## INTRODUCTION

Problem statement of the project is “Space - Efficient Algorithms for Reachability in Directed Geometric Graphs

Using the problem statement as the base we came up with an application to find the shortest distance between a source to destination for a warehouse robot system.

The domain of the application comes under graph exploration and implementations

The applications will find the shortest path for robot so it can spend less resources reaching that destination.

Since the computers in moving robots have space and power limitations, using shortest distance technique will help us achieve better efficiency

The goal of graph Reachability is to determine whether each graph has a path from one vertex to another.

We will aim to construct a space-efficient algorithm for reachability in a directed geometric graph in this project.

The application will determine the quickest course for the robot to take in order for it to use fewer resources to reach its objective.

Because the computers in moving robots have limited space and power, we can improve efficiency by employing the shortest distance technique.

## PROBLEM STATEMENT

## Problem statement – “Space - Efficient Algorithms for Reachability in Directed Geometric Graphs”

## Domain – Graph exploration and implementations

## LITERATURE SURVERY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Authors | Title | Publishing | Techniques |
| 1 | Niranka Banerjee, Sankardeep Chakraborty, Venkatesh Raman | Improved Space Efficient Algorithms for BFS, DFS and Applications | Agronomy forSustainable Development volume(2015) | Explores the time and space complexities of BFS and DFS and experiments on the algorithms to improve the relative space complexity |
| 2 | Sujoy Bhore and Rahul Jain | Space-Efficient Algorithms for Reachability in Geometric Graphs | IEEE(2015) | The paper explains the space related optimizations while maintaining a polynomial time bound. |
| 3 | [WangShu-Xi](https://www.sciencedirect.com/science/article/pii/S1877705812001208" \l "!) | The Improved Dijkstra's Shortest Path Algorithm and Its Application | School of Information Technology, The University of International Business and Economics, Bei Jing, China,(2017) | The shortest path problem exists in variety of areas. A well known shortest path algorithm is Dijkstra's, also called “label algorithm”. |
| 4 | A Fitriansyah, N W Parwati, DR Wardhani andN Kustian | Dijkstra's Algorithm to Find Shortest Path of Tourist Destination in Bali | [Journal of Physics: Conference Series](https://iopscience.iop.org/journal/1742-6596), [Volume 1338](https://iopscience.iop.org/volume/1742-6596/1338), [The 2nd International Conference on Applied Sciences Mathematics and Informatics 9–11 August 2018, Bandar Lampung, Indonesia](https://iopscience.iop.org/issue/1742-6596/1338/1)(2019) | Bali is one of many small island in Indonesia and referred as "*The Island of Gods*". Bali has varieties of tourist destination e.g. art villages, ecotourism, historical building, spiritual tourism and temples. Dijkstra's algorithm is an algorithm that used to be solution in finding shortest path problem. It can be use to find the shortest route between one tourist destination and all other tourist destinations. |

## SOLUTION STRATEGY

## The applications will find the shortest path for robot so it can spend less resources reaching that destination.

## Proposed Technique is using Dijkstras Algorithm.

## It is easy to implement with the graph data structure and the time and space complexity is comparable with other solutions.

## Prerequisites before implementing the algorithm

## Visiting the nodes

## Exploration of nodes

## Depth first search algorithm and functionality

## Time complexity – O(logV)

## Space complexity - O(V) .

## Since the computers in a mobile robot can be limiting the problem statement objective of using the most space saving algorithm is justified.

## FLOWCHART

## Diagram Description automatically generated

## HARDWARE AND SOFTWARE USED

Hardware –

• Processor –

1. AMD Athlon 3000G and above

2. Intel i3 4th gen and above

• Ram – 4Gs

• Storage - 240gs SSD / 512gb HHD

Software –

• Windows 10

• Python 3.8 and above

• visual studio code

Webhosting – flask

**IMPLEMENTATION**

main.py

import **graphlib**

from **flask** import \*

import **sys**

**app** = **Flask**(\_\_name\_\_)

nodes = ["0", "1", "2", "3","4"]

init\_graph = {}

for node in nodes:

    init\_graph[node] = {}

init\_graph["0"]["1"] = 3

init\_graph["0"]["3"] = 7

init\_graph["0"]["4"] = 8

init\_graph["1"]["3"] = 4

init\_graph["1"]["2"] = 1

init\_graph["2"]["3"] = 2

init\_graph["3"]["1"] = 4

init\_graph["3"]["4"] = 3

class **Graph**(**object**):

    def **\_\_init\_\_**(self, nodes, init\_graph):

        self.nodes = nodes

        self.graph = self.**construct\_graph**(nodes, init\_graph)

