**Abstract**

Dijkstra’s Algorithm is considered one of the best shorted path algorithms used in maps for generating a shortest route. To understand how maps give us the shortest path from source to destination, we are practically applying Dijkstra’s algorithm on maps.

The major purpose of this project was to learn to apply shortest path algorithm i.e., Dijkstra’s algorithm on map.

Since it’s not easy to create a map ourselves we are extracting map data from OpenStreetMap which is an open and free to use service. Using this data, we are constructing an adjacency matrix to represent connections between any two nodes.

The same matrix is passed to Dijkstra’s Algorithm through which we get the shortest paths from source node to every other node. With the help of folium library, we are generating a map showing start and destination markers and a blue dotted line showing the path.

In this report we are giving the information of all the libraries we used along with a flow chart of how the program is working. Along with some snapshots of the output.

**Table of Contents**

|  |  |
| --- | --- |
| **Abstract** | **i** |
| **Table of Contents** | **ii** |
| **1** **Introduction** | **1** |
| Problem Statement |  |
| **2 Requirement Specification** |  |
| 2.1 Introduction |  |
| 2.2 Hardware Requirement / Software Requirement specification |  |
| **3 Analysis and Design** |  |
| **4 Implementation** |  |
| 4.1 Programming language for implementation |  |
| 4.2 Data Structures/ Packages used/ **Libraries** used |  |
| **5 Applications and Conclusion** |  |
| 5.1 Applications |  |
| 5.2 Conclusion |  |
| **6** **Snapshots** |  |

**1. Introduction**

**PROBLEM STATEMENT**

Extract and parse xml data from a .osm (OpenStreetMap) file, apply Dijkstra's Shortest Pathfinding Algorithm and display the paths generated on a map.

**2. Requirement Specification**

**2.1 Introduction**

**OpenStreetMap**: OpenStreetMap (OSM) is a collaborative project to create a free editable map of the world. We are using specific part of map using longitude and latitude.

**Folium**: It is a python library makes it easy to visualize data that’s been manipulated in Python on an interactive leaflet map. It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map.

The library has a number of built-in tilesets from OpenStreetMap, Map box, and Stamen, and supports custom tilesets with Map box or Cloud made API keys.

**2.2 Hardware Requirement / Software Requirement specification**

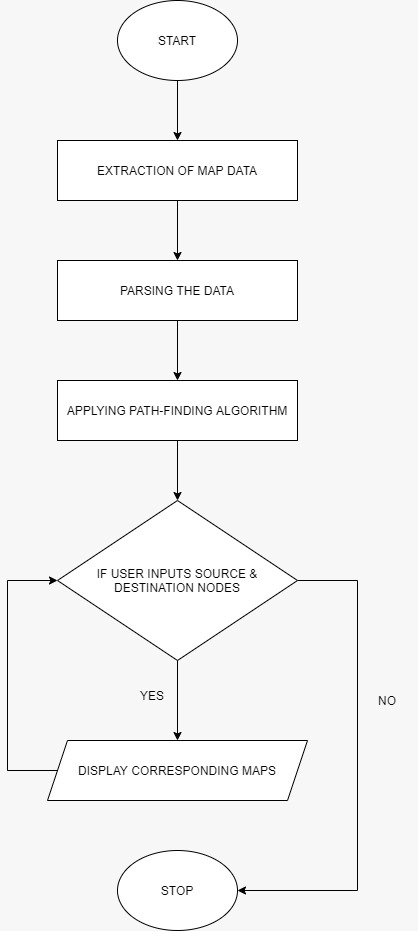
Hardware Requirement specification:

* 6th Gen Intel Core i3 and above
* 4 GB RAM and above

Software Requirement specification:

* Windows 7 and above
* A web-browser
* Python Interpreter

**3. Analysis and Design**



**4. Implementation**

**4.1 Programming language for implementation**

Python: Python is an interpreted high-level general-purpose programming language. Its amazing libraries and various modules make suitable for our project.

**4.2 Data Structures/ packages used/libraries used**

**Data structures:**

* List
* Tuple
* Dictionary

**Libraries:**

* xmltodict
* folium
* numpy
* webbrowser
* os
* sys

**5. Applications and Conclusion**

**5.1 Applications**

As the title suggests applying shortest path algorithm on maps, so one of the main applications of our project is finding shortest path which helps in navigation as well. When we have to find the shortest distance in maps from one place to another, or from your location to the nearest desired location.

Here comes our project shortest path algorithms which should be applied in maps to find shortest path to a desired location, as there are various routes/paths connecting them but it has to be minimum distance.

**5.2 Conclusion**

Working on this project was a wonderful learning experience for us.

This project took us through various phases of project development and gave us real insight into how Dijkstra's Algorithm works.

Here are the things we learned:

\* Parsing the map data provided and creating an adjacency matrix from it

\* Modifying the matrix so as to only consider roads that are traversable (unlike sewage, waterways, etc.)

\* Applying Dijkstra's Algorithm on this matrix to generate the shortest paths between any two co-ordinates

\* Creating a UI to input desired source & destination nodes

\* Displaying the shortest path between these nodes on an interactive map

**6. Snapshots**

