ISSN 2090-3359 (Print)
ISSN 2090-3367 (Online)

ΑΔΣ

Advances in Decision Sciences

Volume 23
Issue 3
September 2019

Published by Asia University, Taiwan

ADS@ASIAUNIVERSITY

Graph Theory and Environmental Algorithmic Solutions to Assign Vehicles: Application to Garbage Collection in Vietnam*

Buu-Chau Truong

Faculty of Mathematics and Statistics, Ton Duc Thang University

Ho Chi Minh City, Vietnam

Kim-Hung Pho

Fractional Calculus, Optimization and Algebra Research Group

Faculty of Mathematics and Statistics, Ton Duc Thang University

Ho Chi Minh City, Vietnam

Van-Buol Nguyen

General Faculty, Binh Duong Economics & Technology University

Binh Duong City, Vietnam

Bui Anh Tuan

Department of Mathematics Education, Teachers College, Can Tho University, Vietnam.

Wing-Keung Wong**

Department of Finance, Fintech Center, Big Data Research Center, Asia University, Taiwan

and

Department of Medical Research, China Medical University Hospital, Taiwan

and

Department of Economics and Finance, the Hang Seng University of Hong Kong, China

Revised: July 2019

* The authors wish to thank a reviewer for very helpful comments and suggestions. The fifth author would like to thank Robert B. Miller and Howard E. Thompson for their continuous encouragement. Grants from Ton Duc Thang University, Binh Duong Economics & Technology University, Can Tho University, Asia University, China Medical University Hospital, Hang Seng University of Hong Kong, Research Grants Council of Hong Kong, and Ministry of Science and Technology (MOST), Taiwan are acknowledged.

** Corresponding author: wong@asia.edu.tw

Abstract

The problem of finding the shortest path including garbage collection is one of the most important problems in environmental research and public health. Usually, the road map has been modeled by a connected undirected graph with the edge representing the path, the weight being the length of the road, and the vertex being the intersection of edges. Hence, the initial problem becomes a problem finding the shortest path on the simulated graph. Although the shortest path problem has been extensively researched and widely applied in miscellaneous disciplines all over the world and for many years, as far as we know, there is no study to apply graph theory to solve the shortest path problem and provide solution to the problem of "assigning vehicles to collect garbage" in Vietnam. Thus, to bridge the gap in the literature of environmental research and public health. We utilize three algorithms including Fleury, Floyd, and Greedy algorithms to analyze to the problem of "assigning vehicles to collect garbage" in District 5, Ho Chi Minh City, Vietnam. We then apply the approach to draw the road guide for the vehicle to run in District 5 of Ho Chi Minh city. To do so, we first draw a small part of the map and then draw the entire road map of District 5 in Ho Chi Minh city. The approach recommended in our paper is reliable and useful for managers in environmental research and public health to use our approach to get the optimal cost and travelling time.

Keywords: Fleury algorithm, Floyd algorithm, Greedy algorithm, shortest path

JEL: A11, G02, G30, O35

1 Introduction

The concept of graph theory has been developed since the seventeen century by the famous Mathematician Leonhard Euler (Euler, 1736) to give a solution to the problem of finding a way to cross the seven bridges in Konigsberg city. Afterward, the usage of graph theory has been widely used in many different areas and the theory has been helping many academics and practitioners to solve many well-known problems in the history. Finding the shortest path is one of the classic problems by using graph theory to simulate and conduct algorithms to obtain solution effectively and comprehensively. To date, academics have developed some good algorithms to get better optimal solutions to solve the problem.

There are several applications by using graph theory, for example, automatic path guidance, computer network signal transmission, global positioning signal (GPS) path, etc. Finding the shortest path is one of the most classic problems by using graph theory. The shortest path cycle through all the edges on the connected graph is known as the Euler cycle (Euler, 1736). The theory has been extended and applied recently.

For instance, Lawler (1972) presents the procedure to computing the k best solutions to discrete optimization problems with its application to the shortest path problem. Handler and Zang (1980) provide to the dual algorithm for the constrained shortest path problem. Ahuja *et al.* (1990) introduce to the faster algorithms for the shortest path problem. Hassin (1992) presents approximated schemes for the restricted shortest path problem. Montemanni and Gambardella (2004) introduce the exact algorithm for the robust shortest path problem with interval data. In addition, Agafonov and Myasnikov (2016) present a method to get reliable shortest path search in time-dependent stochastic networks with application in GIS-based traffic control, etc.

Furthermore, there are numerous works studying the problem of getting the shortest path. For example, Feillet *et al.* (2004) provide an exact algorithm to solve the problem of getting the elementary shortest path with resource constraints, especially on the application of vehicle routing problems. Garaix *et al.* (2010) present to solve the vehicle routing problems with alternative paths with application on on-demand transportation. Chassein and Goerigk (2015) introduce a new bound to get the midpoint solution in minmax regret optimization with an application to the robust shortest path problem. Zeng *et al.* (2017) recommend to use the heuristic k-shortest path algorithm to determine the most eco-friendly path with a travel time constraint with application on the support vector machine. Aly and Cleemput

(2017) propose to use the improved protocol to securely solve the shortest path problem and apply the approach to combinatorial auctions. There are many other works studying the problem of getting the shortest path. Readers may refer to, for example, Deng *et al.* (2012), Lozano *et al.* (2013), Zhang *et al.* (2013), Mullai *et al.* (2017), Marinakis *et al.* (2017), Broumi *et al.* (2018), and Kumar *et al.* (2018) for more information.

The waste collection is also a very important issue in environmental research and public health. For example, Vimercati *et al.* (2016) study respiratory health in waste collection and disposal workers. Cao, *et al.* (2018) study the relationships between the characteristics of the village population structure and rural residential solid waste collection services and obtain evidence from China. Liang and Liu (2018) present a network design for municipal solid waste collection with application on the Nanjing Jiangbei area. Banyai *et al.* (2019) introduce the optimization of municipal waste collection routing with impact of industry 4.0 technologies on environmental awareness and sustainability, etc.

The problem of finding the shortest path including garbage collection is one of the most important problems in environmental research and public health. It is well known that garbage collection is one of the most urgent tasks for every country in the world because if we do not handle garbage collection well and thoroughly, it will cause environmental pollution, it will greatly affect everyone in the city or even in the entire world. In this connection, every country in the world takes this issue very seriously, and thus, it is important to study the problem of assigning vehicles to collect garbage.

Although the shortest path problem has been extensively researched and widely applied in miscellaneous disciplines all over the world and for many years, as far as we know, there is no study to apply graph theory to solve the shortest path problem and provide solution to the problem of "assigning vehicles to collect garbage" in Vietnam. Thus, to bridge the gap in the literature. We utilize three algorithms including Fleury, Floyd, and Greedy algorithms to analyze to the problem of "assigning vehicles to collect garbage" in District 5, Ho Chi Minh City, Vietnam. We then apply the approach to draw the road guide for the vehicle to run in District 5 of Ho Chi Minh city. To do so, we first draw a small part of the map and then draw the entire road map of District 5 in Ho Chi Minh city.

The approach recommended in our paper is reliable and useful for managers to use our approach to get the optimal (it is minimal in this case) cost and travelling time. If managers do not use our approach, their travel cost and travelling time will not be optimal and the managers could pay higher price for travelling and spend more time in travelling. In this paper, we only apply the approach to solve the problem to obtain the shortest path for District 5, Ho Chi Minh city, Vietnam. The algorithms recommend in this article can be applied to every place in the world. This is the profound contribution of our paper.

