Integra: Journal of Integrated Mathematics and Computer Science

Journal Homepage: https://integrajimcs.com/index.php/integra/index

integra
Journal of Integrated Mathematics and Computer Science

Research Paper

Solving the Traveling Salesman Problem on a Directed Graph Using Greedy Algorithm (Case Study: Locations of BRI Bank in Bandar Lampung City)

Rehsya Nurfabella[1], Siti Laelatul Chasanah[1]*, Notiragayu[1]

[1]*Department of Mathematics , Faculty of Mathematics and Natural Sciences, Universitas Lampung, Lampung, 35145, Indonesia*
***Corresponding author**: siti.chasanah@fmipa.unila.ac.id

Keywords
Travelling Salesman Problem, Directed Graph, BRI Banks, Greedy Algorithm, Python

Abstract
The traveling salesman problem is the idea that a salesman must discover the shortest path between an origin point and many destination points, returning to the origin point after visiting the destination point once. In this study, the Greedy Algorithm will be used to solve the Traveling Salesman Problem on a directed graph which represented BRI Banks in Bandar Lampung city. The locations of the banks are represented by points, while the journey time between BRI Banks is represented by lines. According to the results, 130 minutes was the same amount of time spent manually and with the Python software.

Received: 29 December 2023, Accepted: 6 February 2024
https://doi.org/10.26554/integra.2024117

1. INTRODUCTION

Nowadays, banking has an important function in advancing the country's economy. Every sector involved in financial transactions requires bank services. Each bank works with money suppliers who distribute money to banks that work with these suppliers. In the distribution of goods, the efficient and best tour is needed to make the trip more efficient. Efforts to minimize expenses are also called optimization. One of the problems involving the optimization process is the Travelling Salesman Problem (TSP) [1].

Conveniently, TSP can be described as the problem of determining the shortest circuit that must be traveled by a salesman, who departs from the origin city and visits each city exactly once and then returns to the city of departure. Cities can be represented as graph nodes, while lines represent roads that connect two cities. The weight on an edge represents the distance between two cities [2].

The TSP is considered one of the classic problems in network design problems and is very popular. The problem was mathematically formulated by the Irish mathematician, Willian Rowan Hamilton who invented the icosian game, a mathematical game in 1856. The TSP shares similar properties with this game, hence it is also known as one of the Hamiltonian problems. The TSP gained popularity in scientific circles in the United States and Europe throughout the 1950s and 1960s after efforts to solve it were rewarded by the RAND Corporation in Santa Monica [3].

The TSP is considered as a NP-hard problem. Consequently, it is computationally challenging to determine the ideal solution (the shortest route), especially as the number of cities increases, even though a solution can be tested somewhat rapidly [4, 5, 6]. Because it becomes computationally costly to determine the shortest path among the numerous alternative routes as the number of cities increases, the TSP is essentially an NP-hard issue. The main objective of TSP is to find the minimum length of the Hamiltonian circuit of a graph [7].

TSP has been widely investigated and many methods have been developed, both exact and heuristics. Among the methods that have been developed are Brute Force, Nearest Neighbour, Genetic Algorithm, Particle Swarm Optimization, Branch and Bound, and many others. The Brute and Force with Graphic Processing Unit is used to solve the TSP [8]. Some researchers that employ the Brute and Force method including [9, 10]. The Nearest Neighbour Heuristics is used by [11, 12, 13] for finding the best distribution route, and Hougardy & Wilde [14] used The Nearest Neighbour Heuristics for metric TSP. Genetics Algorithm is used by Doumi et al [15] to solve the TSP, and Duka [16] used Branch and Bound in Lindo Programming to solve the TSP. The Cheapest Insertion Heuristics (CIH) is used to determine

the best route for product distribution [17, 18, 19]. Additional details regarding numerous TSP can be found in [20].

