VOL- VII ISSUE- II FEBRUARY 2020 PEER REVIEW e-JOURNAL IMPACT FACTOR 6.293 ISSN 2349-638x

Flow Chart Method To Find Optimal Cost To Deliver Message To Desired Location

Ms Arati Uttamrao Patil
B.E Computer Science and Engineering, Student.
TKIET,Warananagar.

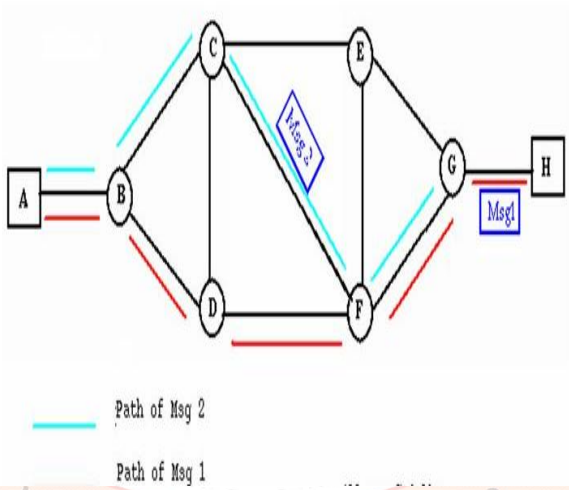
Abstract:

If any one wants to send a message to his friend that is from one location to another location. So he has hired a messenger. The cost of messenger service depends on the distance it travels to deliver message to desired location.So he wants to optimize the cost. This method provide the facility to the messenger to deliver message to desired location within minimum cost. And the cost rate is depend upon the total distance travelled to send the message to desired location.

***Keywords**: Optimal Cost To Deliver Message, Flow Chart Method*

a. Introduction:

A Messenger is a person or a thing that sends a message from one location to another location.A person wants to send a message to his friend that is from one location to another location. So he has hired a messenger. The cost of messenger service depends on the distance it travels to deliver message to desired location.So he wants to optimize the cost.



b. Problem Statement:

A person wants to find optimal cost to deliver message to desired location.

c. Problem Description:

The person wants the messenger to deliver message to desired location within minimum cost. And the cost rate is depend upon the total distance travelled to send the message to desired location. Hence first we have to find shortest path between two given nodes. The shortest path between the source node and destination node is calculated using "Dijkstra's shortest path algorithm". We will take weight matrix as an input from the user which consists the distances from one node to all its neighbours. The user have to again input two nodes i.e. source & destination. And after getting the shortest path we will calculate optimal cost based on the cost rate of messenger.

Input:

We will take distance matrix as a input. In this matrix we are going to take distance of one node to all its neighbours according to user's need. Then the two nodes are taken as an input from the user as a source and destination.

i.e

1)Weight matrix

2) Source node and Destination node

Output: Display the minimal cost needed for delivery of message to desired destination. If the message can't be delivered display "Impossible" instead.

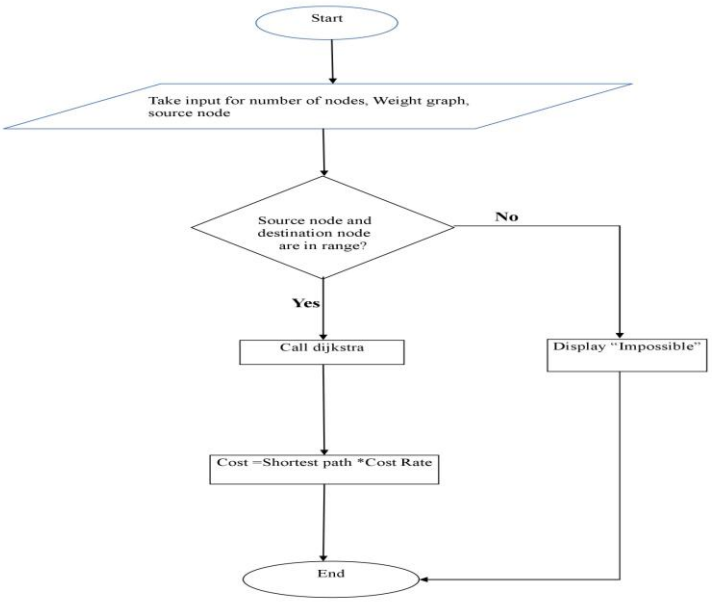
i.e

1) Shortest path
2) Optimal cost

Sample input	Sample output
Enter the no of nodes you want : 4 Enter first node in which you want connection : 1 Enter second node in which you want connection : 2 Enter the distance bet them : 1 . . . Enter the source node : 1 Enter the destination node : 2	Shortest distance:1 Shortest path is : 2->1 Total cost is Rs.5

2. Proposed Design:

a. Flowchart:



b. Algorithm: Algorithm for project.