The rest of the paper is structured as follows. In Section 2, we will discuss all definitions and notations being used in our paper. The methodology will be introduced in Section 3. In Section 4, we utilize three algorithms including Fleury, Floyd, and Greedy algorithms to analyze to the problem of "assigning vehicles to collect garbage" in District 5, Ho Chi Minh City, Vietnam. The last section gives some concluding remarks and inferences in our paper.

2 Definitions and Notations

In this section, we will discuss all definitions and notations being used in our paper.

2.1 Graph

Graph theory has been developed for long with good applications. With the acid of strong development in both electronic computers and informatics, the theory has developed rapidly in the last century and becomes more interesting. Applications of graph theory include traffic maps of different cities, organizational charts for agencies, computer network and neural network. In general, graph is defined as follows: Graph (G) is a discrete structure $G = (V, E)$ consisting of vertices and edges connecting the vertices, where V and E are sets of vertices and edges, respectively, in which E could be a pair (u, v) where u and v are two vertices of V. Figures 1 and 2 illustrate two different forms of graphs in practice.

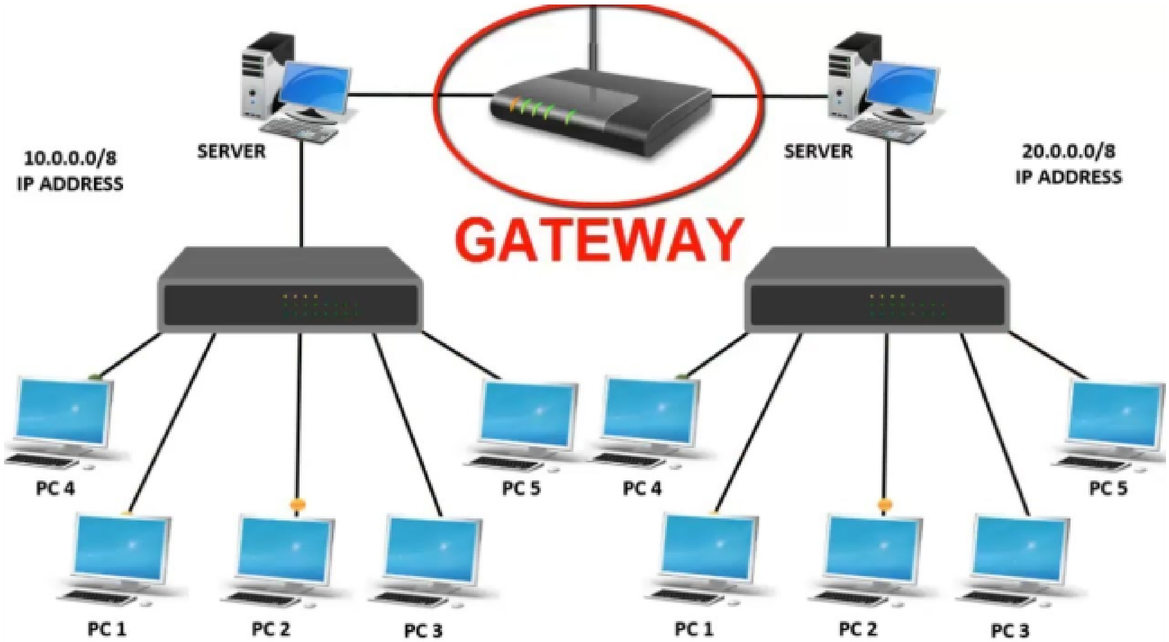

Figure 1: Computer network

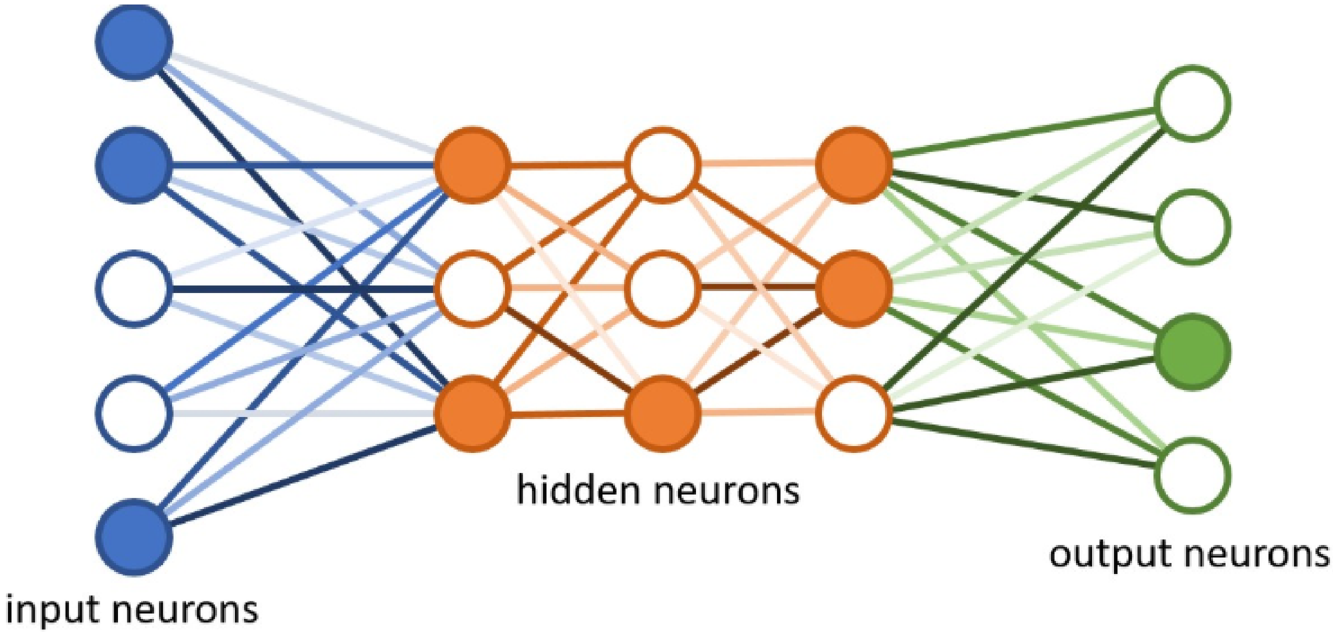

Figure 2: Neural network

2.1.1 Undirected graph and directed graph

Graph can be classified into two categories: undirected graph and directed graph. An undirected graph is a graph that contains only undirected edges (regardless of direction), while a directed graph is a graph that contains directed edges. Obviously, replacing each undirected edge with two corresponding directions, each undirected graph can be represented by a directed graph.

In addition, graph can also be classified as another two distinguish categories: single graph and multi graph. Single graph is a graph in which each pair of vertices is connected by not more than one edge (which can also be treated as graph). On the other hand, multi-graph is a graph whose vertex pairs are connected with more than one edge.

2.1.2 Degree of graph

The degree of vertex $v \in V$, denoted by $deg(v)$, is the total number of edges associated with. Furthermore, one also divide it into two categories: isolated vertex and leaf vertex. A vertex with degree 0 is called an isolated vertex. A vertex with degree 1 is called a leaf vertex or end vertex.

