



Pune Institute of Computer Technology

Dhankawadi,Pune

Maharashtra 411043

DEPARTMENT OF COMPUTER ENGINEERING

HPC MINI PROJECT REPORT

ON

“Dijkstra’s Algorithm using OpenMPI”

Submitted by:

Roll No. 41414 Riya Disawal

Roll No. 41412 Ritika Deshpande

Roll No. 41402 Aditi Kukde

Under the Guidance of

Prof. Hemant Shinde

Abstract

Dijkstra's Shortest Path First algorithm, SPF algorithm is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. For a given source node in the graph, the algorithm finds the shortest path between that node and every other. For big data sets this can take a long time for finding the shortest distance between source and destination nodes. Multi-core programming can accelerate the time taken. By parallelization not only do we significantly speed-up the compilation, but enable fast completion and getting output using OpenMPI.

Table of Contents

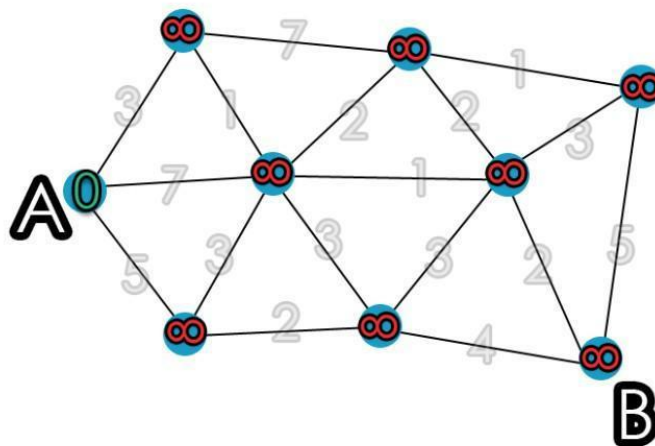
1.INTRODUCTION	04
2.OBJECTIVE	05
3.TEST CASES	06
4.RESULT	07
5.CONCLUSION	08
6.REFERENCES	09

1.Introduction

One algorithm for finding the shortest path from a starting node to a target node in a weighted graph is Dijkstra's algorithm. The algorithm creates a tree of shortest paths from the starting vertex, the source, to all other points in the graph. The Dijkstra algorithm uses labels that are positive integers or real numbers, which are totally ordered. It can be generalized to use any labels that are partially ordered, provided the subsequent labels (a subsequent label is produced when traversing an edge) are monotonically non-decreasing. This generalization is called the generic Dijkstra shortest-path algorithm.

The graph has the following:

- vertices, or nodes, denoted in the algorithm by v or u ;
- weighted edges that connect two nodes: (u,v) denotes an edge, and $w(u,v)$ denotes its weight.



2. Objective

- To implement a Dijkstra's algorithm using parallelism.
- To compare the speed-up and efficiency of serial and parallel implementation of Dijkstra's algorithm.

3.Test Case

- 1)

6 8

0140

0215

1220

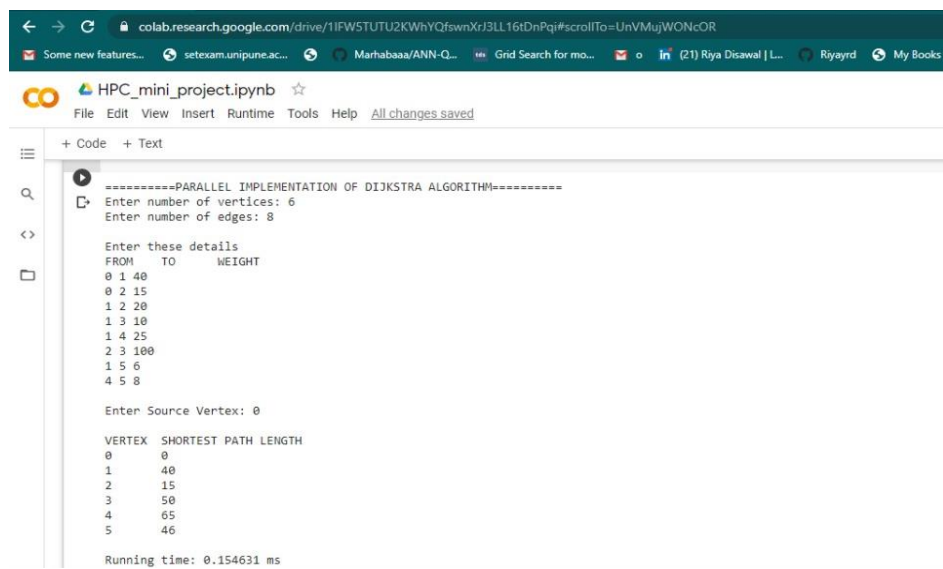
1310

1425

23100

1 5 6

4 5 8



```
=====PARALLEL IMPLEMENTATION OF DIJKSTRA ALGORITHM=====
Enter number of vertices: 6
Enter number of edges: 8

Enter these details
FROM    TO    WEIGHT
0 1 40
0 2 15
1 2 20
1 3 10
1 4 25
2 3 100
1 5 6
4 5 8

Enter Source Vertex: 0

VERTEX  SHORTEST PATH LENGTH
0       0
1       40
2       15
3       50
4       65
5       46

Running time: 0.154631 ms
```

5. Result

Shortest Path between source and destination could be found out using Dijkstra's Algorithm with parallel computation using OpenMPI.

6. Conclusion

We successfully completed Dijkstra's algorithm using MPI and parallelization.

7. References

- <https://www.geeksforgeeks.org>
- <https://www.open-mpi.org/doc/v4.0/>
- https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
- <https://brilliant.org/wiki/dijkstras-short-path-find>