Step 1. Start the execution of the program.

Step 2. Take input for number of places, adjacency weighed matrix.

Step 3. Display the weighed matrix.

Step 4. Take input for source node and destination node.

Step 5. Check whether the source node and destination node are in range. If yes go to step 6, otherwise step 8

Step 6. Call dijkstra function

Step 7. Calculate cost

Step 8. Display "impossible"

Step 9. End of execution.

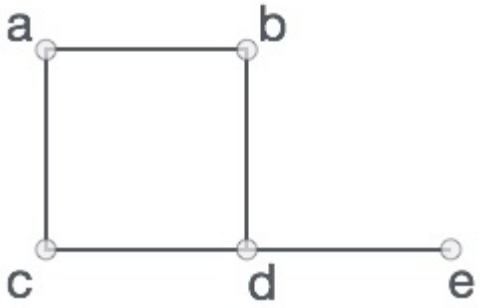
3. SOLUTION TECHNIQUES:

a. Data Structure Used:

1. Graph:

A graph is a pictorial representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented

by points termed as vertices, and the links that connect the vertices are called edges.
Formally, a graph is a pair of sets (V, E), where V is the set of vertices and E is the set of edges, connecting the pairs of vertices. Take a look at the following graph −
Graph Basics

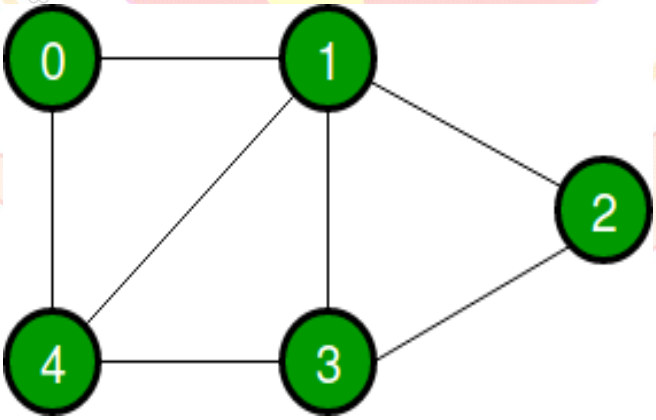


In the above graph,
V = {a, b, c, d, e}
E = {ab, ac, bd, cd, de}

Graph is a data structure that consists of following two components:
a) A finite set of vertices also called as nodes.
b) A finite set of ordered pair of the form (u,v) called as edge.

The pair of form (u,v) indicates that there is an edge from vertex u to vertex v. The edges may contain weight or value or cost. Graph can be represented in Adjacency matrix and Adjacency list.

Graph:



2. Adjacency Matrix:

Adjacency matrix is a 2D array of sized V*V where V is the number of vertices in graph. Let 2Darray be adj[][],a slot adj[i][j]=1 indicates that there is an edge from vertex I to vertex j. Adjacency matrix for undirected graph is always symmetric. If adj[i][j]= w, then there is an edge from vertex I to vertex j with weight w.

Representation of graph by Adjacency Matrix:

	0	1	2	3	4
0	0	1	0	0	1
1	1	0	1	1	1
2	0	1	0	1	0
3	0	1	1	0	1
4	1	1	0	1	0

b. Algorithm Used:

The shortest path is computed using "Dijkstra Algorithm".

Step-1: Each node is labelled with its distance from the source node along the best known path.

Step-2: Initially no paths are known so all nodes are labelled with infinity.

Step-3: As the algorithm proceeds and paths are found, the labels may change reflecting better paths.

Step-4: A label may be either tentative or permanent.