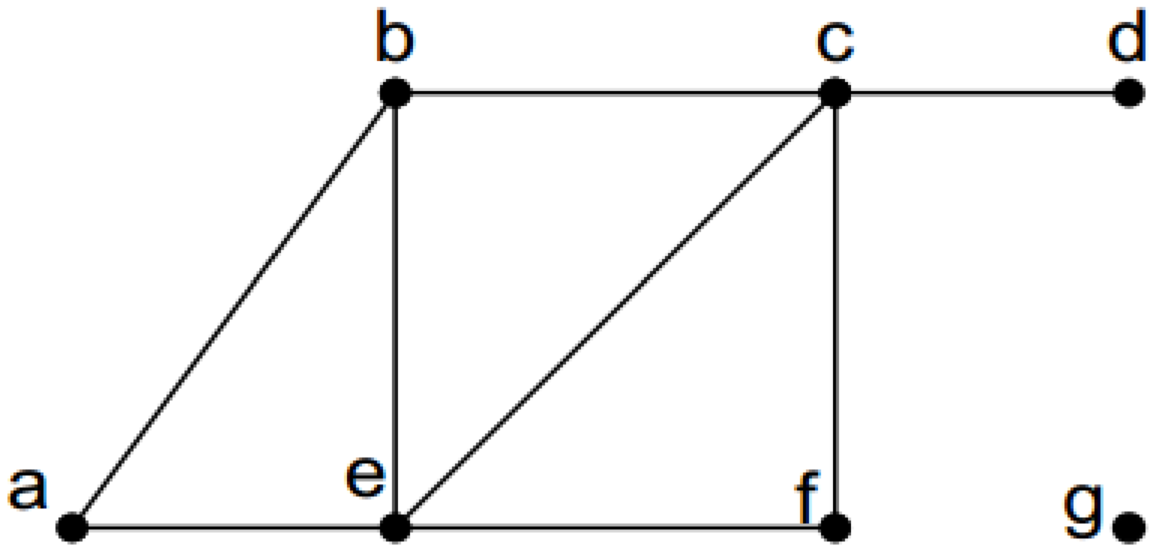

Figure 3: undirected graph G

Considering the graph G displayed in Figure 3 with the set of vertices $V = \{a, b, c, d, e, f, g\}$ and the set of edges $E = \{(a, b), (a, e), (b, c), (b, e), (c, e), (c, d), (c, f)\}$, the degree of vertexes are $deg(a) = deg(f) = 2, deg(b) = 3, deg(c) = deg(e) = 4, deg(d) = 1, deg(g) = 0$. It can be seen that vertex g is an isolated vertex and vertex d is a leaf vertex.

2.1.3 Graph Representation

In order to store graphs and perform various algorithms properly, we have to present graphs on computers nicely, and use appropriate data structures to describe graphs. Choosing which data structure to present graphs has a great impact in the algorithmic efficiency. Therefore, selecting the appropriate data structure to present the graph will depend on each specific problem. One of the most ubiquitous ways to present graphs is to use incidence matrix or adjaceny matrix (Harary, 1962). We describe the approach in the following.

Suppose that $G = (V, E)$ is a single graph with n number of vertices (symbol $|V|$). Without losing generality, the vertices can be numbered as $1, 2, ..., n$. Under this setting, we can present the graph by using the following square matrix $A = [a[i, j]]$ with dimension n:

$$a[i,j] = \begin{cases} 1 & \text{for any } \ (i,j) \in E \ , \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

For any i, we set $a[i, i] = 0$ in (1).

For multi-plots graph, the representation is similar. We note that if (i, j) is the edge, then, instead of wring "1" as what is done in the single graph as shown in (1), we write the number of edges connected between the vertex i and vertex j in the cell of $[i, j]$ as shown in the following:

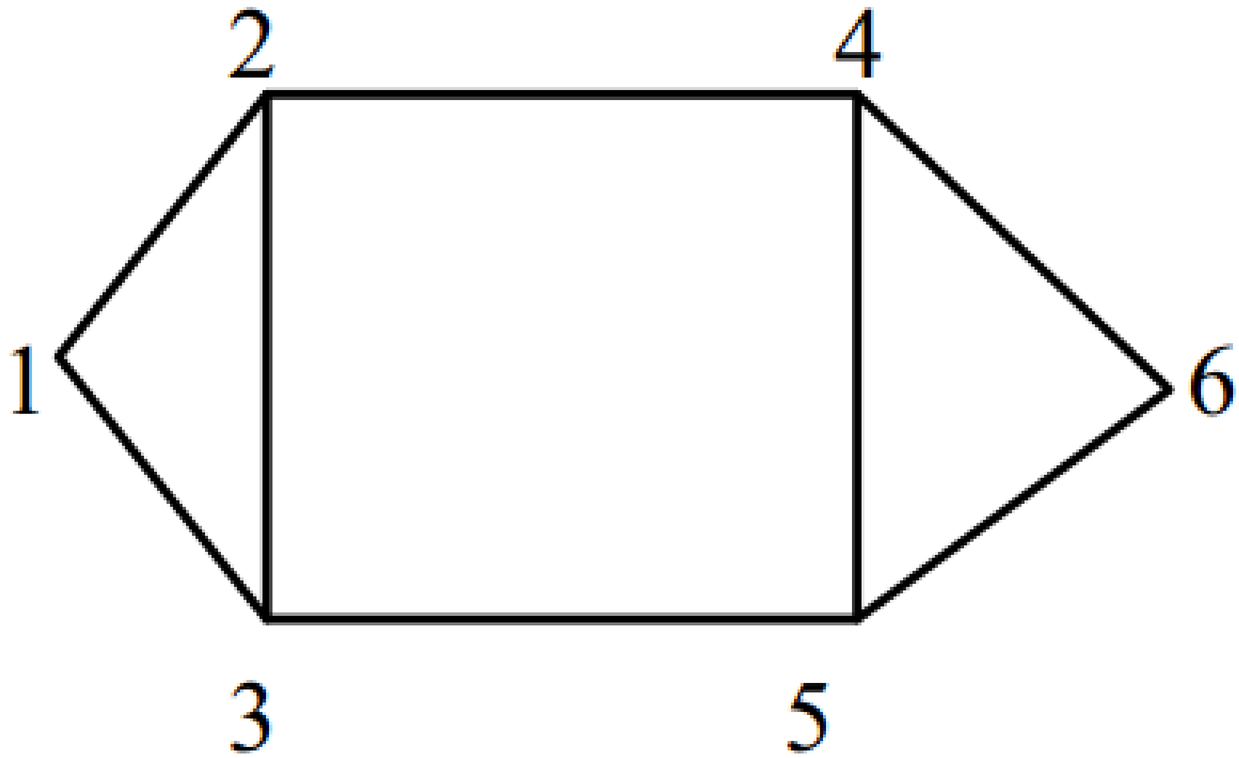

Figure 4: Undirected graph unweighted G

Considering the graph G is provided in Figure 4, we perform the undirected graph unweighted by using matrix A as follows:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

2.2 Path, Cycles, Conjunctions on Graphs

Let the sequence of the path of length k from vertex u to vertex v on scalar graph $G =< V, E >$ to be

$$x_0, x_1, \cdots, x_{k-1}, x_k \ ,$$

where k is a positive integer, $x_0 = u$, $x_k = v$, and $(x_i, x_{i+1}) \in E$ for $i = 0, 1, 2, ..., k-1$. Then, the **path** can be presented as the following series of edges:

$$(x_0, x_1), (x_1, x_2), \cdots, (x_{k-1}, x_k).$$

Let vertex u is the top vertex and vertex v is the end vertex of the path, then **cycle** is the path with the top vertex coinciding with the last vertex (u = v). **Single path** and **single cycle** are the corresponding path and cycle, respectively, in which no edge is repeated.

2.3 Euler Cycle, Euler Path and Euler Graph

Giving an undirected graph $G = (V, E)$, the **Euler cycle** is a cycle that goes through every edge and every vertex of a graph; however, each side does not go more than once. The **Euler path** is the path that goes through every edge and every vertex of the graph; however, each side does not go more than once.

On the other hand, for any directed graph $G = (V, E)$, the **directed Euler cycle** is the cycle that goes through every edge and every vertex; however, each edge does not go more than once. The **directed Euler path** is the path that goes through every edge and every vertex; however, each edge does not go more than once. The graph that contains the Euler cycle is called the **Euler graph**. We need to review the following two most crucial theorems before we discussed the theory.

