Name : Ansari M.Saeem M.Saleem

Uid : 2019430001

Subject : NAD

Expt no : 4

Aim : Write a program to implement Link State Routing (Dijkstra Algorithm).

Aim:

Write a program to implement Link State Routing (Dijkstra Algorithm).

Objectives:

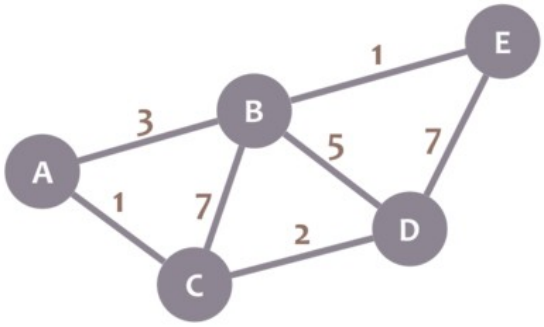
- To apply link state routing strategy to find optimal path for data transmission in a network.
- To find the shortest path from source node to every other node even using Dijkstra algorithm.
- To calculate routing table of a network.

Theory:

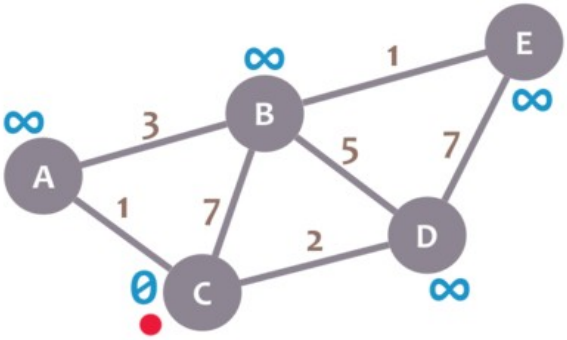
Link-State protocol is performed by every switching node in the network (i.e., nodes that are prepared to forward packets in the Internet, these are called routers). The basic concept of link-state routing is that every node constructs a map of the connectivity to the network, in the form of a graph, showing which nodes are connected to which other nodes. Each node then independently calculates the next best logical path from it to every possible destination in the network. Each collection of best paths will then form each node's routing table.

Dijkstra's algorithm finds a shortest path from a single source node, by building a set of nodes that have minimum distance from the source. This algorithm works for both directed and undirected graphs. It works only for connected graphs. The graph should not contain negative edge weights.

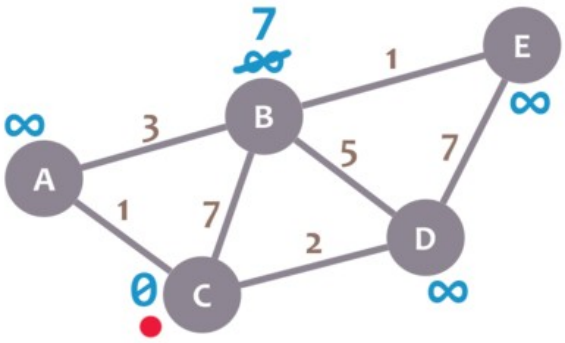
Methodology :



