**Software Requirements Specification**

For

**Comparative Analysis of Shortest Path Algorithms**

May 2022

Prepared by

|  |  |  |
| --- | --- | --- |
| **Sap Id** | **Roll no.** | **Name** |
| **500076347** | **R177219143** | Rajneesh |
| **500075224** | **R177219170** | Shantanu Jaswal |
| **500076581** | **R177219178** | Somya Sharma |
| **500076703** | **R177219209** | Vanshika Parashar |



Department of Informatics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| Table of Content | | 2 |
| Revision History | | 3 |
| 1 | Introduction | 4 |
|  | 1.1 Purpose of the Project | 4 |
|  | 1.2 Target Beneficiary | 4 |
|  | 1.3 Project Scope | 4 |
|  | 1.4 References | 4 |
| 2 | Project Description | 4 |
|  | 2.1 Reference Algorithm | 4 |
|  | 2.2 Data/ Data structure | 5 |
|  | 2.3 SWOT Analysis | 5 |
|  | 2.4 Project Features | 5 |
|  | 2.5 User Classes and Characteristics | 5 |
|  | 2.6 Design and Implementation Constraints | 5 |
|  | 2.7 Assumption and Dependencies | 6 |
| 3 | System Requirements | 6 |
|  | 3.1 User Interface | 6 |
|  | 3.2 Software Interface | 6 |
|  | 3.3 Database Interface | 6 |
|  | 3.4 Protocols | 6 |
| 4 | Non-functional Requirements | 6 |
|  | 4.1 Performance requirements | 6 |
|  | 4.2 Security requirements | 6 |
|  | 4.3 Software Quality Attributes | 6 |
| 5 | Other Requirements | 7 |
| Appendix A: Glossary | | 7 |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Change** | **Reason for Changes** | **Mentor Signature** |
| 21/03/2022 | GANTT CHART CHANGES | SMALL ERROR |  |
| 19/04/2022 | TEXT SIZE CHANGES | NOT ORDERED PROPERLY |  |

1

1. INTRODUCTION

1.1 Purpose of the Project

The main reason to choose this topic is to compare and analyze the algorithms for finding the Shortest Path by using the Dijkstra, Bellman-Ford and Floyd-Warshall Algorithm Comparing, analyzing and plotting the graphs of each algorithm makes it easy to understand which algorithm is best for finding the shortest path for a particular type of question.

This project is created to find the shortest path very easily. Shortest Path Algorithm states that the algorithm to find the path which has minimal Distance (or path) between two nodes (or vertices). There are some Examples where the shortest path algorithm can be used like Web mapping like: Google Maps, Transportation Systems, Computer Networks like the Internet, route planning etc. [1].

1.2 Target Beneficiary

The main objective of this project is to compare all the algorithms by plotting their graphs on

the basis of their time complexity and give the output in written form as well as in graphical

form step by step which makes it easy to understand the shortest path and tells which

Algorithm should be used to solve the question more efficiently for a particular type of

question.

1.3 Project Scope

* Least-cost paths are calculated for instance to establish tracks of electricity lines

and oil pipelines.

* Network Routing Protocol.
* Road Networks.
  1. References

1. Amgad Madkour, Walid G. Aref, Faizan Ur Rehman, Mohamed Abdur Rahman, Saleh Basalamah. A Survey of Shortest-Path Algorithms. Purdue University, West Lafayette, USA, Umm Al-Qura University, Makkah, KSA, may 8, 2017.
2. Kairanbay Magzhan, Hajar Mat Jani ― A Review and Evaluations of Shortest Path Algorithms, the international journal of scientific & technology research volume 2, issue 6, June 2013, pp. 99-104.

# ‘Shortest path problem’ Wikipedia. Available at: <https://en.wikipedia.org/wiki/Shortest_path_problem>

# ‘Dijkstra’s shortest path algorithm’ GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/>.

# ‘Bellman–Ford Algorithm’ GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23>

2

# ‘Floyd Warshall Algorithm’ GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16>.

* PROJECT DESCRIPTION

2.1 Reference Algorithm

* Dijkstra algorithm- Given a graph and a source vertex in the graph, find the shortest paths from the source to all vertices in the given graph. [1]

Dijkstra’s Algorithm is a single-source shortest path problem algorithm which has non-negative edges weight.

Single source shortest path states that finding the shortest path from the source vertex to all other vertices [3].The node at which we are starting is called the initial node. It was discovered by the computer scientist Edsger W. Dijkstra in 1956.

* Bellman-Ford Algorithm- Given a graph and a source vertex source in graph, find shortest paths [2] from source to all vertices in the given graph. The graph may contain negative weight edges. [5]. Bellman-Ford Algorithm isa single-source shortest path problem algorithm which can have negative edge weight also. It was founded by Alfonso Shimbel in 1955, yet is rather named after Richard Bellman and Lester Ford Jr., who distributed it in 1958 and 1956. Bellman-Ford algorithm works on the path from the starting vertex to the other vertices. It is more versatile and has the capability to solve the problem by the graph which contains negative edge weights. In real life, some examples where negative weight edges can exist like cash flow, heat released in a chemical reaction, heat absorbed in chemical reaction etc. [3]. If a graph contains a "negative cycle" (cycle whose edges sum is negative value).
* Floyd Warshall- The problem is to find shortest distances between every pair of vertices in a given edge weighted directed Graph. [6] This algorithm is used for finding the shortest path between all pairs of vertices. The Floyd-Warshall Algorithm was proposed by Robert Floyd in 1962. The graph used in Floyd-Warshall Algorithm is a weighted graph with positive or negative edge weights. The Floyd-Warshall Algorithm is used for both directed and undirected weighted graphs. It is applied in negative edges but not for negative cycles (sum of edges in cycle is equal to negative value) [3]. Floyd-Warshall Algorithm is also known by different names as Floyd’s algorithm, Roy-Floyd algorithm, WFI algorithm or Roy-Warshall algorithm.