In this study we will discuss about finding the best tour of locations of BRI banks in Bandar Lampung city. Because some roads in Bandar Lampung city are one-way roads, then the locations of the banks and the road connecting them can be represented by directed graph, and that is possible that the time needed from location A to B, and from B to A are different. To find the best tour connecting all the banks, the greedy algorithms will be employed.

2. METHODS

A graph is a discrete structure consisting of points and lines. A graph is a set pair $G(V, E)$, where $V = \{v_1, v_2, v_3, ..., v_n\}V$ is a non-empty set of points and $E = e_{ij}|i, j \epsilon V, E$ is the set of lines connecting a pair of points in the graph. When the lines of a graph have no designated orientation, the graph is said to be undirected. The directed graph $G(V, A)$ is a graph with directed lines, and those lines are usually called as arcs.

Greedy Algorithm

The Greedy Algorithm is an algorithm that can determine the shortest path between the points to be used by taking continuously and adding it to the path to be passed. The following are the steps needed to solve TSP by greedy algorithm:

1. Start from any destination, if the origin is fixed then start from that destination.
2. Evaluate the distance to the destination.
3. Select the destination with the closest distance and repeat step two until all destinations have been passed.
4. Count the number of destinations that have been passed.

3. RESULTS AND DISCUSSION

3.1 The Data

Table 1 gives information about the location of the BRI banks, and Table 2 displays the time needed to go from one bank location to other banks. Figure 2 shows the graph representation of the locations. Note that in Figure 1, the black and red line indicate different direction.

Note that in Figure 1, the black and red line indicate different direction.

3.2 Implementation of Greedy Algorithm

The implementation of the Greedy Algorithm in solving the Traveling Salesman Problem to determine the shortest path between BRI Banks in Bandar Lampung city as follows:

1. Select the starting point.
2. Choose the shortest path to the next unvisited point.
3. Add the nearest point to the path and remove it from the unvisited points.
4. Add the fastest travel time from the current point to the nearest point to the total travel time.
5. Move to the next closest point.
6. Repeat steps 2, 3, 4, and 5 until all points are visited.

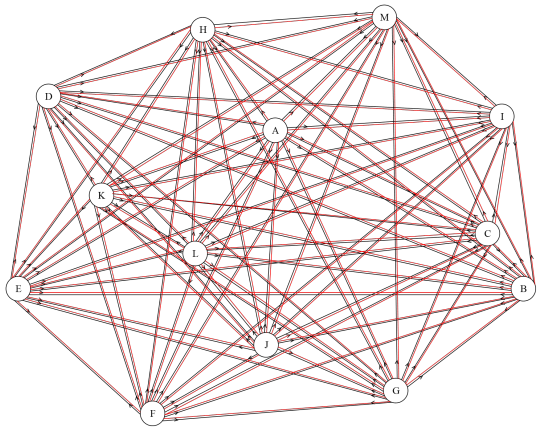

Figure 1. Graph Representation for the Problem

7. Return to the starting point.
8. Finish.

There are 13 banks under consideration. Figure 2 displays all possible tours (every location is considered as the starting point).

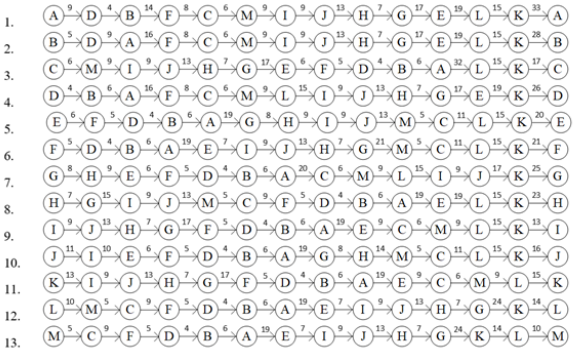

Figure 2. The 13 Paths Obtained with Different Starting Point

The travel time obtained from the calculation of the Greedy Algorithm is:

