark it as visited

6. Create the distance matrix, by storing the cost of vertices from vertex no. 0 to n-1 from the source vertex

distance[i]=cost[0][i];

7. Choose a vertex w, such that distance[w] is minimum and visited[w] is 0. Mark visited[w] as 1.

8. Recalculate the shortest distance of remaining vertices from the source.

9. Only, the vertices not marked as 1 in array visited[ ] should be considered for recalculation of distance. i.e. for each vertex v

if(visited[v]==0) distance[v]=min(distance[v]

distance[w]+cost[w][v])

10. Stop

**Program:**

/\* Dijkstra’s Shortest Path \*/

#include <stdio.h>

#include <stdlib.h>

#define INFINITY 9999

#define MAX 10

void dijkstra(int G[MAX][MAX], int n, int startnode);

void main()

{

int G[MAX][MAX], i, j, n, u;

printf("Enter no. of vertices: ");

scanf("%d", &n);

printf("Enter the adjacency matrix:\n");

for(i=0; i<n; i++)

for(j=0; j<n; j++)

scanf("%d", &G[i][j]);

printf("Enter the starting node: ");

scanf("%d", &u);

dijkstra(G, n, u);

}

void dijkstra(int G[MAX][MAX], int n,int startnode)

{

int cost[MAX][MAX], distance[MAX], pred[MAX];

int visited[MAX],count, mindistance, nextnode, i, j;

for(i=0; i<n; i++)

for(j=0; j<n; j++)

if(G[i][j] == 0)

cost[i][j] = INFINITY;

else

cost[i][j] = G[i][j];

for(i=0; i<n; i++)

{

distance[i] = cost[startnode][i];

pred[i] = startnode;

visited[i] = 0;

}

distance[startnode] = 0;

visited[startnode] = 1;

count = 1;

while(count < n-1)

{

mindistance = INFINITY;

for(i=0; i<n; i++)

if(distance[i] < mindistance && !visited[i])

{

mindistance = distance[i];

nextnode=i;

}

visited[nextnode] = 1;

for(i=0; i<n; i++)

if(!visited[i])

if(mindistance + cost[nextnode][i] < distance[i])

{

distance[i] = mindistance +

cost[nextnode][i];

pred[i] = nextnode;

}

count++;

}

for(i=0; i<n; i++)

if(i != startnode)

{

printf("\nDistance to node%d = %d", i,distance[i]);

printf("\nPath = %d", i);

j = i;

do

{

j = pred[j];

printf("<-%d", j);

} while(j != startnode);

}

}

**Result:**

Thus Dijkstra's algorithm is used to find shortest path from a given vertex.