University of Information Technology and Sciences (UITS)

Lab Report - 7

IT-214 : Algorithm Lab

# Dijkstras Algorithm

*Submitted To:*

Sumaiya Akhtar Mitu
Lecturer,
Department of IT, UITS

*Submitted By:*

Name: Nazmul Zaman
Student ID:2014755055
Department of IT UITS

3 july 2022

## Contents

## 1 Abstrac

In mathematics and computer science, an algorithm is a finite sequence of well-defined instructions, typically used to solve a class of specific problems or to perform a computation. Algorithms are used as specifications for performing calculations and data processing.

## 2 Introduction

In this task we are going to learn about Dijkstra's algorithm and implementation .

Dijkstra's algorithm allows us to find the shortest path between any two vertices of a graph.

It differs from the minimum spanning tree because the shortest distance between two vertices might not include all the vertices of the graph.

Dijkstra's Algorithm Complexity

Time Complexity: O(E Log V)

where, E is the number of edges and V is the number of vertices.

Space Complexity: O(V)

## 3 Theory

Dijkstra's Algorithm finds the shortest path between a given node (which is called the "source node") and all other nodes in a graph. This algorithm uses the weights of the edges to find the path that minimizes the total distance (weight) between the source node and all other nodes

## 4 Procedure

Dijkstra's algorithm :
We need to maintain the path distance of every vertex. We can store that in an array of size v, where v is the number of vertices.

We also want to be able to get the shortest path, not only know the length of the shortest path. For this, we map each vertex to the vertex that last updated its path length.

Once the algorithm is over, we can backtrack from the destination vertex to the source vertex to find the path.

A minimum priority queue can be used to efficiently receive the vertex with least path distance.

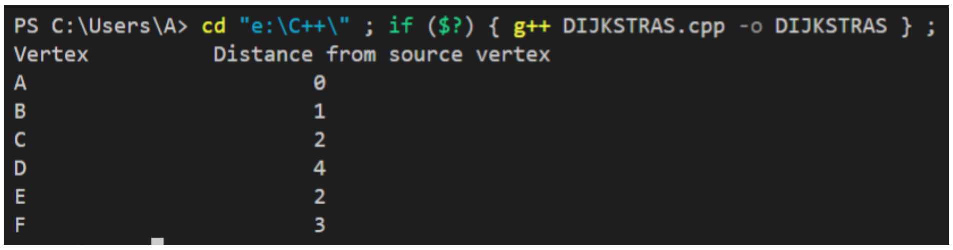
## 5 Code Implementation

### 5.1 ( Dijkstras Algorithm

```cpp

// Nazmul Zaman Bsc in IT

#include<iostream>
#include<climits>
using namespace std;

int miniDist(int distance[], bool Tset[]) // finding minimum distance
{
    int minimum=INT_MAX,ind;

    for(int k=0;k<6;k++)
    {
        if(Tset[k]==false && distance[k]<=minimum)
        {
            minimum=distance[k];
            ind=k;
        }
    }
    return ind;
}

void DijkstraAlgo(int graph[6][6],int src) // adjacency matrix
{
    int distance[6]; // // array to calculate the minimum distance for each node
    bool Tset[6];// boolean array to mark visited and unvisited for each node


    for(int k = 0; k<6; k++)
    {
        distance[k] = INT_MAX;
        Tset[k] = false;
    }

    distance[src] = 0;   // Source vertex distance is set 0

    for(int k = 0; k<6; k++)
    {
        int m=miniDist(distance,Tset);
        Tset[m]=true;
        for(int k = 0; k<6; k++)
        {
            // updating the distance of neighbouring vertex
            if(!Tset[k] && graph[m][k] && distance[m]!=INT_MAX &&
    distance[m]+graph[m][k]<distance[k])
                distance[k]=distance[m]+graph[m][k];
        }
    }
    cout<<"Vertex\t\tDistance from source vertex"<<endl;
    for(int k = 0; k<6; k++)
    {
        char str=65+k;
        cout<<str<<"\t\t\t"<<distance[k]<<endl;
    }
}

int main()
{
    int graph[6][6]={
        {0, 1, 2, 0, 0, 0},
        {1, 0, 0, 5, 1, 0},
        {2, 0, 0, 2, 3, 0},
        {0, 5, 2, 0, 2, 2},
        {0, 1, 3, 2, 0, 1},
        {0, 0, 0, 2, 1, 0}};
    DijkstraAlgo(graph,0);
    return 0;
}


Output

Vertex
```

?

## 5.2 Output



```
PS C:\Users\A> cd "e:\C++\" ; if ($?) { g++ DIJKSTRAS.cpp -o DIJKSTRAS } ;
Vertex          Distance from source vertex
A                       0
B                       1
C                       2
D                       4
E                       2
F                       3
```

Figure 1: OUTPUT

## 6 Conclusion

In this lab report we learn Dijkstra's algorithm and there implementation deeply.It solves the shortest-path problem for any weighted, directed graph with non-negative weights. It can handle graphs consisting of cycles, but negative weights will cause this algorithm to produce incorrect results its also helps to our real life as like : It is used in Google Maps.
It is used in finding Shortest Path.
It is used in geographical Maps.

## 7 References

1.https://www.w3schools.com/algorithm/algo$_{i}ntro.asp$

2.https://www.javatpoint.com/algo$_{i}ntro.asp$

3.https://github.com/ZamanNazmul