

# Assignment - Graph

Rabbani Nur Kumoro  
21/472599/PA/20310

*Department of Computer Sciences and Electronics, Universitas Gadjah Mada Building C, 4th Floor North  
Sekip, Bulaksumur Yogyakarta Indonesia 55281*

**Abstract** - A shortest-path algorithm finds a path containing the minimal cost between two vertices in a graph. Dijkstra's Algorithm is known as the shortest path source, through this assignment we will discuss the algorithm and its applications that the algorithm has in the modern-day. Dijkstra Algorithm has been the backbone of every navigation system such as Google Maps, Waze, and Uber.

**Keywords** – Shortest Path, Dijkstra Algorithm, Graph, Data Structures and Algorithm, Maps, Computer Science, Python

## I. INTRODUCTION

Dijkstra's Algorithm is one of the greedy algorithms used to optimize and find the shortest path between nodes in a graph. Dijkstra's algorithm is an effective algorithm proposed and invented by Dutch Scientist Edsger.W. Dijkstra in the year 1956. The algorithm may represent, for example, road networks.

There exist many variants for this algorithm. The original variant of the algorithm found the shortest path between two nodes, whereas the variant fixes a single node as the source and then finds the shortest path to other nodes, but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree. And this is the concept that is implemented by

Google Maps to calculate and show us the shortest path between two points.

The remainder of this assignment paper is laid out as follows. In Section II, we'll discuss what kind of problem we are facing. Section III will give an overview of the problem and how it could be turned into a graph. In Section IV we'll discuss My solution to solve the problem in detail using Pseudocode and Python. Section V will give a conclusion to the assignment.

## II. PROBLEM

Suppose we are on a vacation during the holiday season. We use a car as our main vehicle to reach our destination. Because of the holiday season, there will be a lot of traffic on the road. We utilize Google Maps to find the best route to our destination since they are based on this algorithm. Through the app, we are going to use Dijkstra's Algorithm to find the shortest-path route to reach our destination.

## III. HOW TO DEFINE THE PROBLEM TO BECOME A GRAPH

Below is a graph that shows a map of the existing highway network in Central Java Province. If we want to start our vacation journey from Brebes to Wonosobo and every time we pass through the city it will take longer than passing through a few

### 4.1 Graph Representation

```
# Graph Representation
graph = {'Brebese': ['Tegal', 'Slawi'],
        'Tegal': ['Pemalang', 'Slawi'],
        'Slawi': ['Tegal', 'Purwokerto'],
        'Purwokerto': ['Purbalingga', 'Cilacap', 'Kroya', 'Kebumen'],
        'Cilacap': ['Purwokerto', 'Kroya'],
        'Kroya': ['Cilacap', 'Purwokerto', 'Kebumen'],
        'Kebumen': ['Kroya', 'Purwokerto'],
        'Purworejo': ['Kebumen', 'Magelang'],
        'Magelang': ['Purworejo', 'Wonosobo', 'Temanggung', 'Boyolali'],
        'Purbalingga': ['BanjarNEGARA', 'Purwokerto', 'Pemalang'],
        'BanjarNEGARA': ['Purbalingga', 'Wonosobo'],
        'Pemalang': ['Tegal', 'Pekalongan'],
        'Wonosobo': ['BanjarNEGARA', 'Temanggung', 'Magelang'],
        'Temanggung': ['Salatiga', 'Wonosobo', 'Magelang'],
        'Pekalongan': ['Pemalang', 'Kendal'],
        'Kendal': ['Pekalongan', 'Semarang', 'Temanggung'],
        'Boyolali': ['Klaten', 'Salatiga', 'Solo'],
        'Salatiga': ['Temanggung', 'Semarang', 'Boyolali'],
        'Semarang': ['Kendal', 'Demak', 'Salatiga'],
        'Klaten': ['Boyolali'],
        'Solo': ['Boyolali', 'Purwodadi', 'Sragen', 'Sukoharjo'],
        'Purwodadi': ['Demak', 'Kudus', 'Blora', 'Solo'],
        'Demak': ['Semarang', 'Purwodadi', 'Kudus', 'Solo'],
        'Kudus': ['Demak', 'Rembang', 'Purwodadi'],
        'Rembang': ['Kudus', 'Blora'],
        'Blora': ['Rembang', 'Purwodadi', 'Sragen'],
        'Sragen': ['Blora', 'Solo'],
        'Sukoharjo': ['Wonogori', 'Solo'],
        'Wonogori': ['Sukoharjo']}
```

In the code above, we use a dictionary to create a graph and use a list to store vertices that are neighbors of a vertex. For example, Brebes is connected to Tegal and Slawi, Klaten is only connected to Boyolali, and so on.

In this section we will discuss the Program's Algorithm by using Pseudocode.

