

Distribution Route Optimization of Zakat Al-Fitr Based on the Branch-and-Bound Algorithm

Noor Saif Muhammad Mussafi¹

¹Departement of Mathematics. UIN Sunan Kalijaga. Indonesia

noor.mussafi@uin-suka.ac.id

ABSTRACT

Article History:

Received : 04-08-2022

Revised : 07-12-2022

Accepted : 16-12-2022

Online : 12-01-2023

Keywords:

Zakat al-fitr;

'Amil;

Distribution route optimization;

Branch-and-bound algorithm;

Traveling salesman problem.



The short interval between the collecting and distribution of zakat al-fitr is a recurring issue. As a result, 'amil does not always pay attention to the ideal route, leading in inefficient transportation expenditures. This study aims to minimize the amount of vehicle mileage that affects fuel consumption. The branch-and-bound algorithm was employed to overcome the distribution route optimization problem by proposing the shortest circuit that traverses each district exactly once and returns to its original district. The procedures involve data collecting, graph analysis, branch-and-bound analysis, MATLAB code development, and the recommendation of the best route. The results indicate that the branch-and-bound algorithm can numerically solve the distribution route optimization corresponding to traveling salesman problem. Furthermore, according to a case study of zakat al-fitr distribution conducted by Eradication of Illiteracy Al Quran (PBHA), the total optimal distance of the computational-based algorithm was 152.9 km, with inter-village routes starting from Sidorejo and then via Sumberarum, Pendoworejo, Gerbosari, Banjaroyo, Banjarasri, Sendangagung, Tuksono, Argodadi, Triwidadi, Jatimulyo, Giripurwo, and ends in Sidorejo.



<https://doi.org/10.31764/jtam.v7i1.10375>



This is an open access article under the **CC-BY-SA** license

A. INTRODUCTION

In the month of Ramadan, prior to the Muslim festive season celebration known as eid al-fitr, every Muslim (whether male or female, slave or free, wealthy or poor, young or elderly) is obligated to fulfill zakat al-fitr by donating a certain amount of essential food or paying an equal monetary amount. The items must be provided as much as one sha' in the form of local staple foods, such as rice, beans, lentils, wheat, corn, and so forth (Al-Qaradawi, 2000; Perdana & Tunali, 2020). The two advantages of zakat al-fitr are (1) the purification of a Muslim's fast from needless activities and hurtful words; and (2) the expression of sympathy for the needy so that their eid al-fitr day are fulfilled and they do not beg. This viewpoint is based on the hadith narrated by Ibn 'Abbas: "The Messenger of Allah enjoined Zakat-ul-fitr on the one who fasts (i.e. fasted during the month of Ramadan) to purify him from any indecent act or speech and for the purpose of providing food for the needy. It is accepted as Zakah for the person who pays it before the Eid prayer and it is Sadaqah (i.e. voluntary charity) for the person who pays it after the Eid prayer" (Al-Albani, 2014).

'Amil, persons who appointed by the government or community to collect, manage, and distribute zakat (Fathoni, Suryani, & Cahyo, 2020; Usmani & Qazi, 2010), has a narrow time interval between the collection and distribution deadlines. The last collection is on the last day of Ramadan, at dusk, and must be handed to all recipients before the following morning eid al-fitr prayer (Dhar, 2013). As a result, for the purpose of the virtue of the zakat al-fitr reward, effective distribution management is required, because any delay will merely be counted as *sodaqoh*, as explained in the hadith above. One way to mitigate the issues mentioned above is to use distribution route optimization so that services can be completed in a reasonable amount of time.

The Traveling Salesman Problem (TSP) is a combinatorial optimization problem that has piqued humanity's interest in identifying the shortest distribution route (Dahiya & Sangwan, 2018; Grymin & Jagiełło, 2016; Xiong, Wu, & Wu, 2017). The branch-and-bound algorithm is one of the TSP-solving algorithms that can be used to achieve the goal of this study regarding to the distribution route. It is a recommended method to produce exact algorithms solving NP-hard combinatorial optimization and discrete problems (Gmys, Mezmaz, Melab, & Tuyttens, 2016; Grymin & Jagiełło, 2016; Rajarajeswari & Maheswari, 2020; Yu, Xu, & Schober, 2020). The idea behind the algorithm is to use the estimated bound and the least cost search strategy to accelerate the finding of the solution vertex. In particular, it decomposes the problem into subproblems over a tree structure known as a branch-and-bound tree (Cochran, Cox, Keskinocak, Kharoufeh, & Smith, 2011; Huang et al., 2021; Morrison, Jacobson, Sauppe, & Sewell, 2016).

