Computer Networks - Exp 4

Kartik Jolapara

60004200107 - B1

Aim

To implement Djikstra’s shortest path algorithm.

Theory

The Dijkstra Algorithm is a very famous greedy algorithm. It is used for solving the single source shortest path problem. It computes the shortest path from one particular source node to all other remaining nodes of the graph. Condition: It is important to note the following points regarding Dijkstra AlgorithmDijkstra algorithm works only for connected graphs. The Dijkstra algorithm works only for those graphs that do not contain any negative weight edge. The actual Dijkstra algorithm does not output the shortest paths. It only provides the value or cost of the shortest paths. By making minor modifications in the actual algorithm, the shortest paths can be easily obtained. The Dijkstra algorithm works for directed as well as undirected graphs. Working of Dijkstra's Algorithm: Dijkstra's Algorithm works on the basis that any subpath B -> D of the shortest path A -> D between vertices A and D is also the shortest path between vertices B and D. Each subpath is the shortest path Djikstra used this property in the opposite direction i.e we overestimate the distance of each vertex from the starting vertex. Then we visit each node and its neighbors to find the shortest subpath to those neighbors. The algorithm uses a greedy approach in the sense that we find the next best solution hoping that the end result is the best solution for the whole problem.

Code

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

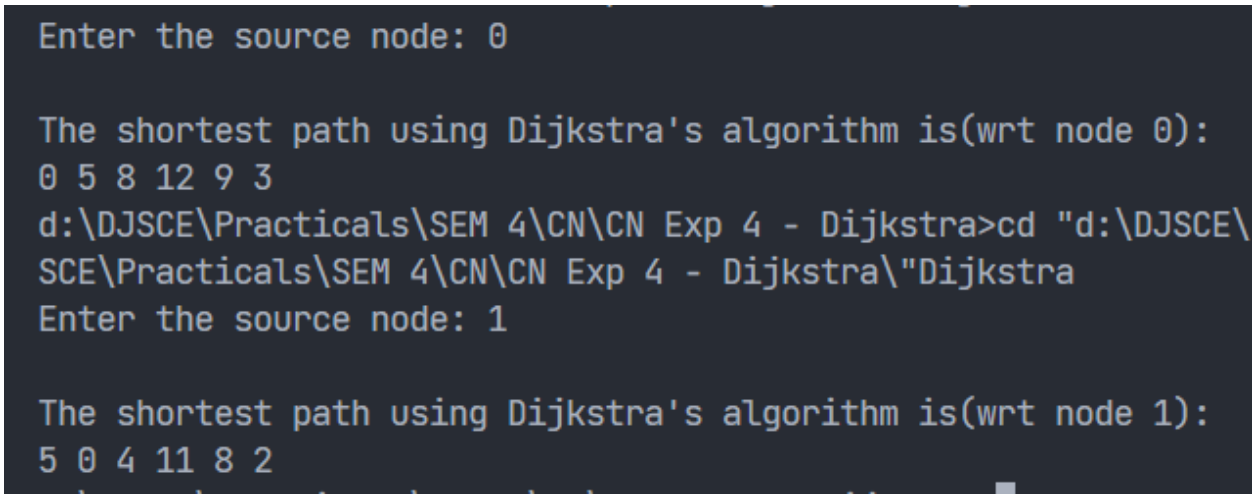
```
#include <stdbool.h>

#define MIN(x, y) (((x < y) ? x : y))

int main()
{
    int n = 6;
    int cost[6][6] = {
        {0, 7, 42069, 42069, 42069, 3},
        {7, 0, 4, 42069, 42069, 2},
        {42069, 4, 0, 8, 5, 5},
        {42069, 42069, 8, 0, 3, 42069},
        {42069, 42069, 5, 3, 0, 6},
        {3, 2, 5, 42069, 6, 0}};
    int vis[6] = {0, 0, 0, 0, 0, 0};
    int d[6];
    int source;
    printf("Enter the source node: ");
    scanf("%d", &source);
    vis[source] = 1;
```

```
for (int i = 0; i < n; i++)
{
    d[i] = cost[source][i];
}

// for (int i = 0; i < n; i++)
// {
//     printf("%d HEREEE %d\n", d[i], vis[i]);
// }
for (int i = 0; i < 6; i++)
{
    int min, check = 1;
    for (int j = 0; j < 6; j++)
    {
        if (vis[j] == 0 && check == 1)
        {
            min = j;
            check = 0;
        }
        if (vis[j] == 0 && d[j] < d[min])
        {
```

```
            min = j;
        }
        // printf("%d ", d[j]);
    }
    // printf("\n");
    // printf("ANS: %d\n", min);
    vis[min] = 1;
    for (int j = 0; j < 6; j++)
    {
        // d[j] = (d[j] < d[min] + cost[min][j]) ? d[j] : (d[min] + cost[min][j]);
        d[j] = MIN(d[j], (d[min] + cost[min][j]));
    }
    // for (int j = 0; j < n; j++)
    // {
    //     printf("D%d ", d[j]);
    // }
    // printf("\n");
}
printf("\n");
printf("The shortest path using Dijkstra's algorithm is(wrt node %d): \n", source);
for (int i = 0; i < n; i++)
```

```
    {
        printf("%d ", d[i]);
    }

    return 0;
}
```

Output



Conclusion

Dijkstra’s shortest path algorithm has been successfully executed.