    def **construct\_graph**(self, nodes, init\_graph):

        graph = {}

        for node in nodes:

            graph[node] = {}

        graph.**update**(init\_graph)

        for node, edges in graph.**items**():

            for adjacent\_node, value in edges.items():

                if graph[adjacent\_node].get(node, False) == False:

                    graph[adjacent\_node][node] = value

        return graph

    def **get\_nodes**(self):

        return self.nodes

    def **get\_outgoing\_edges**(self, node):

        connections = []

        for out\_node in self.nodes:

            if self.graph[node].get(out\_node, False) != False:

               connections.**append**(out\_node)

        return connections

    def **value**(self, node1, node2):

        return self.graph[node1][node2]

def **dijkstra\_algorithm**(graph, start\_node):

    unvisited\_nodes = **list**(graph.get\_nodes())

    shortest\_path = {}

    previous\_nodes = {}

    max\_value = **sys**.maxsize

    for node in unvisited\_nodes:

        shortest\_path[node] = max\_value

    shortest\_path[start\_node] = 0

    while unvisited\_nodes:

        current\_min\_node = None

        for node in unvisited\_nodes:

            if current\_min\_node == None:

                current\_min\_node = node

            elif shortest\_path[node] < shortest\_path[current\_min\_node]:

                current\_min\_node = node

        neighbors = graph.get\_outgoing\_edges(current\_min\_node)

        for neighbor in neighbors:

            tentative\_value = shortest\_path[current\_min\_node] + graph.value(current\_min\_node, neighbor)

            if tentative\_value < shortest\_path[neighbor]:

                shortest\_path[neighbor] = tentative\_value

                previous\_nodes[neighbor] = current\_min\_node

        unvisited\_nodes.**remove**(current\_min\_node)

    return previous\_nodes, shortest\_path

def **print\_weight**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**("Min value {}.".**format**(shortest\_path[target\_node]))

    return "Min value {} ".**format**(shortest\_path[target\_node])

def **print\_node**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**(" -> ".**join**(**reversed**(path)))

    return " -> ".**join**(**reversed**(path))

**@app.route**("/")

def **index**():

     return **render\_template**("index.html");

**@app.route**("/index2")

def **index2**():

     return **render\_template**("index2.html");

**@app.route**("/saveDetails", methods=["POST", "GET"])

def **saveDetails**():

    if request.method == "POST":

        try:

            start = request.form["startingnode"]

            desti = request.form["destinationnode"]

**print**(start)

**print**(desti)

            graph = **Graph**(nodes, init\_graph)

            previous\_nodes, shortest\_path = **dijkstra\_algorithm**(graph=graph, start\_node=start)

            msg = **print\_node**(previous\_nodes, shortest\_path, start\_node=start, target\_node=desti)

            msg1 = **print\_weight**(previous\_nodes, shortest\_path, start\_node=start, target\_node=desti)

        except:

            msg = "counldnt start or nodes dont exist"

        finally:

            return **render\_template**("suc.html", msg=msg,msg1=msg1)

if \_\_name\_\_ == "\_\_main\_\_":

**app**.**run**(debug=True)

Libraries –

import **graphlib**

from **flask** import \*

import **sys**

Graph initialization –

nodes = ["0", "1", "2", "3","4"]

init\_graph = {}

for node in nodes:

    init\_graph[node] = {}

init\_graph["0"]["1"] = 3

init\_graph["0"]["3"] = 7

init\_graph["0"]["4"] = 8

init\_graph["1"]["3"] = 4

init\_graph["1"]["2"] = 1

init\_graph["2"]["3"] = 2

init\_graph["3"]["1"] = 4

init\_graph["3"]["4"] = 3

Dijkstra algorithm –

class **Graph**(**object**):

    def **\_\_init\_\_**(self, nodes, init\_graph):

        self.nodes = nodes

        self.graph = self.**construct\_graph**(nodes, init\_graph)

    def **construct\_graph**(self, nodes, init\_graph):

        graph = {}

        for node in nodes:

            graph[node] = {}

        graph.**update**(init\_graph)

        for node, edges in graph.**items**():

            for adjacent\_node, value in edges.items():

                if graph[adjacent\_node].get(node, False) == False:

                    graph[adjacent\_node][node] = value

        return graph

    def **get\_nodes**(self):

        return self.nodes

    def **get\_outgoing\_edges**(self, node):

        connections = []

        for out\_node in self.nodes:

            if self.graph[node].get(out\_node, False) != False:

                connections.**append**(out\_node)

        return connections

    def **value**(self, node1, node2):

        return self.graph[node1][node2]