3 Methodology

In this section, we introduce to three algorithms: Fleury, Floyd, and Greedy algorithms that will be used in this paper. In addition, we provide steps to solve the shortest path problem. We first present to the Fleury algorithm.

3.1 Fleury algorithm

The **Fleury algorithm** can be used to find the Euler cycle. Readers may refer in Eiselt et al. (1995) for more information. We now describe the procedure to get the Fleury algorithm. To do so, we first need the input and output as follows:

Input: Graph $G \neq \emptyset$, no isolated vertices.

Output: Euler C cycle of G, or conclusion G has no Euler cycle.

We now ready to describe the procedure to get the Fleury algorithm as follows:

Procedure 1

Step 1: Select any starting vertex v_0, set $v_1 := v_0$, $C := (v_0)$, and $H := G$.

Step 2: If $H = \emptyset$, then C is concluded to be the Euler cycle, and end the procedure; otherwise, go to Step 3.

Step 3: Select the next edge:

If vertex v_1 is a hanging vertex and only vertex v_2 and adjacency v_1 exist, then select edge (v_1, v_2) and go to Step 4.

If vertex v_1 is not a hanging vertex and if every edge associated with v_1 is a bridge, then there is no Euler cycle and end the procedure.

Conversely, select edge (v_1, v_2) which is not a bridge in H, add the path C on vertex v_2, and go to Step 4.

Step 4: Delete the edge just passed, and delete the isolated vertex:

Remove from H edge (v_1, v_2). If H has an isolated peak, then remove it H, set $v_1 := v_2$, and go to Step 2.

3.2 Floyd algorithm

The Floyd algorithm first introduced by Robert Floyd in 1962 (Floyd, 1962) is used to solve all the problems of finding the shortest distance between any pair of vertices in a given edge weighted directed graph. Now, we briefly describe the algorithm. To do so, we first need the input and output as follows:

Input: The connected graph $G = (V, E)$ with $V = \{1, 2, ..., n\}$ has weight $w(i, j)$ for all sectors (i, j).

Output: The matrix is $D = [d(i, j)]$ where $d(i, j)$ is the shortest path length from i to j for all pairs (i, j). To help readers easily access the algorithm, we describe the procedure as follows:

Procedure 2

Step 1: This is the initialization step in which the symbol D_0 is a starting matrix such that $D_0 = [d_0(i, j)]$ with $d_0(i, j) = w(i, j)$ if there exists an arc (i, j) and $d_0(i, j) = +\infty$ if there is no arc (i, j). Setting $k := 0$.

Step 2: If $k = n$, then finish and in this situation $D = D_n$ is the matrix with the shortest path length; otherwise, increase k by 1 unit $(k := k + 1)$ and go to Step 3 below.

Step 3: Calculate the matrix D_k according to D_{k-1}. For every pair (i, j) with $i = 1, \cdots, n$ and $j = 1, \cdots, n$ we perform the following:
If $d_{k-1}(i, j) > d_{k-1}(i, k) + d_{k-1}(k, j)$ then we let $d_k(i, j) := d_{k-1}(i, k) + d_{k-1}(k, j)$.
Conversely, we let $d_k(i, j) := d_{k-1}(i, j)$.
Return to Step (2).

3.3 Greedy algorithm

The Greedy algorithm first introduced by Edmonds (1971) is an algorithmic paradigm that obtain the solution step by step, by choosing the next step that offers the most obvious and immediate benefit. So, choosing local optimal solution in each step leads to obtain the global optimal solution is best fit for Greedy's approach. At each selected step, the algorithm will "select the best result" defined by the function "select the best value" (it could be the max or min value). If the result is accepted, it will become the solution of the problem; otherwise, the solution will be eliminated. Now, we briefly describe the algorithm. To do so, we first need the input and output as follows:

Input: Matrix A.

Output: Set the x value from set S to be found.

We now ready to describe the procedure to obtain the Greedy algorithm as follows:

Procedure 3

Step 1: Select S from A.

The property "greedy" of the algorithm is oriented by the function "Selection".

Step 2: Initialization: $S = \emptyset$

While $A \neq \emptyset$

Select the best element of A to assign to $x : x = Select(A)$

Step 3: Update objects to choose: $A = A - \{x\}$

If $S \cup \{x\}$ satisfies the requirement of the problem, then

Update solution: $S = S \cup \{x\}$.

3.4 Solving the shortest path problem

Now, we turn to discuss how to use all the above algorithms to solve the shortest path problem by using the following steps:

Step 1: Find all vertices with odd degree based on the input graph matrix.

For each vertex having odd degree, find the shortest path between every pair of vertices. In this step, the Floyd algorithm will be applied to find the shortest path between every pair of vertices on the graph.

Step 2: From the odd-degree vertices found in Step 1, redraw the new graph as the full graph (each vertex connects to all remaining vertices). The weight of each edge on the full graph is the shortest path value found in Step 1.

Step 3: Find the maximal pair with minimum weight on the full graph using by Greedy algorithm. Add the found edges to the original matrix by using the path found the Floyd algorithm. Change the graph to a satisfactory form with all vertices that have even degrees.

Step 4: Use the Fleury algorithm to find the Euler cycle on this new graph and output the result.

We turn to use the approaches discussed in the above to solve the real problem in Vietnam.

4 Drawing the road guide for the vehicle to run in District 5 of Ho Chi Minh city

Ho Chi Minh City is the largest city in Vietnam, one of Vietnam's most important economic, political, cultural and educational centers, and the largest commercial center for Chinese in Vietnam while District 5 is an urban district under Ho Chi Minh City. Thus, studying the problem "Assigning vehicles to collect garbage" in District 5, Ho Chi Minh City, Vietnam is a very important issue in Vietnam.

Suppose that manager in District 5 need to assign a vehicle to collect garbage along the main road of the district. The waste is collected by individual garbage truck that collects the waste from the alley to the main road. Every morning the garbage truck comes from the Agency, goes through the road to collect garbage and then returns to the Agency to finish the day's work by the end of the afternoon. The requirement of the problem requires the vehicle to go through the road and return to the agency. So in order to save travel cost, the problem requires drawing the road guide for the vehicle to run to obtain the most minimal cost. The map of District 5 in Ho Chi Minh city, Vietnam is illustrated as in Figure 5.

The map abstracted by a scalar interconnection matrix represents the following paths: The vertices are intersections and edges are roads with a known length (actual length is taken from www.diadiem.com). The graph of modeling map of District 5 in Ho Chi Minh city, Vietnam with no the weight and the weight is provided in Figures 6 and 7, respectively.