- Start and end at A, the total travel time is 163 minutes.
- Start and end at B, the total travel time is 161 minutes.
- Start and end at C, the total travel time is 146 minutes.
- Start and end at D, the total travel time is 155 minutes.
- Start and end at E, the total travel time is 130 minutes.
- Start and end at F, the total travel time is 143 minutes.
- Start and end at G, the total travel time is 139 minutes.
- Start and end at H, the total travel time is 149 minutes.
- Start and end at I, the total travel time is 132 minutes.
- Start and end at J, the total travel time is 130 minutes.
- Start and end at K, the total travel time is 132 minutes.
- Start and end at L, the total travel time is 132 minutes.
- Start and end at M, the total travel time is 132 minutes.

The best tour is start and end at E or J (there are two best tours), with total time needed 130 minutes.

We not only solve the problem manually, but also solve using

Table 1. The Location of the BRI Banks

No	Point	District	BRI Bank	Address
1	A	East Teluk Betung	BRI Bank Unit Kotakarang	Laksamana R.E.Martadinata Rd. No.6 TBB, Perwata, East Teluk Betung District, Bandar Lampung, Lampung 35451
2	B	Bumi Waras	BRI Bank Unit Bumi Waras	Sultan Hasanudin Rd. No.55, Kangkung, Bumi Waras District. Bandar Lampung, Lampung 35221
3	C	Kedamaian	BRI Bank Unit Kedamaian	Antasari Rd. No.125d, Tj. Baru, Kedamaian District, Bandar Lampung, Lampung 35131
4	D	North Teluk Betung	BRI Bank Unit Sumur Batu	Diponegoro Rd. No.88, Sumur Batu, North Teluk Betung District, Bandar Lampung, Lampung 35212
5	E	Center Tanjung Karang	BRI Bank Unit Center Tj. Karang	Kapten A. Rivai Rd. No.7, Penengahan, Center Tanjung Karang, Bandar Lampung, Lampung 35145
6	F	East Tanjung Karang	BRI Bank Unit East Tj. Karang	Raden Intan Rd. Rt 03 Rw 04, No. 245, Kotabaru, Tanjung Karang Kotabaru, East Tanjung Karang District, Bandar Lampung, Lampung 35127
7	G	Kemiling	BRI Bank Unit Kemiling	Teuku Cik Ditiro Rd., Sumber Rejo, Kemiling District, Bandar Lampung, Lampung 35155
8	H	Langkapura	BRI Bank Unit Langlapura	Imam Bonjol Rd. No.576, Langkapura, Langkapura Districy, Bandar Lampung, Lampung 35118
9	I	Kedaton	BRI Bank Unit Kedaton	Mall Boemi Kedaton, Basement, Block GB 17, Teuku Umar R, No 1, Kedaton District, Bandar Lampung, Lampung 35141
10	J	Rajabasa	BRI Bank Unit Unila	Prof. Dr. Ir. Sumantri Brojonegoro Rd. No.1, Gedong Meneng, Rajabasa District, Bandar Lampung, Lampung 35141
11	K	Tanjung Senang	BRI Bank Jatimulyo	Ratu Dibalau Rd.No.20, Way Kandis, Kec. Tanjung Senang District, Bandar Lampung, Lampung 35365
12	L	Sukarame	BRI Bank Unit IAIN	Letkol H. Endo Suratmin Rd., IAIN Raden Intan Campus, Sukarame District, Bandar Lampung, Lampung 35142
13	M	Way Halim	BRI Bank Unit Urip Sumoharjo	Urip Sumoharjo Rd., Gn. Sulah, Way Halim District, Bandar Lampung, Lampung 35132