def **dijkstra\_algorithm**(graph, start\_node):

    unvisited\_nodes = **list**(graph.get\_nodes())

    shortest\_path = {}

    previous\_nodes = {}

    max\_value = **sys**.maxsize

    for node in unvisited\_nodes:

        shortest\_path[node] = max\_value

    shortest\_path[start\_node] = 0

    while unvisited\_nodes:

        current\_min\_node = None

        for node in unvisited\_nodes:

            if current\_min\_node == None:

                current\_min\_node = node

            elif shortest\_path[node] < shortest\_path[current\_min\_node]:

                current\_min\_node = node

        neighbors = graph.get\_outgoing\_edges(current\_min\_node)

        for neighbor in neighbors:

            tentative\_value = shortest\_path[current\_min\_node] + graph.value(current\_min\_node, neighbor)

            if tentative\_value < shortest\_path[neighbor]:

                shortest\_path[neighbor] = tentative\_value

                previous\_nodes[neighbor] = current\_min\_node

        unvisited\_nodes.**remove**(current\_min\_node)

    return previous\_nodes, shortest\_path

def **print\_weight**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**("Min value {}.".**format**(shortest\_path[target\_node]))

    return "Min value {} ".**format**(shortest\_path[target\_node])

def **print\_node**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**(" -> ".**join**(**reversed**(path)))

    return " -> ".**join**(**reversed**(path))

printing the result functions -

def **print\_weight**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**("Min value {}.".**format**(shortest\_path[target\_node]))

    return "Min value {} ".**format**(shortest\_path[target\_node])

def **print\_node**(previous\_nodes, shortest\_path, start\_node, target\_node):

    path = []

    node = target\_node

    while node != start\_node:

        path.**append**(node)

        node = previous\_nodes[node]

    path.**append**(start\_node)

**print**(" -> ".**join**(**reversed**(path)))

    return " -> ".**join**(**reversed**(path))

Flask links –

**@app.route**("/")

def **index**():

     return **render\_template**("index.html");

**@app.route**("/index2")

def **index2**():

     return **render\_template**("index2.html");

**@app.route**("/saveDetails", methods=["POST", "GET"])

def **saveDetails**():

    if request.method == "POST":

        try:

            start = request.form["startingnode"]

            desti = request.form["destinationnode"]

**print**(start)

**print**(desti)

            graph = **Graph**(nodes, init\_graph)

            previous\_nodes, shortest\_path = **dijkstra\_algorithm**(graph=graph, start\_node=start)

            msg = **print\_node**(previous\_nodes, shortest\_path, start\_node=start, target\_node=desti)

            msg1 = **print\_weight**(previous\_nodes, shortest\_path, start\_node=start, target\_node=desti)

        except:

            msg = "counldnt start or nodes dont exist"

        finally:

            return **render\_template**("suc.html", msg=msg,msg1=msg1)

index.html –

<!DOCTYPE *html*>

<html *lang*="en">

<head>

  <meta *charset*="utf-8">

  <meta *content*="width=device-width, initial-scale=1.0" *name*="viewport">

  <title>Index</title>

  <meta *content*="" *name*="description">

  <meta *content*="" *name*="keywords">

*<!-- Google Fonts -->*

  <link *href*="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i,600,600i,700,700i" *rel*="stylesheet">

*<!-- Vendor CSS Files -->*

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap/css/bootstrap.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap-icons/bootstrap-icons.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/boxicons/css/boxicons.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/glightbox/css/glightbox.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/remixicon/remixicon.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.css')}}" *rel*="stylesheet">

*<!-- Template Main CSS File -->*

  <link *href*="{{ url\_for('static',filename='css/style.css')}}" *rel*="stylesheet">

</head>

<body>

*<!-- ======= Header ======= -->*

  <header *id*="header" *class*="fixed-top">

    <div *class*="container d-flex align-items-center justify-content-between">

      <h1 *class*="logo"><a *href*="index.html">S2D</a></h1>

*<!-- Uncomment below if you prefer to use an image logo -->*

*<!-- <a href="index.html" class="logo"><img src="assets/img/logo.png" alt="" class="img-fluid"></a>-->*

      <nav *id*="navbar" *class*="navbar">

        <ul>

          <li><a *class*="nav-link scrollto active" *href*="/">Home</a></li>

          <li><a *class*="nav-link scrollto" *href*="index2">Start</a></li>

        </ul>

        <i *class*="bi bi-list mobile-nav-toggle"></i>

      </nav>*<!-- .navbar -->*

    </div>

  </header>*<!-- End Header -->*

*<!-- ======= Hero Section ======= -->*

  <section *id*="hero" *class*="d-flex align-items-center">

    <div *class*="container position-relative" *data-aos*="fade-up" *data-aos-delay*="100">