Arrays: graph, path, all\_path, new\_path

Strings: start, end

**Fig. 4.2. Pseudocode**

1. function will be filled with parameters.
2. if start city equals to end city then it will return the result to find\_path.
3. if start city are not available in the graph, then it will return None.
4. the program will do a loop to begin.

5. if the first city of the graph is not on the path, then we will find a new\_path by looping from the first function recursively.
6. index y is the amount of data from the new\_path.
7. we will add data from all paths with existing data from new\_path index y.
8. then at the end the function will return all\_path.

### 4.3 Code Implementation

For the implementation, we are going to use Python.

Below are the functions that are used for some features:

#### 1. Function to Define All Paths

```
def find_path(graph, start, end, path=[]):
    path = path + [start]
    if start == end:
        return [path]
    if not start in graph:
        return []
    all_path = []
    for nodes in graph[start]:
        if nodes not in path:
            path_path = find_path(graph, nodes, end, path)
            for new_path in path_path:
                all_path.append(new_path)
    return all_path
```

This function will find and return all paths from start point to end point or destination.

Suppose we want to find all possible paths that can be passed from Semarang to Boyolali, then this function will return all paths in the form of a list.

#### 2. Function to Count the Data

```
def counter(data):
    route = ""
    for y in range(0, len(data)):
        if y < len(data) - 1:
            route += data[y] + " > "
        else:
            route += data[y]
    return route
```

This function will calculate the data from the graph representation paths and it's a dictionary that stores objects as keys and counts as values.

#### 3. Function to Input the Cities

```
Start = input("Enter Starting Point of the City : ")
Finish = input("Enter End Point of the City : ")
```

This function will input the user's starting point and end point of the city.

#### 4. Function to find the amount of routes, shortest-longest path and list of passable routes.

```
data_x = find_path(graph, Start, Finish)
print(f"Number of Routes: {len(data_x)}")
min = data_x[0]
max = []
for x in data_x:
    if len(x) < len(min):
        min = x
    if len(x) > len(max):
        max = x

print(f"Shortest-Path Route : {counter(min)}")
print(f"Longest-Path Route : {counter(max)}")
print(f"List of Passable Routes : ")

for x in range(0, len(data_x)):
    print(f"ROUTE {x+1} : ")
    print(counter(data_x[x]))
```

This function will calculate the data from the graph and user input. It will represent all of the routes.

## 4.4 Results

From the functions that are used to create the program, we can see the result in form of a console program.

### 1. Users can input their desire location.

Enter Starting Point of the City : Brebes

Enter End Point of the City : Wonosobo

### 2. Users can find out their routes throughout the output. The output will display the Number of Routes, Shortest-Path Route, Longest-Path Route, and List of Passables Route.

Number of Routes: 142

Shortest-Path Route : Brebes > Slawi > Purwokerto > Purbalingga > Banjarnegara > Wonosobo

```
Longest-Path Route : Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang >
List of Passable Routes :
ROUTE 1 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 2 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 3 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 4 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 5 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 6 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang > Blora > Sragen > Solo > Boyoli
ROUTE 7 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > Solo > Boyolali > Salatiga > Te
ROUTE 8 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > Solo > Boyolali > Salatiga > Te
ROUTE 9 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > Solo > Boyolali > Salatiga > Te
ROUTE 10 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > Solo > Boyolali > Salatiga > Te
ROUTE 11 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali > Salatiga > Temanggung > Wonos
ROUTE 12 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali > Salatiga > Temanggung > Hagel
ROUTE 13 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali > Salatiga > Temanggung > Hagel
ROUTE 14 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali > Salatiga > Temanggung > Hagel
ROUTE 15 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali > Salatiga > Temanggung > Hagel
ROUTE 16 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purwodadi > Solo > Boyolali > Sal
ROUTE 17 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purwodadi > Solo > Boyolali > Sal
ROUTE 18 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purwodadi > Solo > Boyolali > Sal
ROUTE 19 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purwodadi > Solo > Boyolali > Sal
ROUTE 20 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purwodadi > Solo > Boyolali > Sal
ROUTE 21 :
Brebes > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Sragen > Solo > Boyolali > Salatig
ROUTE 22 :
ROUTE 114 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Salatiga > Temanggung > Hagelang > Purworejo > Kebumen >
ROUTE 115 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Salatiga > Temanggung > Hagelang > Purworejo > Kebumen >
ROUTE 116 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Senarang > Salatiga > Temanggung > Hagelang > Wonosobo
ROUTE 117 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Temanggung > Wonosobo
ROUTE 118 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Temanggung > Hagelang > Purworejo > Kebumen > Kroya > Cilacap > Purn
ROUTE 119 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Temanggung > Hagelang > Purworejo > Kebumen > Kroya > Purwokerto > P
ROUTE 120 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Temanggung > Hagelang > Purworejo > Kebumen > Purwokerto > Purbaling
ROUTE 121 :
Brebes > Slawi > Tegal > Pemalang > Pekalongan > Kendal > Temanggung > Hagelang > Wonosobo
ROUTE 122 :
Brebes > Slawi > Purwokerto > Purbalingga > Banjarnegara > Wonosobo
ROUTE 123 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang >
ROUTE 124 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Kudus > Rembang >
ROUTE 125 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > S
ROUTE 126 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Blora > Sragen > S
ROUTE 127 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali >
ROUTE 128 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Purwodadi > Solo > Boyolali >
ROUTE 129 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purn
ROUTE 130 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purn
ROUTE 131 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Purn
ROUTE 132 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Rembang > Blora > Srag
ROUTE 133 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Purwodadi > Blora > Sr
ROUTE 134 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Purwodadi > Blora > Sr
ROUTE 135 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Purwodadi > Solo > Boy
ROUTE 136 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Kudus > Purwodadi > Solo > Boy
ROUTE 137 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Solo > Boyolali > Salatiga > T
ROUTE 138 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Demak > Solo > Boyolali > Salatiga > T
ROUTE 139 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Salatiga > Temanggung > Wonosobo
ROUTE 140 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Senarang > Salatiga > Temanggung > Hagelang > Won
ROUTE 141 :
Brebes > Slawi > Purwokerto > Purbalingga > Pemalang > Pekalongan > Kendal > Temanggung > Wonosobo
```

