DESIGN AND ANALYSIS OF ALGORITHMS

EXPERIMENT 6

Name: Janhavi R Deshmukh

Branch: CSE(DS) D1

UID: 2021700017

Aim: –Experiment on Greedy approach-single source shortest path Dijktra's algoritm.

Theory:

The algorithm maintains a set of visited vertices and a set of unvisited vertices. It starts at the source vertex and iteratively selects the unvisited vertex with the smallest tentative distance from the source. It then visits the neighbours of this vertex and updates their tentative distances if a shorter path is found. This process continues until the destination vertex is reached, or all reachable vertices have been visited.

Algorithm:

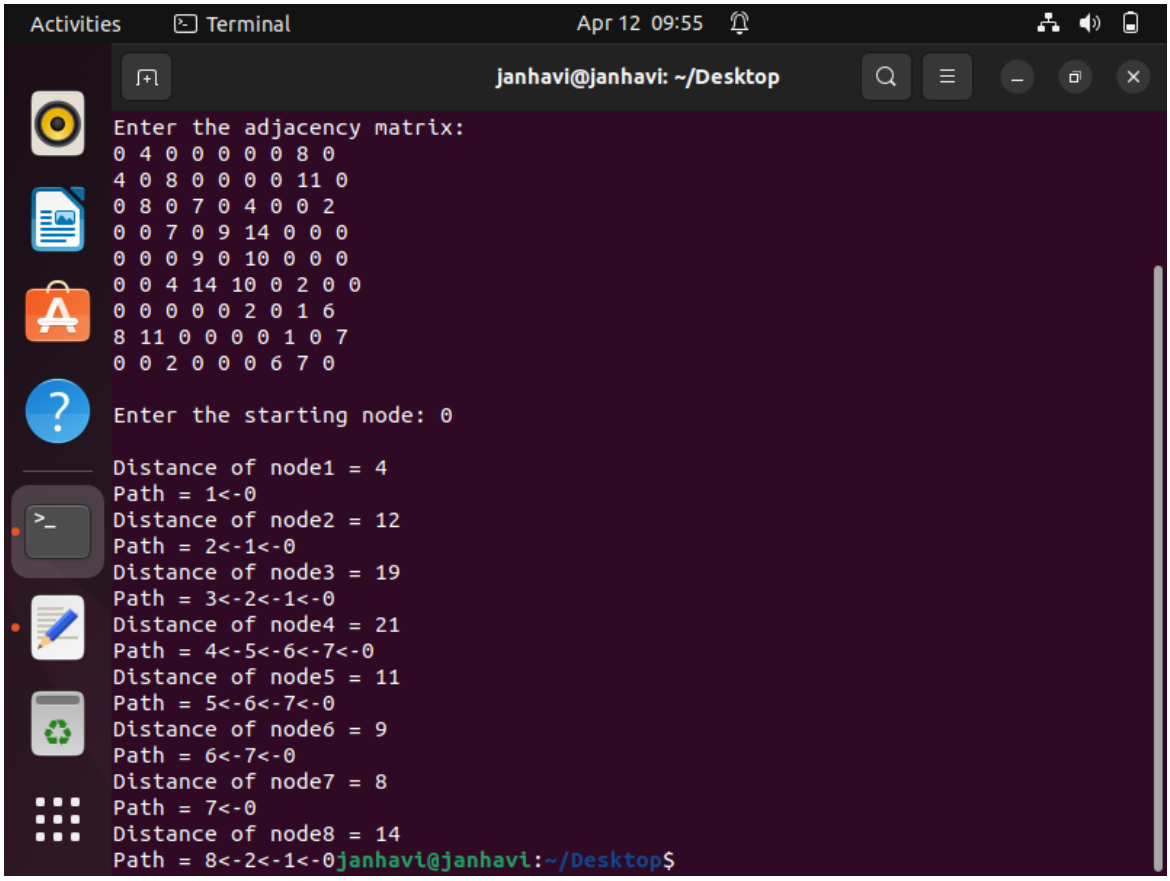
1. A tentative distance value is assigned to every node; this value is set to zero for the initial node, and to infinity for all other nodes.
2. All nodes unvisited are marked, and the initial node is set as current. An unvisited set (a set of all the unvisited nodes) consisting of all the nodes is created.
3. For the current/initial node, take into account all the unvisited nearby nodes, and calculate their tentative distances. Make a comparison of the current assigned value and the newly calculated tentative distance; assign the smaller value. For example: if the current/initial node *X* was marked with a distance of 4, and the edge connecting it with a nearby neighbour node *Y* has length 1, then the distance through *X* to *Y* will be 4 + 1 = 5. If *Y* was previously assigned a value greater than 5, then change it to 5. Otherwise, keep the value as it is.
4. A visited node is never to be checked again. So, after finishing above steps with all the neighbours of the current node, make that node as visited and remove is from the unvisited set.
5. Stop the algorithm if, when planning a route between two specific nodes, the destination node has been marked visited.
6. Also, stop the algorithm if, when planning a complete traversal, the smallest tentative distance among the nodes in the unvisited set is infinity. This case is a result of no connection between the initial node and the remaining unvisited nodes.
7. Find the unvisited node assigned with the smallest tentative distance value, and this will be the new "current mode". Go back to step 3, and continue.

Program:

```
#include<stdio.h>
#include<stdlib.h>
#define INFINITY 9999
#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode)
{
int cost[MAX][MAX],distance[MAX],pred[MAX];

int visited[MAX],count,mindistance,nextnode,i,j;
//pred[] stores the predecessor of each node
//count gives the number of nodes seen so far
//create the cost matrix
for(i=0;i<n;i++)
for(j=0;j<n;j++)
if(G[i][j]==0)
cost[i][j]=INFINITY;
else
cost[i][j]=G[i][j];
//initialize pred[],distance[] and visited[]
for(i=0;i<n;i++)
{
distance[i]=cost[startnode][i];
pred[i]=startnode;
visited[i]=0;
}
distance[startnode]=0;
visited[startnode]=1;
count=1;
while(count<n-1)
{
mindistance=INFINITY;
//nextnode gives the node at minimum distance
for(i=0;i<n;i++)
if(distance[i]<mindistance&&!visited[i])
{
mindistance=distance[i];
nextnode=i;
}
//check if a better path exists through nextnode
visited[nextnode]=1;
for(i=0;i<n;i++)
if(!visited[i])
if(mindistance+cost[nextnode][i]<distance[i])
{
distance[i]=mindistance+cost[nextnode][i];
pred[i]=nextnode;
```

```
}
count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
if(i!=startnode)
{
printf("\nDistance of node%d = %d",i,distance[i]);
printf("\nPath = %d",i);
j=i;
do
{
j=pred[j];
printf("<-%d",j);
}while(j!=startnode);
}
}

int main()
{
int G[MAX][MAX],i,j,n,u;
printf("\n\t--DIJKSTRA'S ALGORITHM--");
printf("\n\nEnter no. of vertices: ");
scanf("%d",&n);
printf("\nEnter the adjacency matrix: \n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&G[i][j]);
printf("\nEnter the starting node: ");
scanf("%d",&u);
dijkstra(G,n,u);
return 0;
}
```

Output:



Conclusion:

I understood how to find the shortest single path by Dijktra;s algorithm using greedy approach.