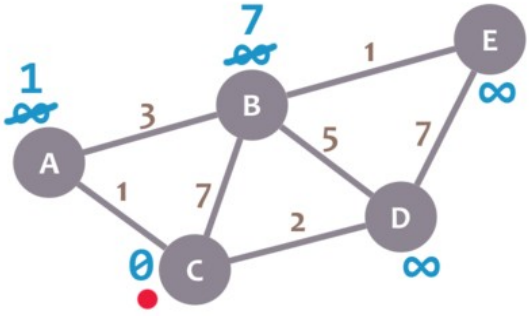
Step 1



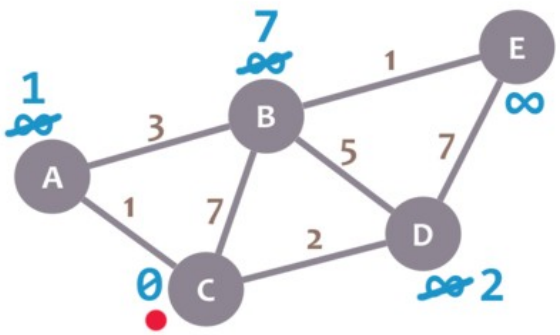
Step 2



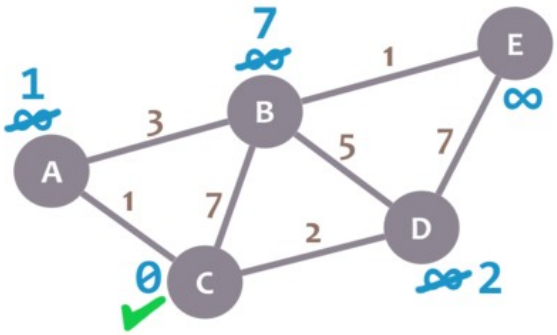
Step 3



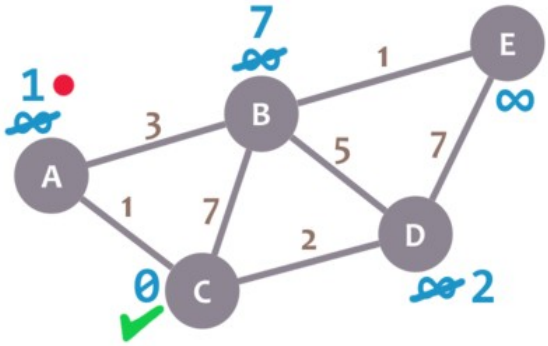
Step 4



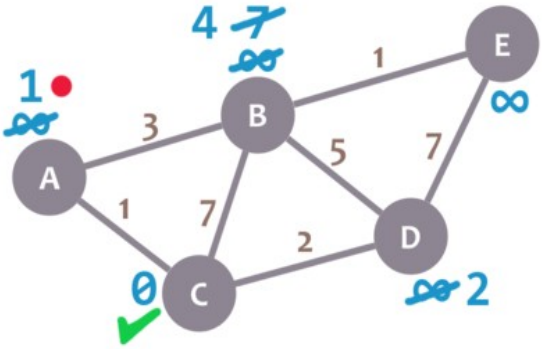
Step 5



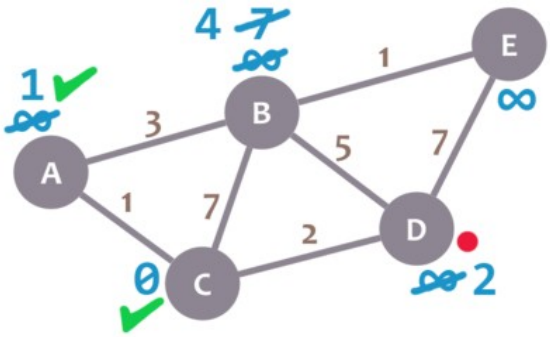
Step 6



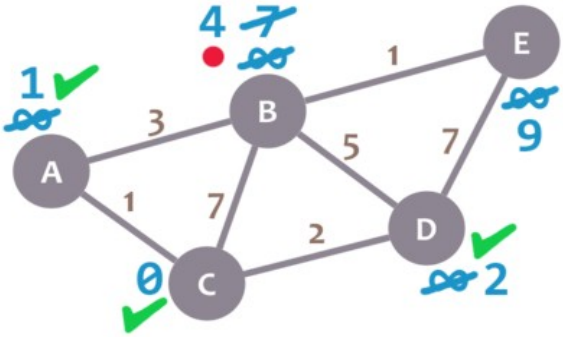
Step 7



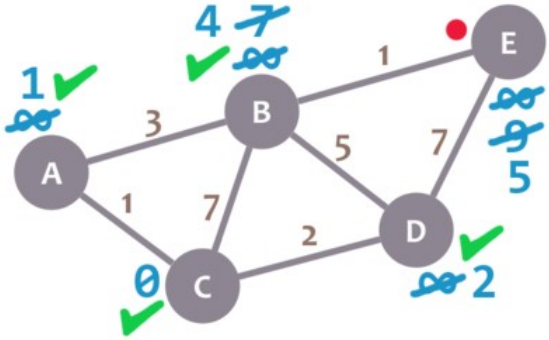
Step 8



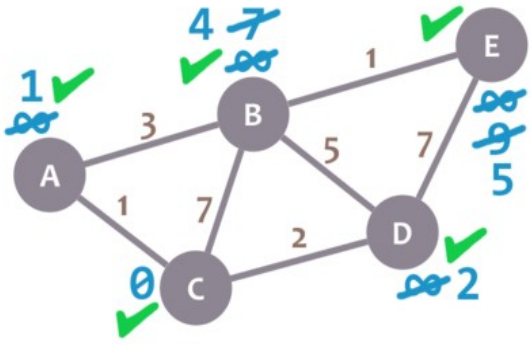
Step 9



Step 9



Step 10



Step 11

Working-

Following are the steps used for finding the solution-

Step 1; Set dist[s]=0, S=ϕ // s is the source vertex and S is a 1-D array having all the visited vertices

Step 2: For all nodes v except s, set dist[v]= ∞

Step 3: find q not in S such that dist[q] is minimum // vertex q should not be visited

Step 4: add q to S // add vertex q to S since it has now been visited

Step 5: update dist[r] for all r adjacent to q such that r is not in S //vertex r should not be

visiteddist[r]=min(dist[r], dist[q]+cost[q][r])