## V. CONCLUSION

We can conclude that this program is a success. The program is a “Smart and Simple” navigation system for finding the best route to reach our desired destination.

We managed to find the shortest-path using Dijkstra’s Algorithm for the vacation. Our vacation starts from Brebes, the program showed us the Shortest-Path Route out of 142 possible routes to Wonosobo and that is from Brebes > Slawi > Purwokerto > Purbalingga > Banjarnegara > Wonosobo.

## VI. ACKNOWLEDGEMENTS

I very much appreciate the guidance and advice that Mr. Wahyono, S. Kom., Ph.D., throughout the Data Structures and Algorithm course for this semester. Thank you for all the knowledge that you have given to all of us, students.

I also am aware of the program's drawbacks that it isn’t comparable to other navigation system apps out there that use complex programming languages and many design keys. But throughout the completion of the assignment, I am proud of what I can create and of how it could be used and applied by the users.

## VII. REFERENCES

- [1] Alibi, M., 2019. APLIKASI PENCARIAN RUTE TERCEPAT KENDARAAN BERMOTOR DI KOTA MALANG MENGGUNAKAN ALGORITMA DIJKSTRA DAN MULTIPLE REGRESSION. [online] Etheses.uin-malang.ac.id. Available at: <<http://etheses.uin-malang.ac.id/16569/1/13650098.pdf>> [Accessed 6 May 2022].

- [2] Fadhil Mukhlif, and Abdu Saif, 2020. Comparative Study On Bellman-Ford And Dijkstra Algorithms. International Conference on Communication, Electrical and Computer Networks, p.255f0742. [online] researchgate.net Available at: <[https://www.researchgate.net/publication/340790429\\_Comparative\\_Study\\_On\\_Bellman-Ford\\_And\\_Dijkstra\\_Algorithms](https://www.researchgate.net/publication/340790429_Comparative_Study_On_Bellman-Ford_And_Dijkstra_Algorithms)> [Accessed 5 May 2022].
- [3] Sitompul, E., 2015. Discrete Mathematics Dr.-Ing. Erwin Sitompul - ppt video online download. [online] Slideplayer.com. Available at: <<https://slideplayer.com/slide/2433835/>> [Accessed 7 May 2022].
- [4] Ganga. 2021. The Algorithms Behind The Working Of Google Maps | CodeChef. [online] Available at: <<https://blog.codechef.com/2021/08/30/the-algorithms-behind-the-working-of-google-maps-dijkstras-and-a-star-algorithm/>> [Accessed 5 May 2022].
- [5] YOON MI, K., 2019. Dijkstra Algorithm: Key to Finding the Shortest Path, Google Map to Waze. [online] Medium. Available at: <<https://medium.com/@yk392/dijkstra-a-lgorithm-key-to-finding-the-shortest-path-google-map-to-waze-56ff3d9f92f0>> [Accessed 5 May 2022].
- [6] Nedich, A., 2009. Lecture 18 Solving Shortest Path Problem: Dijkstra's Algorithm. [online] Ifp.illinois.edu. Available at: <[http://www.ifp.illinois.edu/~angelia/ge330fall09\\_dijkstra\\_l18.pdf](http://www.ifp.illinois.edu/~angelia/ge330fall09_dijkstra_l18.pdf)> [Accessed 6 May 2022].
- [7] Singhal, A., 2016. Dijkstra Algorithm | Example | Time Complexity | Gate Vidyalay. [online] Gatevidyalay.com. Available at: <<https://www.gatevidyalay.com/dijkstras-algorithm-shortest-path-algorithm/>> [Accessed 6 May 2022].