Researchers have focused a lot of emphasis on the application of the branch-and-bound algorithm in realistic combinatorial optimization. Some real-world issues are taken into account, including the Electrical Multiple Units (EMUs), single-machine scheduling, and close-enough traveling salesman problem (CETSP). Lu, Zhou, Yue, & Chen (2016) found that EMUs circulation scheduling in railway network can be designed using branch-and-bound algorithm under certain condition. Over a pre-step of greedy strategy, they demonstrated that the objective value of solutions and the execution time of branch-and-bound algorithm triumph over other heuristics. Additionally, Zegordi & Yavari (2018) suggests that in a single-machine scheduling problem with non-identical work release dates, the overall completion durations be minimized. All the proposed method, including lower bound, upper bound, and dominance properties, have been used in the branch-and-bound algorithm and have shown their potency in solving hard and large-size problems quickly. Prior to this, Coutinho, Nascimento, Pessoa, & Subramanian (2016), discusses CETSP, which is restricted by an obligation of salesman to visit a particular area. They applied an exact branch-and-bound algorithm and second order cone programming to examine 824 examples, and in the end they were able to find the best solutions for open problems with up to a thousand vertices. This research is a modification of Coutinho, Nascimento, Pessoa, & Subramanian (2016) using a realistic and different public space. i.e., distribution route of zakat al-fitr. In order to achieve the optimal results and the shortest amount of time, this study was also inspired by Lu, Zhou, Yue, & Chen (2016) and Zegordi & Yavari (2018). More specifically, as a means to obtain the least operational cost, this research improves the application of the branch-and-bound method to the route network and creates the systemic steps of the branch-and-bound algorithm that are translated into the MATLAB

programming language. Therefore, this study aims to minimize the chosen route to distribute zakat al-fitr in Sleman city and Kulon Progo city so that it will be positively correlated with time efficiency.

B. METHODS

In general, this research involves two diciplines: mathematics for route optimization and fiqh (Islamic jurisprudence) for zakat al-fitr regulation and implementation. So, this applied research sought to identify the best distribution route of zakat al-fitr. The research was conducted at a non-profit organization named Eradication of Illiteracy Al Quran (PBHA) in Klodran, Sumberagung, Sleman, Special Region of Yogyakarta that specialized in eradication of illiteracy Al Quran and social help. The study allows for two phases, including quantitative data analysis in a computing laboratory and qualitative data analysis (Figure 1).

1. Quantitative data analysis

The data used includes ‘amil, recipient pockets, and the kilometer-long separations between the sites. A graph representation of the identified data will then be created. To find the shortest route for distributing zakat al-fitr, the traveling salesman problem theory, matrix reduction, and branch-and-bound algorithm were applied. Finally, using MATLAB software and the principles of algorithm analysis, construct a computational programming for the corresponding case study, as shown in Figure 1.

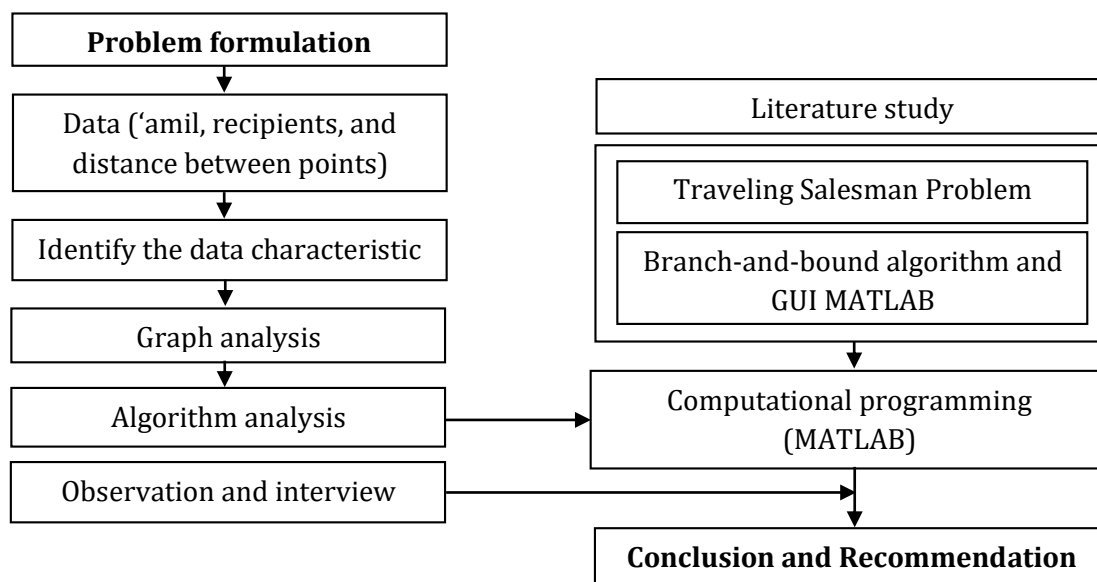


Figure 1. Research flowchart

2. Qualitative data analysis

This step involved gathering as much information as possible based on the issue and the available data through observation and interview. The following will outline the research’s theoretical framework which includes branch-and-bound algorithm and GUI MATLAB.

a. Branch-and-bound algorithm

According to Suyanto (2010) and Zumaytis & Karnalim (2017), the branch-and-bound algorithm that based on reduced cost matrix has the following steps:

- 1) Create a cost matrix that indicates the connectivity and the weight/distance to travel between each vertex. Fill it with infinity (∞) if two vertices are not connected.
- 2) Form the reduced matrix by subtracting each row and column with the lowest weight. Once each row and column contains at least an element of zero, the matrix is reduced. The cost of the root vertex is equal to the total of all the row and column reducer.
- 3) Let A be a reduced matrix for vertex R and S be a branch of vertex R . For each vertex connected to the starting vertex, do this to reduced matrix A .
 - a) Replace the value of all elements in row R and column S into ∞
 - b) Replace $(S,1)$ into ∞
 - c) Re-reduce the matrix
 - d) Let $\hat{c}(S)$ be the minimum cost of travel through vertex S , $\hat{c}(R)$ be the minimum cost of travel through root vertex R , $A(i, j)$ weight of edges (i, j) in the reduced matrix, and r be the sum of all reducer in the reduced matrix for vertex S . Then

$$\hat{c}(S) = \hat{c}(R) + A(i, j) + r \quad (1)$$

- 4) Select one of the active vertices that has the lowest cost, then expand it by repeating step c on that vertex. When there is no longer any active vertex, stop.

b. GUI MATLAB

GUIs, also known as graphic user interfaces, are simply items provided by MATLAB that enable users work with numerical computing and data processing using graphical interfaces like push buttons, sliders, radio buttons, toggle buttons, pop-up menus, and so on (Mengarelli et al., 2016; Sharma, Sharma, & Sahu, 2017). In MATLAB there are two basic methods for creating GUIs: writing the GUI program from scratch, or using the built-in Graphical User Interface Development Environment (GUIDE). According to Messac (2015), GUIDE allows the user to graphically lay out the GUI, and then MATLAB generates the code for it automatically. Thus, only two files—the fig-file and the m-file—can be used to create the MATLAB GUI program (Figure 2). In general, an m-file is a list of MATLAB commands that may be run in a batch process by just inputting their name. There are two types of m-files, namely script file and function file. This study employed the function m-files, which differ from script files in that they can accept one or more variables as inputs and can return one or more variables as outputs (Lopez, 2014), as shown in Figure 2.

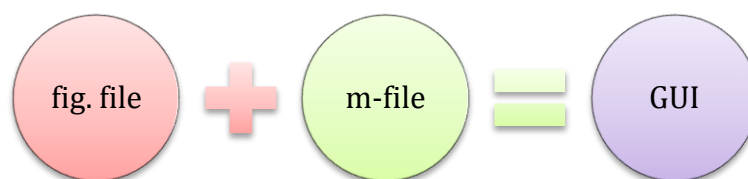


Figure 2. GUI formation

C. RESULT AND DISCUSSION

1. Data identification

At present, PBHA has hundreds of students studying Al Quran and is able to gather zakat, infaq, and sodaqoh from the public to be distributed to designated locations. Each year, as an 'amil, they collect and deliver more than a ton of rice from donations to the eleven foster villages. There are still certain challenges, particularly in the distribution of zakat al-fitr, such as the issue of the delay between receiving and dispersing rice. Till one in the morning (1 Shawwal), they get leftover rice from mosques in Sleman and Yogyakarta city. However, the rice needs to reach the destination locations (which are pretty far away) before Eid prayers, or else zakat al-fitr will merely be converted into sodaqoh. For this reason, 'amil needs an efficient zakat al-fitr distribution route to ensure that rice is delivered on time.

According to Mr. Abdullah Efendi's responses to an interview, there are eleven foster villages in the districts of Sleman and Kulon Progo covered by PBHA. The starting point (PBHA headquarters) is in (1) Sidorejo Sleman and the foster villages include; (2) Sendangagung Sleman; (3) Banjarasri Kulon Progo; (4) Banjaroyo Kulon Progo; (5) Gerbosari Kulon Progo; (6) Pendoworejo Kulon Progo; (7) Giripurwo Kulon Progo; (8) Jatimulyo Kulon Progo; (9) Tuksono Kulon Progo; (10) Triwidadi Kulon Progo; (11) Argodadi Sleman; and (12) Sumberarum Sleman. In addition. Table 1 displays the separation between locations in kilometres, as shown in Table 1.

Table 1. Distances between foster villages (kilometres)

Location	1	2	3	4	5	6	7	8	9	10	11	12
1	-	7.4	12.5	16.6	19.1	11.5	15.9	23.6	19.4	18.5	13.5	7.1
2	7.4	-	5.7	9.9	12.3	7.6	12	19.7	21.5	21.5	16.6	6.2
3	12.5	5.7	-	4.9	11.8	11.2	17.1	23.2	26.6	26.6	21.7	11.6
4	16.6	9.9	4.9	-	12.1	15.3	21.2	27.4	30.8	30.5	25.8	15.7
5	19.1	12.3	11.8	12.1	-	16.2	20.9	21.8	33.2	33.2	28.3	18.2
6	11.5	7.6	11.2	15.3	16.2	-	7.3	12.9	20.8	24.2	19.3	8.2
7	15.9	12	17.1	21.2	20.9	7.3	-	10.9	20.4	24.8	18.8	12.5
8	23.6	19.7	23.2	27.4	21.8	12.9	10.9	-	24.1	30.6	24.6	20.3
9	19.4	21.5	26.6	30.8	33.2	20.8	20.4	24.1	-	19.6	13.5	17
10	18.5	21.5	26.6	30.5	33.2	24.2	24.8	30.6	19.6	-	6.3	16.1
11	13.5	16.6	21.7	25.8	28.3	19.3	18.8	24.6	13.5	6.3	-	11.2
12	7.1	6.2	11.6	15.7	18.2	8.2	12.5	20.3	17	16.1	11.2	-