      <div *class*="row justify-content-center">

        <div *class*="col-xl-7 col-lg-9 text-center">

          <h1>Source to destination application for warehouse robot system</h1>

        </div>

      </div>

      <div *class*="text-center">

        <a *href*="index2" *class*="btn-get-started scrollto">Get Started</a>

      </div>

  <div *id*="preloader"></div>

  <a *href*="#" *class*="back-to-top d-flex align-items-center justify-content-center"><i *class*="bi bi-arrow-up-short"></i></a>

*<!-- Vendor JS Files -->*

  <script *src*="{{ url\_for('static',filename='vendor/purecounter/purecounter.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/aos/aos.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/glightbox/js/glightbox.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/isotope-layout/isotope.pkgd.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/php-email-form/validate.js')}}"></script>

*<!-- Template Main JS File -->*

  <script *src*="{{ url\_for('static',filename='js/main.js')}}"></script>

</body>

</html>

Index2.html –

<!DOCTYPE *html*>

<html *lang*="en">

<head>

  <meta *charset*="utf-8">

  <meta *content*="width=device-width, initial-scale=1.0" *name*="viewport">

  <title>Input</title>

  <meta *content*="" *name*="description">

  <meta *content*="" *name*="keywords">

*<!-- Google Fonts -->*

  <link *href*="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i,600,600i,700,700i" *rel*="stylesheet">

*<!-- Vendor CSS Files -->*

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap/css/bootstrap.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap-icons/bootstrap-icons.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/boxicons/css/boxicons.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/glightbox/css/glightbox.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/remixicon/remixicon.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.css')}}" *rel*="stylesheet">

*<!-- Template Main CSS File -->*

  <link *href*="{{ url\_for('static',filename='css/style.css')}}" *rel*="stylesheet">

</head>

<body>

*<!-- ======= Header ======= -->*

  <header *id*="header" *class*="fixed-top">

    <div *class*="container d-flex align-items-center justify-content-between">

        <h1 *class*="logo"><a *href*="index.html">S2D</a></h1>

*<!-- Uncomment below if you prefer to use an image logo -->*

*<!-- <a href="index.html" class="logo"><img src="assets/img/logo.png" alt="" class="img-fluid"></a>-->*

      <nav *id*="navbar" *class*="navbar">

        <ul>

          <li><a *class*="nav-link scrollto active" *href*="/">Home</a></li>

          <li><a *class*="nav-link scrollto" *href*="index2">Start</a></li>

        </ul>

        <i *class*="bi bi-list mobile-nav-toggle"></i>

      </nav>*<!-- .navbar -->*

    </div>

  </header>*<!-- End Header -->*

*<!-- ======= Hero Section ======= -->*

  <section *id*="hero" *class*="d-flex align-items-center">

    <div *class*="container position-middle" *data-aos*="fade-up" *data-aos-delay*="100">

      <div *class*="row justify-content-center">

        <div *class*="col-xl-8 col-lg-7 text-center">

            <form *action* = "./saveDetails" *method*="post">

                <table>

                    <tr><td>Starting Node</td><td><input *type*="text" *name*="startingnode"></td></tr>

                    <tr><td>Destination Node </td><td><input *type*="text" *name*="destinationnode"></td></tr>

                    <tr><td><input *class*="btn-get-started scrollto" *type*="submit" *value*="Submit"></td></tr>

                    <img *src* = "{{ url\_for('static',filename='img/graph.png')}}" *width*=400px align = "right" >

                </table>

                </form>

        </div>

      </div>

  <div *id*="preloader"></div>

  <a *href*="#" *class*="back-to-top d-flex align-items-center justify-content-center"><i *class*="bi bi-arrow-up-short"></i></a>

*<!-- Vendor JS Files -->*

  <script *src*="{{ url\_for('static',filename='vendor/purecounter/purecounter.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/aos/aos.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/glightbox/js/glightbox.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/isotope-layout/isotope.pkgd.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.js')}}"></script>

  <script *src*="{{ url\_for('static',filename='vendor/php-email-form/validate.js')}}"></script>

*<!-- Template Main JS File -->*

  <script *src*="{{ url\_for('static',filename='js/main.js')}}"></script>

</body>

</html>

suc.html

<!DOCTYPE *html*>

<html>

<head>

    <title>suc page</title>

</head>

<body>

</body>

</html>

<!DOCTYPE *html*>

<html *lang*="en">

<head>

  <meta *charset*="utf-8">

  <meta *content*="width=device-width, initial-scale=1.0" *name*="viewport">

  <title>Output</title>

  <meta *content*="" *name*="description">

  <meta *content*="" *name*="keywords">

*<!-- Google Fonts -->*

  <link *href*="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i,600,600i,700,700i" *rel*="stylesheet">