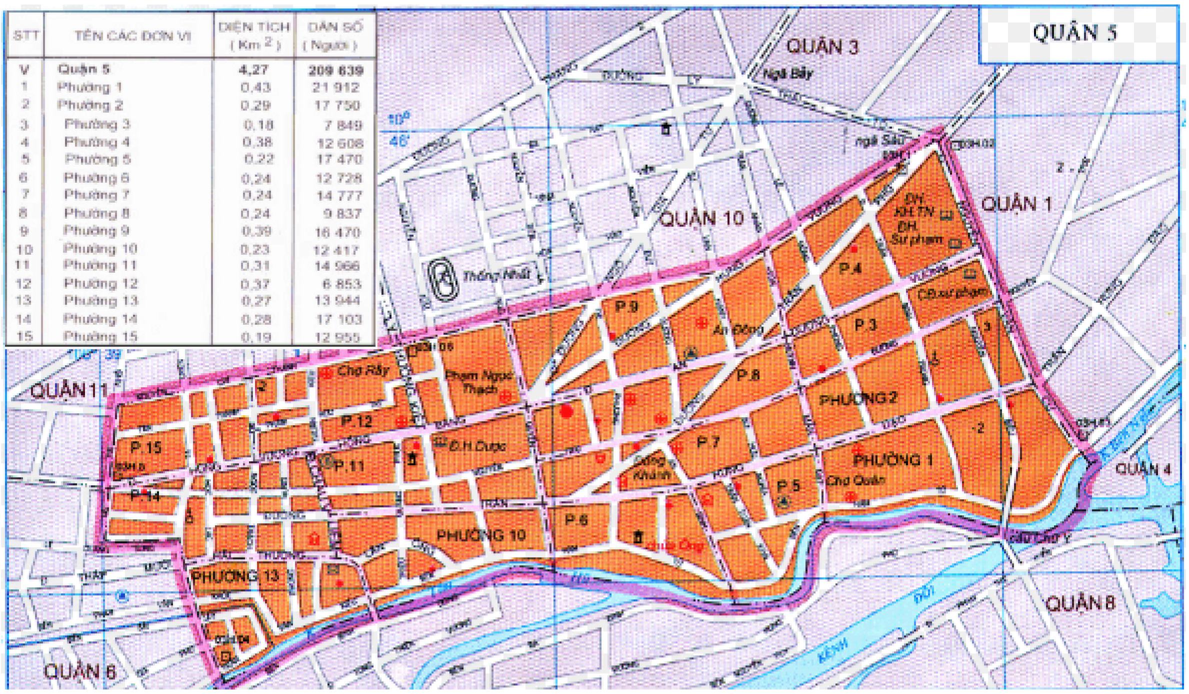

Figure 5: Map of District 5 in Ho Chi Minh city, Vietnam

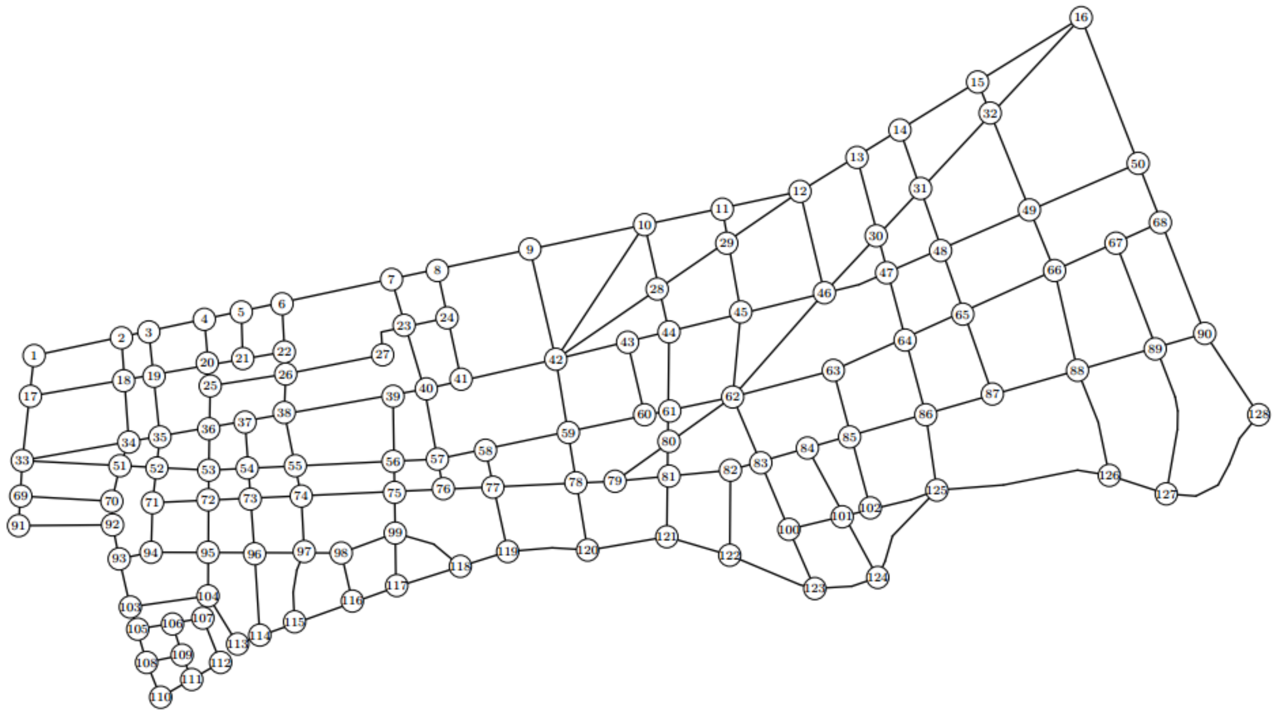

Figure 6: Graph of modeling map of District 5 in Ho Chi Minh city, Vietnam

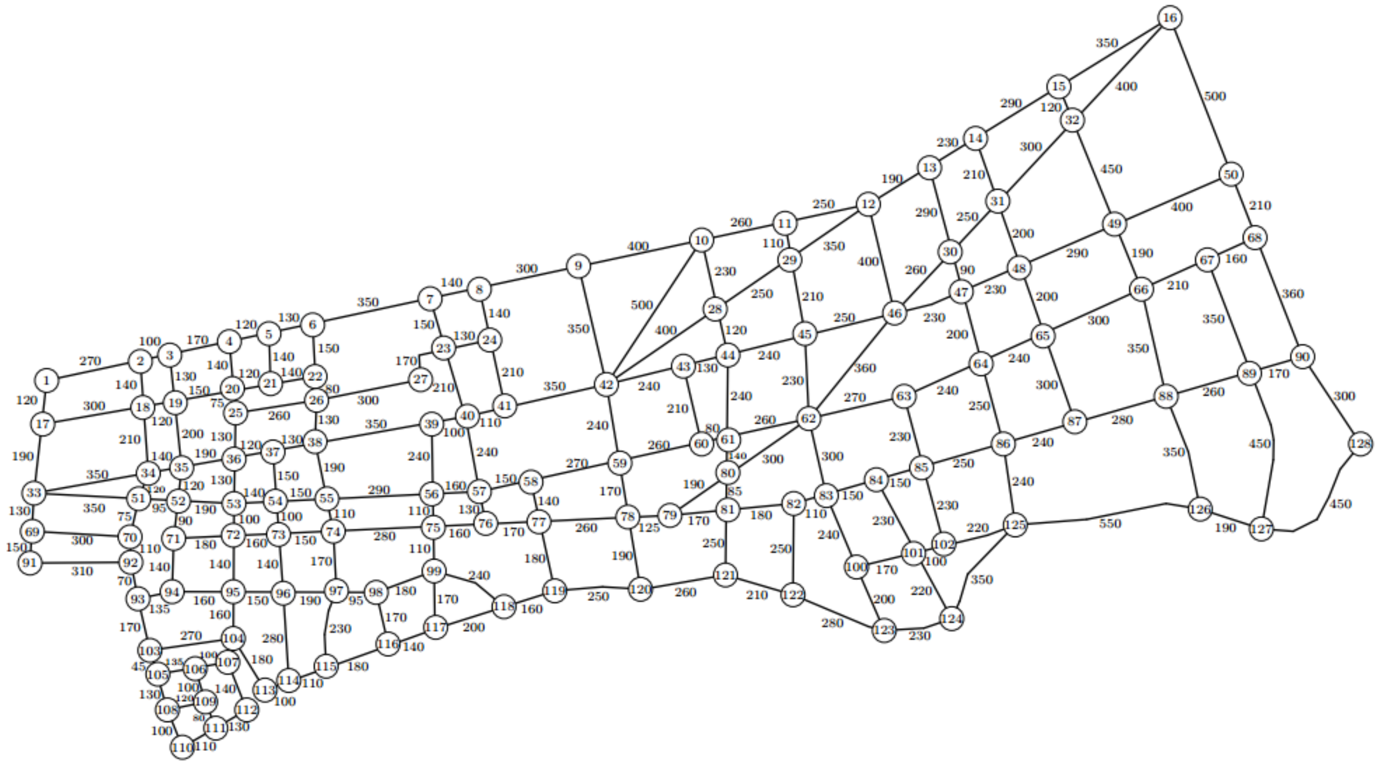

Figure 7: Graph of modeling map of District 5 in Ho Chi Minh city, Vietnam with the weight

In this subsection, we investigate a small part of map of District 5. Taking a part of District 5 map with 13 vertices, we model it with a graph with 13 vertices so that we can find the shortest path.