2. Graph analysis

Let $G = (V, E)$ be a graph contains a non-empty set $V(G)$ of vertices and a finite set $E(G)$ of unordered pairs of vertices called edges. Assuming that the twelve locations are vertex set $V(G) = \{1, 2, 3, \dots, 12\}$, the road joining them or the edges set $E(G) = \{(1, 2), (1, 3), (1, 4), \dots, (1, 12), (2, 1), \dots, (11, 12)\}$ can be obtained. Each edge has the weight or the number representing the distance (in kilometers) between locations. Such graph G along with the distance between the locations (as indicated in Table 1) is illustrated in Figure 3.

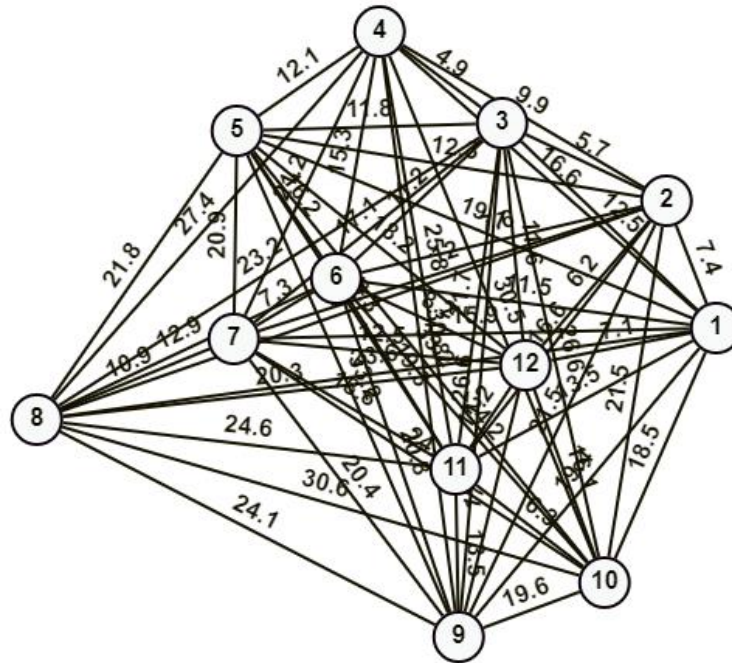


Figure 3. Graph representation

The TSP rule, which is a closed path with a minimum weight that traverses each vertex once, will be used to analyze Figure 3. In practice, the path departs from vertex 1 (PBHA headquarters), visits each of the eleven foster villages once, and then loops back around. Such mechanism is indeed referring to the Hamiltonian graph (Ibrahim & Mussafi, 2013). Note that the TSP mandates the minimum total distance of travel. One of the algorithms for solving TSP is the branch-and-bound algorithm.

3. Branch-and-bound algorithm analysis

A symmetric 12×12 matrix can be used to represent Table 1 (see Figure 4). As previously indicated, the branch-and bound algorithm primarily uses a reduced cost matrix to determine the limit value of cost of the root vertex and iteration to establish the route to be taken. The decreased cost matrix starts with a row reduction and then moves on to a column reduction. Reduce row 1 of the matrix in Figure 4 by 7.1, and so on until the matrix in Figure 5 is obtained, by subtracting each element from all rows by the least c_{ij} value in each of those rows, as shown in Figure 4 and Figure 5.

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	7.4	12.5	16.6	19.1	11.5	15.9	23.6	19.4	18.5	13.5	7.1
2	7.4	-	5.7	9.9	12.3	7.6	12	19.7	21.5	21.5	16.6	6.2
3	12.5	5.7	-	4.9	11.8	11.2	17.1	23.2	26.6	26.6	21.7	11.6
4	16.6	9.9	4.9	-	12.1	15.3	21.2	27.4	30.8	30.5	25.8	15.7
5	19.1	12.3	11.8	12.1	-	16.2	20.9	21.8	33.2	33.2	28.3	18.2
6	11.5	7.6	11.2	15.3	16.2	-	7.3	12.9	20.8	24.2	19.3	8.2
7	15.9	12	17.1	21.2	20.9	7.3	-	10.9	20.4	24.8	18.8	12.5
8	23.6	19.7	23.2	27.4	21.8	12.9	10.9	-	24.1	30.6	24.6	20.3
9	19.4	21.5	26.6	30.8	33.2	20.8	20.4	24.1	-	19.6	13.5	17
10	18.5	21.5	26.6	30.5	33.2	24.2	24.8	30.6	19.6	-	6.3	16.1
11	13.5	16.6	21.7	25.8	28.3	19.3	18.8	24.6	13.5	6.3	-	11.2
12	7.1	6.2	11.6	15.7	18.2	8.2	12.5	20.3	17	16.1	11.2	-