2.2 Characteristic of Data

We will test our dataset through various test cases and scenarios to ensure that the program runs efficiently and produces desired results in different scenarios.

2.3 SWOT Analysis

Strengths:

3

* Representation with the help of graph makes it easier to understand.

Weaknesses:

* It is a very challenging task as the path that is calculated by the algorithm will not be accurate.

Opportunities:

* Would save time and money of the tourists.

Threats:

* A small mistake can cause big changes.

2.4 Project Features

Shortest Path Algorithm states that the algorithm to find the path which has minimal Distance (or path) between two nodes (or vertices).A graph is a mathematical or pictorial representation which contains vertices and edges. With the help of edges, it is possible to walk from one vertex to another vertex because edges are a particular type of line segment joining two vertices [2]. By drawing the graph of any problem then it will be much easier to understand, so it helps people to get knowledge about the concept of the question in a short period of time.

2.5 Design and Implementation Constraints

We have implemented the program in the C language.

4

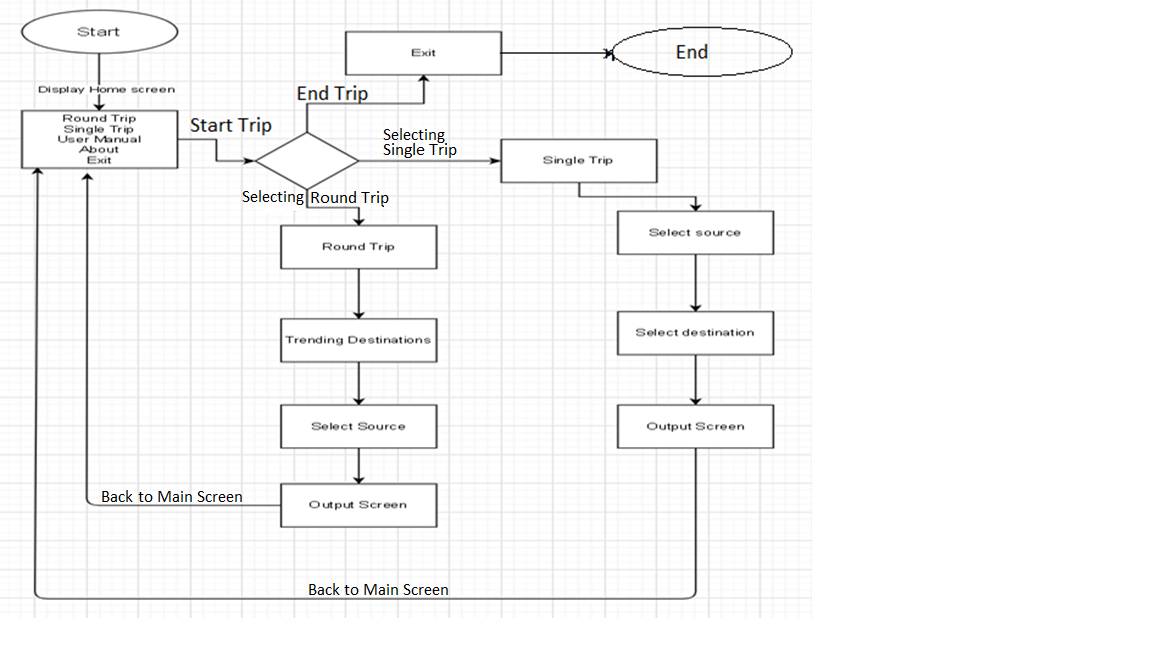


Fig I: Work flow diagram

2.7 Assumption and Dependencies

We assumed that all the data that is collected is accurate. Our project depends on the distance between the cities and the algorithms used.

Also, the values taken are accurate.

3. SYSTEM REQUIREMENTS

3.1 User Interface

We are using Ubuntu Operating System, various C supported IDE Brackets. So, it has various Graphic functions like syntax highlighting, scroll features, graphical windows, etc. These Graphic applications provide menus, toolbars, panes, containers, grids allowing easy controls with the help of various input devices like mouse and keyboard.

3.2 Software Interface

Software interfaces (programming interfaces) are the languages, codes, and messages that programs use to communicate with each other and to the hardware. Examples are the Ubuntu Operating System. In this project, we are using C as a programming language.

5

3.3 Database Interface

We will use a flat-file database. It is a database stored in a file called a flat file. Records follow a uniform format, and there are no structures for indexing or recognizing relationships between records. A flat file can be a plain text file, or a binary file.

3.4 Protocols

No external protocols are used. Project is developed in c only.

4. NON-FUNCTIONAL REQUIREMENTS

4.1 Performance requirements

No such timing relationship for the real-time system as of now.

4.2 Security requirements

The system has no extra security features.

4.3 Software Quality Attributes

MAINTAINABILITY - it is easy to maintain as we don’t need to add anything in the code.

USABILITY – Brackets is easy to use and code in it.

CORRECTNESS - The application's functionality, internal calculations, and algorithm should all be proper.

TESTABILITY – It is easy to test and find defects in the system.

FLEXIBILITY – The code is flexible enough to modify.

INTEROPERABILITY - It should be simple for a product to communicate data or services with other systems. Different system modules should be compatible with various operating systems (macOS, Windows, Linux) & IDEs.

# APPENDIX A: GLOSSARY

SRS: Software Requirement Specification

IDE: Integrated Development Environment

GUI: Graphical User Interface