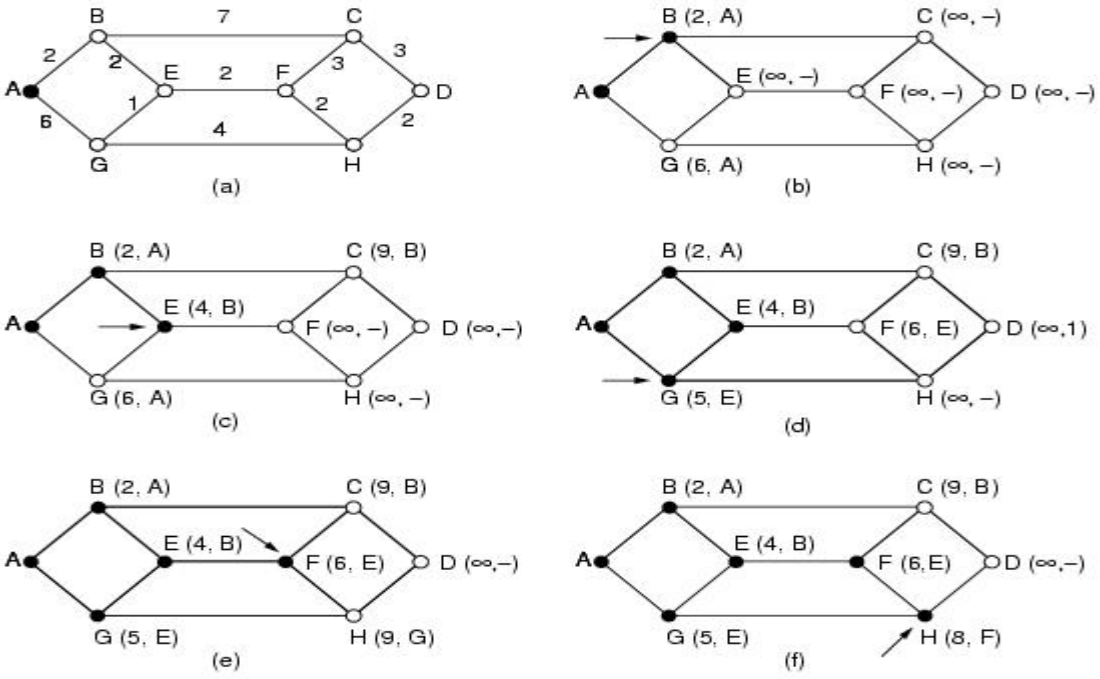
Step-5: Initially all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never change thereafter.

Step-6: If shortest path is found then it is displayed.

Step-7: If no any path found then an "Impossible" message is displayed.

Based on this algorithm optimal cost is being calculated. The messenger moves with the speed of 1 distance per unit time. So optimal cost is calculated based on distance per unit time from source to destination and the cost rate of messenger per unit km to deliver the message.

Example:



4. System Requirements:

a. Hardware Configuration used:

I. Processor: Intel(R) core TM i3-4030U CPU@ 1.90GHz*4
II. Installed Memory-(RAM)-3.8 GB

b. Software Requirements:

I. System Type- 32 bit Operating System
II. Operating System Used: Linux
III. Compiler Used: gcc compiler
IV. Programming Language Used: C

5. Implementation Details:

a. User defined Functions:

1. dijkstra()

dijkstra() is a function which implements dijkstra's algorithm.

We have passed following parameters

1. node:

It represents the total number of places in the graph i.e., size of our adjacency matrix i.e., dist[][].

2. dist[][]:

dist[][] is our adjacency weighed matrix in which we have stored the distances between nodes

3. source:

source is the variable which represents the source node entered by the user.

4. dest:

dest is the variable which represents the destination node entered by the user.

We have used following variables in dijkstra() function:

1. p1[]:

In the variable p1[] we have stored the best known path i.e., shortest path between two nodes till known. The value of p1[] is tentative until we found the best path.

2. p_nm[]:

In the variable p_nm[] we have stored the name of previous node from which we have traced the path. The value of variable p_nm[] is also tentative it stores the name of that node which lies in best known path till then.

b. User Manual:

```
sandesh@sandesh-X555LA:~$ gcc dijexp.c
sandesh@sandesh-X555LA:~$ ./a.out

Enter the Enter the source node  :
1
Enter the destination node  :  3
Enter firs
Shortest path is :
7
Enter seco3 -> 1
Cost per kilometers is Rs.5
Total cost=35
Enter the enter 0 to stop0
sandesh@sandesh-X555LA:~$ ./a.out


Do you want to continue (Enter 0 to stop) :   2

Enter first node in which you want connection  :  2

Enter second node in which you want connection  :  3

Enter the distace bet them  :  6


Do you want to continue (Enter 0 to stop) :   1

Enter first node in which you want connection  :  1

Enter second node in which you want connection  :  3

Enter the distace bet them  :  7


Do you want to continue (Enter 0 to stop) :   0
0       8       7
8       0       6
7       6       0
```

6. References:

a. acm-icpc 2014

b. Book Reference:

Data Structures -Lipchutz

Computer networks- Andrew S. Tanenbaum

c. **web reference:**

www.geeksforgeeks.com

www.stackoverflow.com

www.qoura.com