Figure 4. A symmetric 12x12 matrix

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	0.3	5.4	9.5	12	4.4	8.8	16.5	12.3	11.4	6.4	0
2	1.7	-	0	4.2	6.6	1.9	6.3	14	15.8	15.8	10.9	0.5
3	7.6	0.8	-	0	6.9	6.3	12.2	18.3	21.7	21.7	16.8	6.7
4	11.7	5	0	-	7.2	10.4	16.3	22.5	25.9	25.6	20.9	10.8
5	7.3	0.5	0	0.3	-	4.4	9.1	10	21.4	21.4	16.5	6.4
6	4.2	0.3	3.9	8	8.9	-	0	5.6	13.5	16.9	12	0.9
7	8.6	4.7	9.8	13.9	13.6	0	-	3.6	13.1	17.5	11.5	5.2
8	12.7	8.8	12.3	16.5	10.9	2	0	-	13.2	19.7	13.7	9.4
9	5.9	8	13.1	17.3	19.7	7.3	6.9	10.6	-	6.1	0	3.5
10	12.2	15.2	20.3	24.2	26.9	17.9	18.5	24.3	13.3	-	0	9.8
11	7.2	10.3	15.4	19.5	22	13	12.5	18.3	7.2	0	-	4.9
12	0.9	0	5.4	9.5	12	2	6.3	14.1	10.8	9.9	5	-

Figure 5. A row reduced cost matrix

In contrast, using the matrix in Figure 5 as a guide, each element in each of the columns is reduced by the smallest C_{ij} value in that column, for instance, reducing column 1 of the matrix in Figure 4 by 0.9, and so on until Figure 6 is generated. Consequently, the total sum of all reducers (rows and columns) or the limit of the root vertex $\hat{c}(R) = (7.1 + 5.7 + 4.9 + 4.9 + 11.8 + 7.3 + 7.3 + 10.9 + 13.5 + 6.3 + 6.3 + 6.2) + (0.9 + 6.6 + 3.6 + 7.2) = 110.5$.

Furthermore, the iteration process will be taken as many as 11 iterations. The number of iterations is determined from the number of vertices adjacent to the root vertex. This iteration is used to determine the path of each vertex that must be passed according to the TSP rules. Additionally, the iteration process may require up to 11 iterations. The number of vertices adjacent to the root vertex is used to calculate the number of iterations. The path of each vertex that needs to be passed in accordance with the TSP rules is determined using this iteration. Each iteration's outcome will affect how the following one is calculated. Only the first iteration will be shown in this paper, as shown in Figure 6 and Figure 7.

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	0.3	5.4	9.5	5.4	4.4	8.8	12.9	5.1	11.4	6.4	0
2	0.8	-	0	4.2	0	1.9	6.3	10.4	8.6	15.8	10.9	0.5
3	6.7	0.8	-	0	0.3	6.3	12.2	14.7	14.5	21.7	16.8	6.7
4	10.8	5	0	-	0.6	10.4	16.3	18.9	18.7	25.6	20.9	10.8
5	6.4	0.5	0	0.3	-	4.4	9.1	6.4	14.2	21.4	16.5	6.4
6	3.3	0.3	3.9	8	2.3	-	0	2	6.3	16.9	12	0.9
7	7.7	4.7	9.8	13.9	7	0	-	0	5.9	17.5	11.5	5.2
8	11.8	8.8	12.3	16.5	4.3	2	0	-	6	19.7	13.7	9.4
9	5	8	13.1	17.3	13.1	7.3	6.9	7	-	6.1	0	3.5
10	11.3	15.2	20.3	24.2	20.3	17.9	18.5	20.7	6.1	-	0	9.8
11	6.3	10.3	15.4	19.5	15.4	13	12.5	14.7	0	0	-	4.9
12	0	0	5.4	9.5	5.4	2	6.3	10.5	3.6	9.9	5	-

Figure 6. A column reduced cost matrix

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	-	-	-	-	-	-	-	-	-	-	0
2	-	-	0	4.2	0	1.9	6.3	10.4	8.6	15.8	10.9	6.2
3	6.7	-	-	0	0.3	6.3	12.2	14.7	14.5	21.7	16.8	10.3
4	10.8	-	0	-	0.6	10.4	16.3	18.9	18.7	25.6	20.9	5.9
5	6.4	-	0	0.3	-	4.4	9.1	6.4	14.2	21.4	16.5	0.4
6	3.3	-	3.9	8	2.3	-	0	2	6.3	16.9	12	4.7
7	7.7	-	9.8	13.9	7	0	-	0	5.9	17.5	11.5	8.9
8	11.8	-	12.3	16.5	4.3	2	0	-	6	19.7	13.7	3
9	5	-	13.1	17.3	13.1	7.3	6.9	7	-	6.1	0	9.3
10	11.3	-	20.3	24.2	20.3	17.9	18.5	20.7	6.1	-	0	4.4
11	6.3	-	15.4	19.5	15.4	13	12.5	14.7	0	0	-	-
12	0	-	5.4	9.5	5.4	2	6.3	10.5	3.6	9.9	5	0

Figure 7. A new matrix for first iteration that represent an edge from vertex 1 to vertex 2

