**WEEK-6**

Implement distance vector routing algorithm for obtaining routing tables at each node

Theory:  
Distance Vector routing (DVR) algorithm is unlike Dijkstra's algorithm which is a non-adaptive routing algorithm and means that it is purely static, that is pre-destined and fixed, not flexible in networks where congestions are more prone to occur. DVR is an adaptive routing algorithm in which the information from neighbours is maitained well by each and every node and this helps us to determine the simplest path possible in a changing network. Though, one of the node may fail, still, the destined node is reachable through other possible intermediate nodes that are found out by the DVR algorithm.

**PROGRAM:**

#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("\nEnter the number of nodes : ");

scanf("%d",&nodes);//Enter the nodes

printf("\nEnter the cost matrix :\n");

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

{

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

{

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

costmat[i][i]=0;

rt[i].dist[j]=costmat[i][j];//initialise the distance equal to cost matrix

rt[i].from[j]=j;

}

}

do

{

count=0;

for(i=0;i<nodes;i++)//We choose arbitary vertex k and we calculate the direct distance from the node i to k using the cost matrix

//and add the distance from k to node j

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

for(k=0;k<nodes;k++)

if(rt[i].dist[j]>costmat[i][k]+rt[k].dist[j])

{//We calculate the minimum distance

rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];

rt[i].from[j]=k;

count++;

}

}

while(count!=0);

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

{

printf("\n\n For router %d\n",i+1);

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

{

printf("\t\nnode %d via %d Distance %d ",j+1,rt[i].from[j]+1,rt[i].dist[j]);

}}

printf("\n\n");

return 0;

}