4.1 Drawing a small part of the map

Taking a part of District 5 map with 13 vertices, we model it into a graph with 13 vertices to find the shortest path.

It can be seen that, to address the shortest path problem, one needs to do through the following 4 steps: First, in Step 1, from the initial graph, we find the vertices with odd degrees. Thereafter, we apply the Floyd algorithm to find the shortest path between all these vertices. The result of Step 1 is provided in Figure 8.

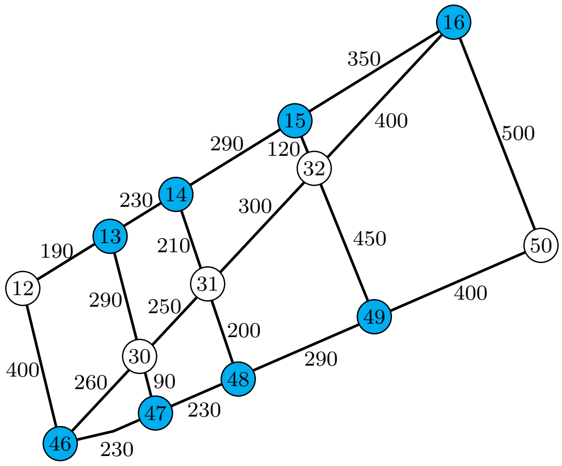

Figure 8: Outcome from Step 1

From Figure 8, we find that the picture can use the following matrix to represent:

$$A_1 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 190 & \infty & \infty & \infty & \infty & \infty & \infty & 400 & \infty & \infty & \infty & \infty \\ 190 & 0 & 230 & \infty & \infty & 290 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & 230 & 0 & 290 & \infty & \infty & 210 & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 290 & 0 & 350 & \infty & \infty & 120 & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 350 & 0 & \infty & \infty & 400 & \infty & \infty & \infty & \infty & 500 \\ \infty & 290 & \infty & \infty & \infty & 0 & 250 & \infty & 260 & 90 & \infty & \infty & \infty \\ \infty & \infty & 210 & \infty & \infty & 250 & 0 & 300 & \infty & \infty & 200 & \infty & \infty \\ \infty & \infty & \infty & 120 & 400 & \infty & 300 & 0 & \infty & \infty & \infty & 350 & \infty \\ 400 & \infty & \infty & \infty & \infty & 260 & \infty & \infty & 0 & 230 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 90 & \infty & \infty & 230 & 0 & 230 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 200 & \infty & \infty & 230 & 0 & 290 & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & 350 & \infty & \infty & 290 & 0 & 400 \\ \infty & \infty & \infty & \infty & 500 & \infty & \infty & \infty & \infty & \infty & \infty & 400 & 0 \end{bmatrix}$$

It can be observed from A_1 matrix that the figure has 13 vertices, but only 8 vertices have odd degrees. The matrix of vertices with odd degrees is illustrated in A_2 matrix as follows:

$$A_2 = \begin{bmatrix} 2 & 3 & 4 & 5 & 9 & 10 & 11 & 12 \\ 0 & 230 & \infty & \infty & \infty & \infty & \infty & \infty \\ 230 & 0 & 290 & \infty & \infty & \infty & \infty & \infty \\ \infty & 290 & 0 & 350 & \infty & \infty & \infty & \infty \\ \infty & \infty & 350 & 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 0 & 230 & \infty & \infty \\ \infty & \infty & \infty & \infty & 230 & 0 & 230 & \infty \\ \infty & \infty & \infty & \infty & \infty & 230 & 0 & 290 \\ \infty & \infty & \infty & \infty & \infty & \infty & 290 & 0 \end{bmatrix}$$

We next find the shortest path matrix between the 13 vertices according to the Floyd algorithm in which the shortest path matrix between vertices have odd degrees is described in

A_3 matrix as follows:

$$A_3 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 190 & 420 & 710 & 1060 & 480 & 630 & 830 & 400 & 570 & 800 & 1090 & 1490 \\ 190 & 0 & 230 & 520 & 870 & 290 & 440 & 640 & 550 & 380 & 610 & 900 & 1300 \\ 420 & 230 & 0 & 290 & 640 & 460 & 210 & 410 & 720 & 550 & 410 & 700 & 1100 \\ 710 & 520 & 290 & 0 & 350 & 670 & 420 & 120 & 930 & 760 & 620 & 470 & 850 \\ 1060 & 870 & 640 & 350 & 0 & 950 & 700 & 400 & 1210 & 1040 & 900 & 750 & 500 \\ 480 & 290 & 460 & 670 & 950 & 0 & 250 & 550 & 260 & 90 & 320 & 610 & 1010 \\ 630 & 440 & 210 & 420 & 700 & 250 & 0 & 300 & 510 & 340 & 200 & 490 & 890 \\ 830 & 640 & 410 & 120 & 400 & 550 & 300 & 0 & 810 & 640 & 500 & 350 & 750 \\ 400 & 550 & 720 & 930 & 1210 & 260 & 510 & 810 & 0 & 230 & 460 & 750 & 1150 \\ 570 & 380 & 550 & 760 & 1040 & 90 & 340 & 640 & 230 & 0 & 230 & 520 & 920 \\ 800 & 610 & 410 & 620 & 900 & 320 & 200 & 500 & 460 & 230 & 0 & 290 & 690 \\ 1090 & 900 & 700 & 470 & 750 & 610 & 490 & 350 & 750 & 520 & 290 & 0 & 400 \\ 1490 & 1300 & 1100 & 850 & 500 & 1010 & 890 & 750 & 1150 & 920 & 690 & 400 & 0 \end{bmatrix}$$

We then go to Step 2 to redraw the full graph with the weights found in Step 1. The result of Step 2 can be presented in Figure 9.

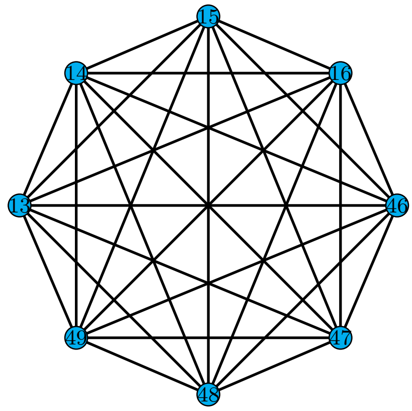

Figure 9: Outcome from Step 2

Using A_3 matrix, we find the shortest path length of the path between odd degree vertices in which the matrix with the shortest length between vertices has odd degrees is provided in A_4 matrix as follows:

$$A_4 = \begin{bmatrix} 2 & 3 & 4 & 5 & 9 & 10 & 11 & 12 \\ 0 & 230 & 520 & 870 & 550 & 380 & 610 & 900 \\ 230 & 0 & 290 & 640 & 720 & 550 & 410 & 700 \\ 520 & 290 & 0 & 350 & 930 & 760 & 620 & 470 \\ 870 & 640 & 350 & 0 & 1210 & 1040 & 900 & 750 \\ 550 & 720 & 930 & 1210 & 0 & 230 & 460 & 750 \\ 380 & 550 & 760 & 1040 & 230 & 0 & 230 & 520 \\ 610 & 410 & 620 & 900 & 460 & 230 & 0 & 290 \\ 900 & 700 & 470 & 750 & 750 & 520 & 290 & 0 \end{bmatrix}$$

We turn to carry out Step 3 to find pairs (there are 5 pairs) with the smallest total weight and exhibit the result of Step 3 in Figure 10.