Table 2. The final result of branch-and-bound algorithm

Iteration	Original vertex	Destination vertex	The minimum cost of travel through $S \hat{c}(S)$	Distance (km)
1	1	2 or 12	111.3	7.4
2	2	5	111.7	19.7
3	5	6	116.1	35.9
4	6	8	122.4	48.8
5	8	7	122.4	59.7
6	7	9	126.6	80.1
7	9	11	132.9	93.6
8	11	10	132.9	99.9
9	10	12	138.3	116
10	12	3 or 4	149.1	127.6
11	3	4	149.1	132.5
12	4	1	149.1	149.1

Vertex 1 and the other 11 connected vertices will be examined in the first iteration's calculation, i.e., (1,2), (1,3), (1,4), ..., (1,12). For vertex 2 (the edge connecting vertex 1 and vertex 2):

- a. To ensure that no pathways leave vertex 1 or enter vertex 2, change all values in row 1 and column 2 of the reduced matrix A to "-".

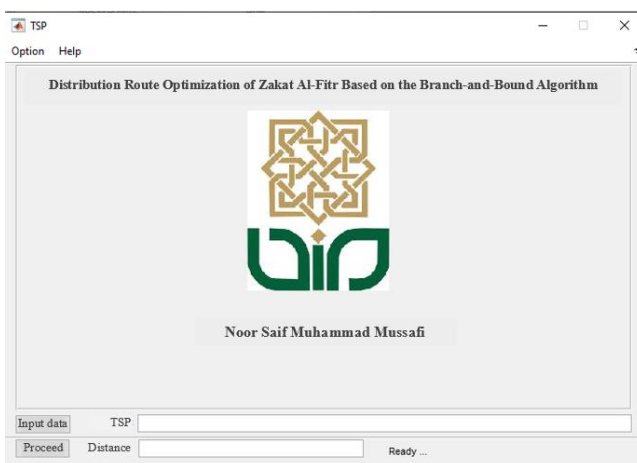
- b. Change A (2,1) to “-“ (element row 2 column 1 in matrix A) to prevent the use of edge (2,1).
- c. Reduce rows and columns with the smallest c_{ij} value. Since there isn't row that doesn't include zero, the column with the smallest c_{ij} value is reduced. The smallest c_{ij} value can be found in column 12 of matrix A. thus reduce all of its elements by 0.5 (for the result see Figure 7). Hence. the total of all reducers is $(r) = 0.5$.
- d. Using equation (1), determine the boundary value of vertex 2 (the edge from vertex 1 to vertex 2) to $\hat{c}(S) = \hat{c}(R) + A(i, j) + r \leftrightarrow \hat{c}(2) = \hat{c}(1) + A(1,2) + r = 110.5 + 0.3 + 0.5 = 111.3$. Thus. the boundary value of vertex 2 or $\hat{c}(2) = 111.3$.

The remaining vertices that have been calculated in the first iteration have a value of $\hat{c}(3) = 117.3, \hat{c}(4) = 120.8, \hat{c}(5) = 116.4, \hat{c}(6) = 115.4, \hat{c}(7) = 122.1, \hat{c}(8) = 123.9, \hat{c}(9) = 116.1, \hat{c}(10) = 122.4, \hat{c}(11) = 126.5, \hat{c}(12) = 111.3$.

Vertex 2 and vertex 12 which has the smallest limit value in the search tree are retrieved once the limit value from vertex 1 to the other 11 vertices has been obtained. Therefore the second iteration will explore either vertex 2 or vertex 12. Furthermore, the analysis of the following iterations (2 to 11) is analogous to the prior iteration and is shown in more detail in Table 2. This results in a visit path with a total distance of 149.1 km either (1-2-5-6-8-7-9-11-10-12-3-4-1) or (1-2-5-6-8-7-9-11-10-12-4-3-1).

4. Computational programming

The experiments in this computational programming were conducted on a PC with Intel Core i3 Processor 7020U running at 2.30 GHz and 8 Gb of RAM. The algorithms were coded in MATLAB R2017a 9.2.0.538062. Row and column reduction matrices, GUI figures, distance matrices, TSP paths, and plotting are only a few of the several types of MATLAB m-file code that have been developed, as shown in Figure 8.



(a)

0	7.4	12.5	16.6	19.1	11.5	15.9	23.6	19.4	18.5	13.5	7.1
7.4	0	5.7	9.9	12.3	7.6	12	19.7	21.5	21.5	16.6	6.2
12.5	5.7	0	4.9	11.8	11.2	17.1	23.2	26.6	26.6	21.7	11.6
16.6	9.9	4.9	0	12.1	15.3	21.2	27.4	30.8	30.5	25.8	15.7
19.1	12.3	11.8	12.1	0	16.2	20.9	21.8	33.2	33.2	28.3	18.2
11.5	7.6	11.2	15.3	16.2	0	7.3	12.9	20.8	24.2	19.3	8.2
15.9	12	17.1	21.2	20.9	7.3	0	10.9	20.4	24.8	18.8	12.5
23.6	19.7	23.2	27.4	21.8	12.9	10.9	0	24.1	30.6	24.6	20.3
19.4	21.5	26.6	30.8	33.2	20.8	20.4	24.1	0	19.6	13.5	17
18.5	21.5	26.6	30.5	33.2	24.2	24.8	30.6	19.6	0	6.3	16.1
13.5	16.6	21.7	25.8	28.3	19.3	18.8	24.6	13.5	6.3	0	11.2
7.1	6.2	11.6	15.7	18.2	8.2	12.5	20.3	17	16.1	11.2	0

(b)

Figure 8. (a) The GUI Matlab case-based appearance and (b) the input data of distance matrix

A form similar to that seen in Figure 8a will appear after running the m-file. A notepad file is used to store the input data related to the distances between locations (Figure 8b). The button "proceed" refers to the action of solving the given distribution problem using the branch-and-bound algorithm to identify the shortest closed path.