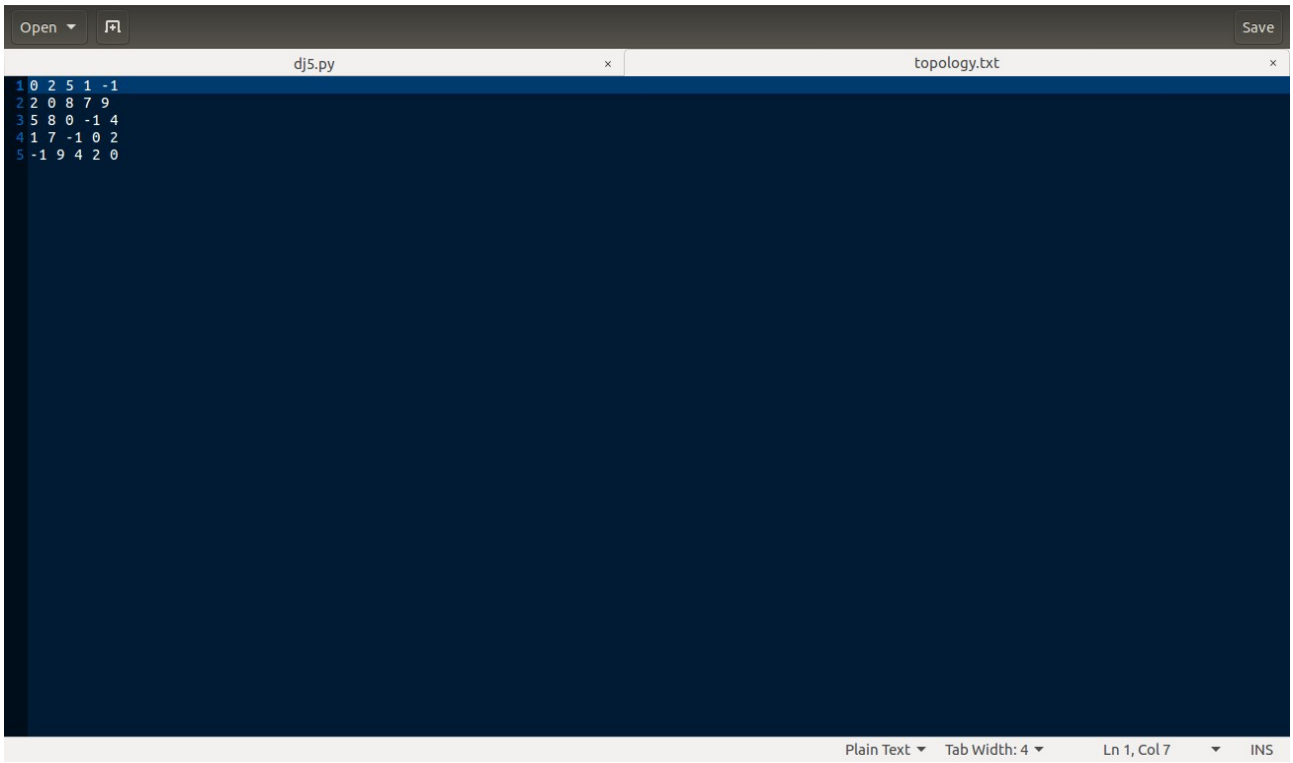
Step 6: Repeat Steps 3 to 5 until all the nodes are in S // repeat till all the vertices have been visited

Step 7: Print array dist having shortest path from the source vertex u to all other vertices

Step 8: Exit

Results :

Input File :



Output 1: Input File

Output :

```
students@CE-Lab3-603-U22:/media/students/TOSHIBA/sem2/nad/4$ python3 dj5.py
######################################################
Link State Routing Simulator

(1) Input Network Topology File
(2) Build a Connection Table
(3) Shortest Path to Destination Router
(4) Exit

######################################################


Command : 1

Input original network topology matrix data file[ NxN distance matrix. (value : -1 for no link, 0 for self loop) : topology.txt

Review original topology matrix:

0
2
5
1
-1

2
0
8
7
9

5
8
0
-1
4

1
7
-1
0
2

-1
```

Output 2: Showing Adjacency Matix

```
5                1

Command : 2

Select a source router : 3

Destination     Interface
1                1
2                1
3                None
4                5
5                5

Command : 2

Select a source router : 4

Destination     Interface
1                1
2                1
3                1
4                None
5                5

Command : 2

Select a source router : 5

Destination     Interface
1                4
2                4
3                3
4                4
5                None

Command : 3

Select a destination router : 1

The shortest path from router 5 to router 1 :
5
4
1
```

Output 3: Calculating Routing Table

```
1
7
-1
0
2

-1
9
4
2
0



Command : 2

Select a source router : 1

Destination     Interface
1                None
2                2
3                3
4                4
5                4

Command : 2

Select a source router : 2

Destination     Interface
1                1
2                None
3                1
4                1
5                1

Command : 2

Select a source router : 3

Destination     Interface
1                1
```

Output 4: Calculating Routing Table

```
3                1
4                None
5                5

Command : 2

Select a source router : 5

Destination     Interface
1                4
2                4
3                3
4                4
5                None

Command : 3

Select a destination router : 1

The shortest path from router 5 to router 1 :
5
4
1


The total cost is :  3

Command : 3

Select a destination router : 3

The shortest path from router 5 to router 3 :
5
3


The total cost is :  4

Command : 4

Good Bye!

students@CE-Lab3-603-U22:/media/students/TOSHIBA/sem2/nad/4$
```

Output 5: Calculating Shortest Path

Conclusion: Here we can conclude that Dijkstra's algorithm can be used to find the optimal path for data transmission in a network using link state routing strategy. We can also obtain the routing table showing cost from source node to destination node.