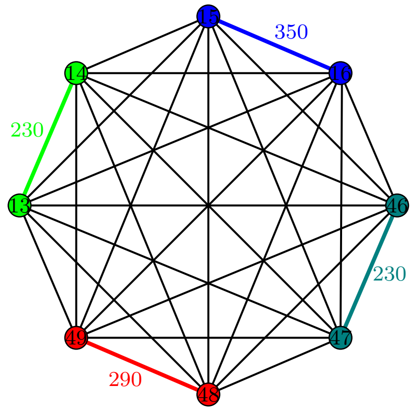

Figure 10: Outcome from Step 3

The pair matrix has the smallest total length (4 pairs). The optimal pair matrix has the smallest weight is presented in Table 1.

The even-degree matrices are obtained after adding pairs illustrated in the following A_5 matrix:

$$A_5 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Finally, we use Step 4 to draw additional edges in the original graph to obtain vertices with even degrees and then use the Fleury algorithm to find the Euler cycle. The result of Step 4 is described in Figures 11 and 12.

Table 1: The optimal pair matrix has the smallest weight

Number of pairs: 4		
Total of the length: 1100		
The beginning vertex	The ending vertex	The length
2	3	230
9	10	230
11	12	290
4	5	350

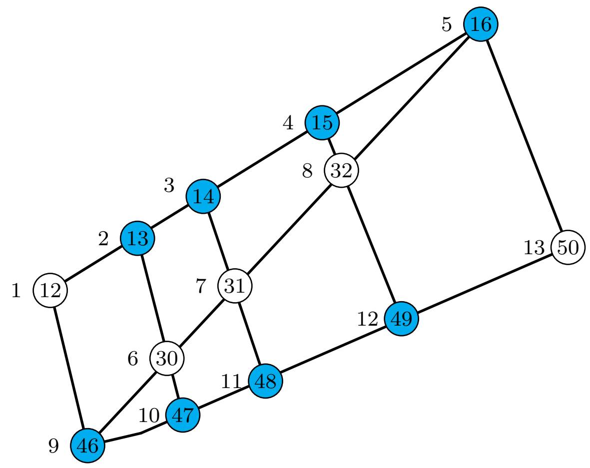

Figure 11: Changing the order vertices in the matrix and the graph

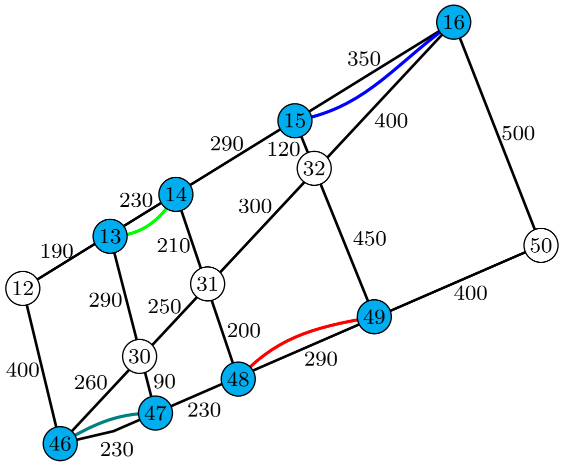

Figure 12: The result of step 4

From the above discussion, one can notice that the cycle paths are: 1 → 2 → 3 → 2 → 6 → 7 → 3 → 4 → 5 → 4 → 8 → 5 → 13 → 12 → 8 → 7 → 11 → 12 → 11 → 10 → 6 → 9 → 10 → 9 → 1.

From the cycle paths and Figure 12, it can be observed that the Euler cycle is 12 → 13 → 14 → 13 → 30 → 31 → 14 → 15 → 16 → 15 → 32 → 16 → 50 → 49 → 32 → 31 → 48 → 49 → 48 → 47 → 30 → 46 → 47 → 46 → 12.

We find that the number of vertices that the cycle goes through is 13, the number of edges that the cycle goes through is 24. And the total length of the road that the vehicle must perform on each journey according to the graph is 6680 meters.

We now discuss to solving about the real problem of the map of District 5 in Ho Chi Minh city, Vietnam in the next sub-section.

4.2 Drawing the entire road map of District 5 in Ho Chi Minh city

The steps to draw the road guide for the vehicle to run in District 5 of Ho Chi Minh city are similar to drawing a small part of the map. However, since there are 128 vertices for the graph of modeling the entire map of District 5 in Ho Chi Minh city, Vietnam, we cannot write the detail of the matrices A_1, A_2, A_3, A_4, and A_5. However, they are available upon request. Over here, we only provide the result of Step 4 for the cycle paths as follows:

1 → 2 → 3 → 2 → 18 → 17 → 33 → 34 → 18 → 19 → 3 → 4 → 5 → 4 → 20 → 19 → 35 → 34 → 51 → 33 → 69 → 70 → 51 → 52 → 35 → 36 → 25 → 20 → 21 → 5 → 6 → 7 → 8 → 7 → 23 → 24 → 8 → 9 → 10 → 11 → 9 → 42 → 10 → 28 → 29 → 11 → 12 → 13 → 14 → 13 → 30 → 31 → 14 → 15 → 16 → 15 → 32 → 16 → 50 → 49 → 32 → 31 → 48 → 47 → 30 → 46 → 12 → 29 → 45 → 44 → 28 → 42 → 41 → 24 → 41 → 40 → 23 → 27 → 26 → 22 → 6 → 22 → 21 → 25 → 26 → 38 → 37 → 36 → 53 → 52 → 71 → 72 → 53 → 54 → 37 → 39 → 38 → 55 → 54 → 73 → 72 → 95 → 94 → 71 → 94 → 93 → 92 → 70 → 104 → 95 → 96 → 73 → 74 → 55 → 56 → 39 → 40 → 57 → 56 → 75 → 74 → 97 → 96 → 114 → 115 → 97 → 98 → 87 → 65 → 48 → 49 → 66 → 65 → 64 → 47 → 46 → 45 → 62 → 46 → 63 → 62 → 61 → 44 → 43 → 42 → 59 → 58 → 57 → 76 → 58 → 77 → 76 → 75 → 99 → 98 → 116 → 117 → 99 → 118 → 119 → 77 → 78 → 59 → 60 → 43 → 60 → 61 → 80 → 62 → 83 → 82 → 81 → 79 → 78 → 120 → 79 → 80 → 81 → 121 → 122 → 82 → 84 → 83 → 100 → 101 → 84 → 85 → 63 → 64 → 86 → 85 → 102 → 101 → 124 → 102 →

$125 \rightarrow 86 \rightarrow 87 \rightarrow 88 \rightarrow 66 \rightarrow 67 \rightarrow 68 \rightarrow 50 \rightarrow 90 \rightarrow 68 \rightarrow 67 \rightarrow 89 \rightarrow 88 \rightarrow 126 \rightarrow 127$ $\rightarrow 89 \rightarrow 90 \rightarrow 128 \rightarrow 127 \rightarrow 126 \rightarrow 125 \rightarrow 124 \rightarrow 123 \rightarrow 100 \rightarrow 123 \rightarrow 122 \rightarrow 121 \rightarrow 120$ $\rightarrow 119 \rightarrow 118 \rightarrow 117 \rightarrow 116 \rightarrow 115 \rightarrow 114 \rightarrow 113 \rightarrow 104 \rightarrow 103 \rightarrow 105 \rightarrow 106 \rightarrow 107 \rightarrow$ $112 \rightarrow 111 \rightarrow 109 \rightarrow 106 \rightarrow 108 \rightarrow 109 \rightarrow 111 \rightarrow 110 \rightarrow 108 \rightarrow 105 \rightarrow 103 \rightarrow 93 \rightarrow 92 \rightarrow$ $91 \rightarrow 69 \rightarrow 17 \rightarrow 1$.