Additionally, the twelve locations will be inspected using a genuine map that was created using Google Maps, as shown in Figure 8a. The plotting code identifies the shortest route of the problem (Figure 8b). By pressing the "proceed" button, the 152.9 km long TSP route of 1-12-6-5-4-3-2-9-11-10-8-7-1 is obtained. The route passes through Sidorejo, Sumberarum, Pendoworejo, Gerbosari, Banjaroyo, Banjarasri, Sendangagung, Tuksono, Argodadi, Triwidadi, Jatimulyo, and Giripurwo before returning to Sidorejo. The map is complemented by yellow and red connecting lines, as seen in the program output (Figure 9b). The red line represents the return to the final vertex while the yellow line represents the departure from the initial vertex. By avoiding the designated region and adopting a more accurate public setting, i.e., distribution route of zakat al-fitr in some form, this research alters Coutinho, Nascimento, Pessoa, & Subramanian (2016) work, and the map guide is the end result.

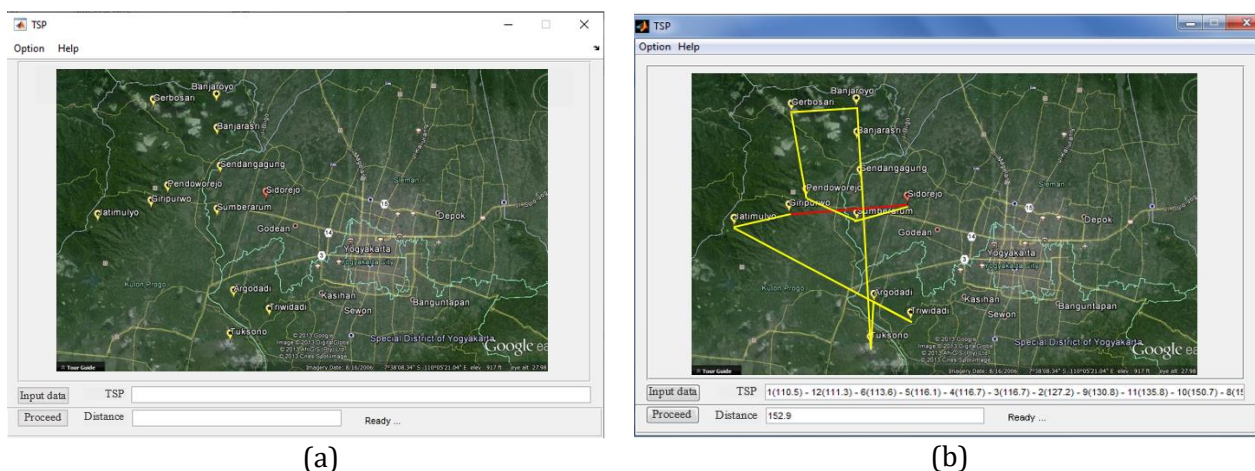


Figure 8. (a) Gmap-based research object location and (b) the result of plotting

D. CONCLUSION AND SUGGESTIONS

Through this research, numerous methodical processes can be developed to optimize the zakat fitrah distribution route at PBBA such that no zakat fitrah switches status to sodaqoh. The five systematic procedures include data collecting, table of distance between points, graph analysis, branch-and-bound algorithm analysis, and comparative studies. Furthermore, it may be suggested that a branch-and-bound algorithm approach can be used as a substitute technique for the community in general or, more particularly, for zakat, infaq, and sodaqoh institutions like PBBA, to determine the minimum distribution route of a closed path both manually and computationally. The results indicate that manual-based algorithmic analysis was somewhat more efficient than computational-based algorithm analysis in producing the total distance distribution of zakat al-fitr for this case study. This study can be expanded upon to be used in areas unrelated to routes, such as research objects based on operating costs. Additionally, manual algorithm analysis yield better results than computational one, leaving room for additional review or enhancement of the MATLAB program's syntax.

ACKNOWLEDGEMENT

I wish to extend my sincere gratitude to PBBA, in particular to Mr. Abdullah Effendi, for all the information and data provided. I also want to thank Head of Mathematics Department and Dean of Faculty of Science of UIN Sunan Kalijaga for their support in making this research possible.