Table 2. The Location of the BRI Banks

Point	A	B	C	D	E	F	G	H	I	J	K	L	M
A	0	10	20	9	19	16	19	20	22	31	34	32	26
B	6	0	17	5	17	14	19	21	20	29	31	28	21
C	19	14	0	12	17	9	23	19	15	20	16	11	6
D	9	4	12	0	12	9	17	15	16	23	26	22	16
E	20	15	9	11	0	6	18	12	7	15	19	19	12
F	15	9	8	5	10	0	17	13	13	20	23	18	13
G	21	21	25	19	17	17	0	8	15	16	24	28	21
H	20	21	17	17	9	10	7	0	9	12	19	23	14
I	26	21	14	17	10	12	16	12	0	9	12	15	10
J	30	25	18	21	15	16	14	13	11	0	17	21	13
K	33	28	17	26	20	21	25	23	13	16	0	14	13
L	30	25	13	22	21	19	30	25	15	21	15	0	10
M	27	19	5	18	11	13	21	16	9	16	13	9	0

the Phyton language programming. Figure 3 shows the screenshot oof the source code, Figure 4 shows the output when the original tour at point E, and Figure 5 displays the output when the original tour at point J.

```
def greedy_tsp(city_distances, start_city):
    """Menggunakan algoritma Greedy untuk menyelesaikan TSP."""
    num_cities = len(city_distances)
    route = [start_city]
    unvisited_cities = set(city_distances.keys())
    unvisited_cities.remove(start_city)

    current_city = start_city
    total_distance = 0

    while unvisited_cities:
        nearest_city = None
        nearest_distance = float('inf')

        # Search for the nearest unvisited cities
        for next_city in unvisited_cities:
            distance = city_distances[current_city][next_city]
            if distance < nearest_distance:
                nearest_distance = distance
                nearest_city = next_city

        # Adding the nearest city to the tout dan remove it from
unvisited cities
        route.append(nearest_city)
        unvisited_cities.remove(nearest_city)
        current_city = nearest_city
        total_distance += nearest_distance

    # Back to origin
    route.append(start_city)
    total_distance += city_distances[current_city][start_city]

    return route, total_distance
```

Figure 3. Screenshot of the Source Code

```
# Solving TSP using Greedy algorithm
start_city = 'E'
route, total_distance = greedy_tsp(city_distances, start_city)

print("optimal tour:", route)
print("Total distance", total_distance)

optimal tour: ['E', 'F', 'D', 'B', 'A', 'G', 'H', 'I', 'J', 'M', 'C', 'L', 'K', 'E']
Total distance: 130
```

Figure 4. Output When the Starting Point is E

```
# Solving TSP using Greedy algorithm
start_city = 'J'
route, total_distance = greedy_tsp(city_distances, start_city)

print("optimal tour:", route)
print("Total distance:", total_distance)

optimal tour: ['J', 'I', 'E', 'F', 'D', 'B', 'A', 'G', 'H', 'M', 'C', 'L', 'K', 'J']
Total distance: 130
```

Figure 5. Output When the Starting Point is J

4. CONCLUSIONS

Based on the Results and Discussion, it can be concluded that the manual calculation and using Python programming language obtained the same result. There are two shortest paths obtained, namely E-F-D-B-A-G-H-I-J-M-C-L-K-E and J-I-E-F-D-B-A-G-H-M-C-L-K-J with a total travel time of 130 minutes.

5. ACKNOWLEDGEMENT

The authors wish to thank the Department of Mathematics, Universitas Lampung for the support.

REFERENCES

[1] Saiful Rohman, La Zakaria, Asmiati Asmiati, and Aang Nuryaman. Optimisasi travelling salesman problem dengan algoritma genetika pada kasus pendistribusian barang pt. pos indonesia di kota bandar lampung. *Jurnal Matematika Integratif*, 16(1):61–73, 2020 (in Indonesia).

[2] Admi Syarif, Wamiliana Wamiliana, and Yasir Wijaya. Evaluasi kinerja metode-metode heuristik untuk penyelesaian travelling salesman problem. *Jurnal Sains MIPA Universitas Lampung*, 2(2), 2012 (in Indonesia).

