5) Experiment Name:- Distance vector Routing

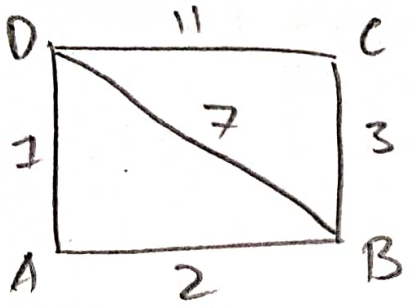
Software Name: code block

Theory: A network routing technique called distance vector routing - determines the shortest route between network nodes. Each node's routing table is repeatedly updated according to function The article will examine how a Distance Vector Program is implemented.

In order to find the best route to a destination, distance vector routing algorithms work on the premise that neighboring nodes exchange routing information. Each node has a routing table that details the costs involved in getting to other, nodes in the network.

The algorithm updates the - routing table with the lowest cost. Paths. The routing table with the lowest cost paths. The routing tables go through the Procedure repeatedly until they -

stabilize and converge. By sharing data with the ~~nearby~~ nearby nodes and weighing the costs of various routes to a destination, the routing tables are updated.



A Router

Min cost for reaching destination B = 2 via B
Min cost for reaching destination C = 5 via B
Min cost for reaching destination D = 1 via D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

B Router

MIN cost for reaching A = 2 via A
" " " " C = 3 via C
" " " " D = 3 via A

Destination	Distance	Next Hop
A	2	A
B	0	B
C	3	C
D	3	D

C Router

Min cost for reaching A = 2 via A
" " " " B = via B
" " " " D = via B

Destination	Distance	Next Hop
A	5	B
B	3	B
C	0	C
D	10	D

D Router

Min cost for reaching A = 1 via A
" " " " B = 3 via A
" " " " C = 10 via B

Destination	Distance	Next Hop
A	1	A
B	3	A
C	10	B
D	0	D

Code:

```
#include <stdio.h>
struct node {
  Unsigned dist [20];
  Unsigned from [20];
} rt [10];
int main() {
int costmat [20] [20];
int nodes, i, j, k, count = 0;
Printf ("\n Enter the number of nodes =");
Scanf ("%d", &nodes);
Printf ("\n Enter the cost matrix :");
```

```
for(i=0; i<nodes; i++)
{
for(j=0; j<nodes; j++)
{
scanf("%d", &costmat[i][j]);
costmat[i][j]=0;
rt[i].dist[j] = costmat[i][j];
rt[i].from[j] = j;
}
}
do{
count=0;
for(i=0; i<nodes; i++){
for(j=0; j<nodes; j++){
for(k=0; k<nodes; k++){
if(rt[i].dist[j] > costmat[i][k]+rt[k]
.dist[j]){
rt[i].dist[j] = costmat[i][k]+rt[k].dist
[j];
rt[i].from[j] = k;
}
}
}
```

```
count++;
}
}while(count<nodes);
for(i=0;i<nodes;i++){
Printf("\n\n For Router %d \n",i+1);
for(i=0; j<nodes; j++){
Printf("Node %d via %d Distanee %d \n",
j+1, rt[i].from[j]+1, rt[i].dist[j]);
}
}
return 0;
}
```

Output:

```
Enter the number of nodes = 4
Enter the cost matrix:
0 2 5 1
2 0 3 3
5 3 0 10
1 3 10 0

For Router 1
Node 1 via 1 Distanee 0
Node 2 via 2 Distanee 2
Node 3 via 3 Distance 5
```

Node 4 via 4 Distance 1

For Router 2

Node 1 via 1 Distance 2
Node 2 via 2 Distance 0
Node 3 via 3 Distance 3
Node 4 via 4 Distance 4

For Router 3

Node 1 via 1 Distance 5
Node 2 via 2 Distance 3
Node 3 via 3 Distance 0
Node 4 via 1 Distance 6

For Router 4

Node 1 via 1 Distance 1
Node 2 via 2 Distance 3
Node 3 via 1 Distance 6
Node 4 via 4 Distance 0

Singh
8/11/23