EXPERIMENT - XI
DISTANCE VECTOR ROUTING PROTOCOL

April 6, 2020

ADITHYA D RAJAGOPAL
ROLL NO : 9
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM

AIM

To implement and simulate algorithm for distance vector routing protocol.

THEORY

Distance Vector Routing Protocol

Distance-Vector Routing Protocol in data networks is basically used to determine the best route for data packets based on distance. Distance-vector routing protocols measure the distance by the number of routers a packet has to pass through, where one router counts as one hop. In some cases, distance-vector protocols also take into account network latency and other factors that influence traffic on a given route. To determine the best route across a network routers, on which a distance-vector protocol is implemented, exchange information with one another, usually routing tables plus hop counts for destination networks and possibly other traffic information. Distance-vector routing protocols also require that a router informs its neighbours of network topology changes periodically.

Distance-vector routing protocols commonly use the Bellman–Ford algorithm and Ford–Fulkerson algorithm to calculate the best possible route. The term distance vector refers to the fact that the protocol manipulates vectors of distances to other nodes in the network. In the earlier days, this routing algorithm was implemented more widely in local area networks with the Routing Information Protocol(RIP).

ALGORITHM

Algorithm 1 Bellman-Ford algorithm

```
1: START
2: procedure BELLMANFORD
3:     Input the size(n) and elements of the element matrix.
4:     for i = 0 to n do
5:         for j = 0 to n do
6:             for k = 0 to n do
7:                 dist[i][j] = min(dist[i][j] , arr[i][k] + dist[k][j])
8:             end for
9:         end for
10:    end for
11: end procedure
12: STOP
```

SOURCE CODE

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

void main()
{
 int n,arr[10][10],dist[10][10],i,j,k;
 char s[10];
 printf("Enter the size of the distance matrix : ");
 scanf("%d",&n);
 printf("Enter * for non-neighbouring vertices...\n");
 printf("Enter the distance matrix:\n");
 for(i=0;i<n;i++)
 printf("\tN%d",i+1);
 printf("\n");
 for(i=0;i<n;i++)
 {
  printf("N%d\t",i+1);
  for(j=0;j<n;j++)
  {
   scanf("%s",s);
   if(strcmp(s,"*")==0)
   arr[i][j]=9999;
   else
   arr[i][j]=atoi(s);
   dist[i][j]=arr[i][j];
  }
 }
 for(i=0;i<n;i++)
 dist[i][i]=0;
 for(i=0;i<n;i++)
 for(j=0;j<n;j++)
 for(k=0;k<n;k++)
```

```
  if(dist[i][j]>arr[i][k]+dist[k][j])
  dist[i][j]=arr[i][k]+dist[k][j];
  printf("Reduced Distance Matrix:\n");
  for(k=0;k<n;k++)
  printf("\tN%d",k+1);
  printf("\n");
  for(i=0;i<n;i++)
  {
   printf("N%d",i+1);
   for(j=0;j<n;j++)
   printf("\t%d",dist[i][j]);
   printf("\n");
  }
}
```

OUTPUT

```
user@adithya:~/s6/np$ gcc p11.c
user@adithya:~/s6/np$ ./a.out
Enter the size of the distance matrix : 3
Enter * for non-neighbouring vertices...
Enter the distance matrix:
        N1      N2      N3
N1      0       1       5
N2      1       0       2
N3      5       2       0
Reduced Distance Matrix:
        N1      N2      N3
N1      0       1       3
N2      1       0       2
N3      3       2       0
user@adithya:~/s6/np$
```

RESULT

Implemented and simulated the algorithm for distance vector routing protocol using C.