*<!-- Vendor CSS Files -->*

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/aos/aos.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap/css/bootstrap.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/bootstrap-icons/bootstrap-icons.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/boxicons/css/boxicons.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/glightbox/css/glightbox.min.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/remixicon/remixicon.css')}}" *rel*="stylesheet">

  <link *href*="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.css')}}" *rel*="stylesheet">

*<!-- Template Main CSS File -->*

  <link *href*="{{ url\_for('static',filename='css/style.css')}}" *rel*="stylesheet">

</head>

<body>

*<!-- ======= Header ======= -->*

  <header *id*="header" *class*="fixed-top">

    <div *class*="container d-flex align-items-center justify-content-between">

      <h1 *class*="logo"><a *href*="index.html">S2D</a></h1>

*<!-- Uncomment below if you prefer to use an image logo -->*

*<!-- <a href="index.html" class="logo"><img src="assets/img/logo.png" alt="" class="img-fluid"></a>-->*

      <nav *id*="navbar" *class*="navbar">

        <ul>

          <li><a *class*="nav-link scrollto active" *href*="/">Home</a></li>

          <li><a *class*="nav-link scrollto" *href*="index2">Start</a></li>

        </ul>

        <i *class*="bi bi-list mobile-nav-toggle"></i>

      </nav>*<!-- .navbar -->*

    </div>

  </header>*<!-- End Header -->*

*<!-- ======= Hero Section ======= -->*

  <section *id*="hero" *class*="d-flex align-items-center">

    <div *class*="container position-relative" *data-aos*="fade-up" *data-aos-delay*="100">

      <div *class*="row justify-content-center">

        <div *class*="col-xl-7 col-lg-9 text-center">

        <div *type*="allign-center">

            <h3>{{msg}}</h3>

            <h3>{{msg1}}</h3>

            <br>

            <img *src* = "{{ url\_for('static',filename='img/graph.png')}}" *width*=400px align = "center" >

        </div>

        </div>

      </div>

  <div *id*="preloader"></div>

  <a *href*="#" *class*="back-to-top d-flex align-items-center justify-content-center"><i *class*="bi bi-arrow-up-short"></i></a>

  <!-- Vendor JS Files -->

  <script src="{{ url\_for('static',filename='vendor/purecounter/purecounter.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/aos/aos.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/glightbox/js/glightbox.min.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/isotope-layout/isotope.pkgd.min.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/swiper/swiper-bundle.min.js')}}"></script>

  <script src="{{ url\_for('static',filename='vendor/php-email-form/validate.js')}}"></script>

  <!-- Template Main JS File -->

  <script src="{{ url\_for('static',filename='js/main.js')}}"></script>

</body>

</html>

**RESULTS**

**Text

Description automatically generated**

*(terminal)*

**Graphical user interface, text, application, website, Teams

Description automatically generated**

*(Home page)*

**Diagram

Description automatically generated**

*(Start/Get Started page)*

**Chart, radar chart

Description automatically generated**

*(Result)*

**CONCLUSION & FUTURE SCOPE**

* Working algorithm

* Working web interface
* Adding an ability to enter the node by the user will be an advantage to our current method
* Moving the web hosting service to Django or hosting it in a web server for testing will be appropriate representation of the intended real-life use

**REFERENCES**

* **Improved Space Efficient Algorithms for BFS, DFS and Applications –**

https://www.researchgate.net/publication/305455844\_Improved\_Space\_Efficient\_Algorithms\_for\_BFS\_DFS\_and\_Applications

* **Space-Efficient Algorithms for Reachability in Geometric Graphs –**

<https://arxiv.org/abs/2101.05235>

* **The Improved Dijkstra's Shortest Path Algorithm and Its Application –**

https://www.sciencedirect.com/science/article/pii/S1877705812001208

* **Dijkstra's Algorithm to Find Shortest Path of Tourist Destination in Bali –**

<https://iopscience.iop.org/article/10.1088/1742-6596/1338/1/012044>

* **Flask** - <https://flask.palletsprojects.com/en/2.1.x/>
* **Implementing Dijkstra’s Algorithm –**

<https://www.udacity.com/blog/2021/10/implementing-dijkstras-algorithm-in-python.html>

* **Dijkstra’s algorithm -**

<https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm>

* **Dijkstra’s algorithm -**

<https://www.programiz.com/dsa/dijkstra-algorithm>