REFERENCES

- Al-Albani, M. N. (2014). *Shahih Al Jami' Ash-Shaghir* (4th ed.). Jakarta: Pustaka Azzam.
- Al-Qaradawi, Y. (2000). *Fiqh Al-Zakat: A Comparative Study of Zakat, Regulations and Philosophy in the Light of Al-Quran and Al-Sunnah*. Jeddah, Kingdom of Saudi Arabia: Scientific Publishing Centre, King Abdul Aziz University, Vol. 1.
- Cochran, J. J., Cox, L. A., Keskinocak, P., Kharoufeh, J. P., & Smith, J. C. (2011). *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons.
- Coutinho, W. P., Nascimento, R. Q. D., Pessoa, A. A., & Subramanian, A. (2016). A branch-and-bound algorithm for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 28(4), 752–765.
- Dahiya, C., & Sangwan, S. (2018). Literature review on travelling salesman problem. *International Journal of Research*, 5(16), 1152–1155.
- Dhar, P. (2013). Zakat as a measure of social justice in Islamic finance: an accountant's overview. *Journal of Emerging Economies and Islamic Research*, 1(1), 1–11. Retrieved from <https://core.ac.uk/download/pdf/328805993.pdf>
- Fathoni, M. A., Suryani, & Cahyo, E. N. (2020). Zakat Management Paradigm: Comparison of Indonesia, Malaysia and Saudi Arabia. *INFERENSI: Jurnal Penelitian Sosial Keagamaan*, 14(2), 267–282. Retrieved from <https://inferensi.iainsalatiga.ac.id/index.php/inferensi/article/view/3417/pdf>
- Gmys, J., Mezma, M., Melab, N., & Tuytens, D. (2016). A GPU-based Branch-and-Bound algorithm using Integer-Vector-Matrix data structure. *Parallel Computing*, 59, 119–139.
- Grymin, R., & Jagiełło, S. (2016). Fast Branch and Bound Algorithm for the Travelling Salesman Problem. In: Saeed, K., Homenda, W. (Eds) *Computer Information Systems and Industrial Management. CISIM 2016*. https://doi.org/https://doi.org/10.1007/978-3-319-45378-1_19
- Huang, L., Chen, X., Huo, W., Wang, J., Zhang, F., Bai, B., & Shi, L. (2021). Branch and bound in mixed integer linear programming problems: A survey of techniques and trends. *ArXiv Preprint ArXiv:2111.06257*.
- Ibrahim, & Mussafi, N. S. M. (2013). *Pengantar Kombinatorika & Teori Graf*. Yogyakarta: Graha Ilmu.
- Lopez, C. P. (2014). *MATLAB Optimization Techniques*. Apress Academic.
- Lu, C., Zhou, L. S., Yue, Y. X., & Chen, R. (2016). A branch and bound algorithm for the exact solution of the problem of EMU circulation scheduling in railway network. *Hindawi: Mathematical Problems in Engineering*. Retrieved from <https://downloads.hindawi.com/journals/mpe/2016/8537089.pdf>
- Mengarelli, A., Cardarelli, S., Verdini, F., Burattini, L., Fioretti, S., & Di Nardo, F. (2016). A MATLAB-based graphical user interface for the identification of muscular activations from surface electromyography signals. *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 3646–3649. IEEE.
- Messac, A. (2015). Optimization in Practice with MATLAB®. In *Cambridge University Press*. <https://doi.org/10.1017/cbo9781316271391>
- Morrison, D. R., Jacobson, S. H., Sauppe, J. J., & Sewell, E. C. (2016). Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19, 79–102.
- Perdana, D. A., & Tunal, F. (2020). Zakat Fitrah: Management, Tradition, And Meaning Of Eidal-FitR. *Fikri: Jurnal Kajian Agama, Sosial Dan Budaya*, 5(2), 223–235. Retrieved from <https://journal.iaimnumetrolampung.ac.id/index.php/jf/article/view/978/620>
- Rajarajeswari, P., & Maheswari, D. (2020). Travelling Salesman Problem Using Branch And Bound Technique. *International Journal of Mathematics Trends and Technology (IJMTT)*, 66(5), 202–206.
- Sharma, M., Sharma, S., & Sahu, G. (2017). Designing MATLAB GUI for various Analog and Digital Communication Systems. *International Journal of Trend in Scientific Research and Development*, 2(1), 1397–1405.
- Suyanto. (2010). *Algoritma Optimasi (Deterministik atau Probabilistik)*. Yogyakarta: Graha Ilmu.

- Usmani, M. I. A., & Qazi, B. A. (2010). *Guide to Zakah: Understanding and Calculation*. Maktaba Ma'ariful Qur'an.
- Xiong, N., Wu, W., & Wu, C. (2017). An improved routing optimization algorithm based on travelling salesman problem for social networks. *Sustainability*, 9(6), 985.
- Yu, X., Xu, D., & Schober, R. (2020). Optimal Beamforming for MISO Communications via Intelligent Reflecting Surfaces. *IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 1–5. <https://doi.org/10.1109/SPAWC48557.2020.9154337>
- Zegordi, S. H., & Yavari, M. (2018). A branch and bound algorithm for solving large-scale single-machine scheduling problems with non-identical release dates. *European Journal of Industrial Engineering*, 12(1), 24–42.
- Zumaytis, S., & Karnalim, O. (2017). Introducing an educational tool for learning branch & bound strategy. *Journal of Information Systems Engineering and Business Intelligence*, 3(1), 8–15.