[3] Norman Biggs. The traveling salesman problem a guided tour of combinatorial optimization. *Bulletin of the London Mathematical Society*, 18(5):514–515, 1986.

[4] Gerhard Reinelt. *The Traveling Salesman: Computational Solutions for Tsp Applications*, volume 840. Springer, 2003.

[5] Mohammad Asim, Ritika Gopalia, and Shivalika Swar. Traveling salesman problem using genetic algorithm. *International Journal of Latest Trends in Engineering and Technology*, 3(3):183–190, 2014.

[6] David B Fogel. An evolutionary approach to the traveling salesman problem. *Biological Cybernetics*, 60(2):139–144, 1988.

[7] Alvendo Wahyu Aranski. Optimization of the smallest road using the traveling salesman problem (tsp) method. *IJISTECH (International Journal of Information System and Technology)*, 6(1):159–166, 2022.

[8] Andrew Wilson, Yuliant Sibaroni, and Izzatul Ummah. Analisis penyelesaian traveling salesman problem dengan metode brute force menggunakan graphic processing unit. *eProceedings of Engineering*, 2(1):1874–1883, 2015 (in Indonesia).

[9] Muchamad Kurniawan, Farida Farida, and Siti Agustini. Rute terpendek algoritma particle swarm optimization dan brute force untuk optimasi travelling salesman problem. *Jurnal Teknik Informatika*, 14(2):191–200, 2021 (in Indonesia).

[10] Sriyani Violina. Analysis of brute force and branch & bound algorithms to solve the traveling salesperson problem (tsp). *Turkish Journal of Computer and Mathematics Education*, 12(8):1226–1229, 2021.

[11] Sitti Rosnafi'an Sumardi, Nur Nilam Sari, and Justin Eduardo Simarmata. Product distribution route using nearest neighbor algorithm. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, 4(3):894–900, 2024.

[12] Imam Sutoyo. Penerapan algoritma nearest neighbour untuk menyelesaikan travelling salesman problem. *Paradigma*, 20(1):101–106, 2018 (in Indonesia).

[13] Anie Lusiani, Siti Samsiyah Purwaningsih, and Euis Sartika. Tsp method using nearest neighbor algorithm at pt. j&t express in bandung. *Jurnal Lebesgue: Jurnal Ilmiah Pen-*

didikan Matematika, Matematika dan Statistika, 4(3):1560–1568, 2023.

[14] Stefan Hougardy and Mirko Wilde. On the nearest neighbor rule for the metric traveling salesman problem. *Discrete Applied Mathematics*, 195:101–103, 2015.

[15] AB Doumi, BA Mahafzah, and H Hiary. Solving traveling salesman problem using genetic algorithm based on efficient mutation operator. *Journal of Theoretical and Applied Information Technology*, 99(15):3768–3781, 2021.

[16] Erjon Duka. Traveling sales problem solved by branch and bound algorithm in lindo programming. *Available at SSRN 3152830*, 2018.

[17] L Virginayoga Hignasari and Eka Diana Mahira. Optimization of goods distribution route assisted by google map with cheapest insertion heuristic algorithm (cih). *Sinergi*, 22(2):132–138, 2018.

[18] Kadek Meliantari, D Putra Githa, and NK Ayu Wirdiani. Optimasi distribusi produk menggunakan metode cheapest insertion heuristic berbasis web. *Merpati*, 6(3):204, 2018 (in Indonesia).

[19] Rio Guntur Utomo, Dian Sa'adillah Maylawati, and Cecep Nurul Alam. Implementasi algoritma cheapest insertion heuristic (cih) dalam penyelesaian travelling salesman problem (tsp). *Jurnal Online Informatika*, 3(1):61–67, 2018 (in Indonesia).

[20] Omar Cheikhrouhou and Ines Khoufi. A comprehensive survey on the multiple traveling salesman problem: Applications, approaches and taxonomy. *Computer Science Review*, 40:100369, 2021.