From our analysis, we find that the number of vertices that the cycle goes through is 128, the number of edges that the cycle goes through is 251, and the total length of the road that the vehicle must perform on each journey according to the graph is 55295.

5 Conclusion

Our contribution in this paper is to introduce in detail the algorithm to solve the problem to obtain the shortest path. We utilize three algorithms including Fleury, Floyd, and Greedy algorithms to analyze to the problem of "Assigning vehicles to collect garbage" in District 5, Ho Chi Minh City, Vietnam. That is to solve the problem to obtain the shortest path. We then apply the approach to draw the road guide for the vehicle to run in District 5 of Ho Chi Minh city. To do so, we first draw a small part of the map and then draw the entire road map of District 5 in Ho Chi Minh city.

The approach recommended in our paper is reliable and useful for managers to use our approach to get the optimal (it is minimal in this case) cost and travelling time. If managers do not use our approach, their travel cost and travelling time will not be optimal and the managers could pay higher price for travelling and spend more time in travelling. In this paper, we only apply the approach to solve the problem to obtain the shortest path for District 5, Ho Chi Minh city, Vietnam.

The algorithms are introduced in this paper can be utilized to address numerous practical issues including path of watering car, checking traffic, mail delivering, etc. These algorithms will work ineffectively if the data set, pictures and figures have missing values or errors in measurement. About the methods to solve the problems have missing values can look at in Little (1992), Pho and Nguyen (2018) and Pho et al. (2019). The algorithms recommend in this article can be applied to every place in the world. This is the profound contribution of our paper.

References

Agafonov, A. A., & Myasnikov, V. V.: Method for the reliable shortest path search in time-dependent stochastic networks and its application to GIS-based traffic control. Computer Optics, 40(2), 275-283 (2016).

Ahuja, R. K., Mehlhorn, K., Orlin, J., & Tarjan, R. E.: Faster algorithms for the shortest path problem. Journal of the ACM (JACM), 37(2), 213-223 (1990).

Aly, A., & Cleemput, S.: An Improved Protocol for Securely Solving the Shortest Path Problem and its Application to Combinatorial Auctions. IACR Cryptology ePrint Archive, 2017, 971 (2017).

Banyai, T., Tamas, P., Illes, B., Stankeviciute, Z., & Banyai, A.: Optimization of municipal waste collection routing: Impact of industry 4.0 technologies on environmental awareness and sustainability. International journal of environmental research and public health, 16(4), 634 (2019).

Broumi, S., Bakali, A., Talea, M., Smarandache, F., Kishore, K. K., & ahin, R.: Shortest path problem under interval valued neutrosophic setting. Journal of Fundamental and Applied Sciences, 10(4S), 168-174 (2018).

Cao, S., Xu, D., & Liu, S.: A Study of the Relationships between the Characteristics of the Village Population Structure and Rural Residential Solid Waste Collection Services: Evidence from China. International journal of environmental research and public health, 15(11), 2352 (2018).

Chassein, A. B., & Goerigk, M.: A new bound for the midpoint solution in minmax regret optimization with an application to the robust shortest path problem. European Journal of Operational Research, 244(3), 739-747 (2015).

Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S.: Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment. Applied Soft Computing, 12(3), 1231-1237 (2012).

Edmonds, J.: Matroids and the greedy algorithm. Mathematical programming, 1(1), 127-136 (1971).

Eiselt, H. A., Gendreau, M., & Laporte, G.: Arc routing problems, part I: The Chinese postman problem. Operations Research, 43(2), 231-242 (1995).

Euler, L.: Mechanica sive motus scienta analytice exposita instar supplementi ad Commentar. Acad. scient. imper. ex typographia Academiae scientarum (1736).

Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C.: An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. Networks: An International Journal, 44(3), 216-229 (2004).

Floyd, R. W.: Algorithm 97: shortest path. Communications of the ACM, 5(6), 345 (1962).

Garaix, T., Artigues, C., Feillet, D., & Josselin, D.: Vehicle routing problems with alternative paths: An application to on-demand transportation. European Journal of Operational Research, 204(1), 62-75 (2010).

Handler, G. Y., & Zang, I.: A dual algorithm for the constrained shortest path problem. Networks, 10(4), 293-309 (1980).

Harary, F.: The determinant of the adjacency matrix of a graph. SIAM Review, 4(3): 202-210 (1962).

Hassin, R.: Approximation schemes for the restricted shortest path problem. Mathematics of Operations research, 17(1), 36-42 (1992).

Kumar, R., Edaltpanah, S. A., Jha, S., Broumi, S., & Dey, A.: Neutrosophic Shortest Path Problem. Neutrosophic Sets & Systems, 23 (2018).

Lawler, E. L.: A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem. Management science, 18(7), 401-405 (1972).

Liang, J., & Liu, M.: Network Design for Municipal Solid Waste Collection: A Case Study of the Nanjing Jiangbei New Area. International journal of environmental research and public health, 15(12), 2812 (2018).

Little, R. J.: Regression with missing X's: a review. Journal of the American Statistical Association, 87(420), 1227-1237 (1992).

Lozano, L., & Medaglia, A. L.: On an exact method for the constrained shortest path problem. Computers & Operations Research, 40(1), 378-384 (2013).

Marinakis, Y., Migdalas, A., & Sifaleras, A.: A hybrid particle swarm optimizationvariable neighborhood search algorithm for constrained shortest path problems. European Journal of Operational Research, 261(3), 819-834 (2017).

Montemanni, R., & Gambardella, L. M.: An exact algorithm for the robust shortest path problem with interval data. Computers & Operations Research, 31(10), 1667-1680 (2004).

Mullai, M., Broumi, S., & Stephen, A.: Shortest path problem by minimal spanning tree algorithm using bipolar neutrosophic numbers. Infinite Study (2017).

Pho, K. H., & Nguyen, V. T.: Comparison of Newton-Raphson algorithm and Maxlik function. Journal of Advanced Engineering and Computation, 2(4), 281-292 (2018).

Pho, K. H., Ly, S., Ly, S., & Lukusa, T. M.: Comparison among Akaike Information Criterion, Bayesian Information Criterion and Vuong's test in Model Selection: A Case Study of Violated Speed Regulation in Taiwan. Journal of Advanced Engineering and Computation, 3(1), 293-303 (2019).

Vimercati, L., Baldassarre, A., Gatti, M., De Maria, L., Caputi, A., Dirodi, A., ... & Bellino, R.: Respiratory health in waste collection and disposal workers. International journal of environmental research and public health, 13(7), 631 (2016).

Zeng, W., Miwa, T., & Morikawa, T.: Application of the support vector machine and heuristic k-shortest path algorithm to determine the most eco-friendly path with a travel time constraint. Transportation Research Part D: Transport and Environment, 57, 458-473 (2017).

Zhang, Y., Zhang, Z., Deng, Y., & Mahadevan, S.: A biologically inspired solution for fuzzy shortest path problems. Applied Soft Computing, 13(5), 2356-2363 (2013).