Jurnal Komputasi

Vol 13 No. 2, 2025

Comparison of the Cheapest Insertion Heuristic Algoritma, Christofides Algorithm, and Nearest Neighbor Algorithm for Determining Hospital Tours in Bandar Lampung City

[1]Thomas Juliansyah, [2]Riska Aulia Putri, [3]Roro Ayu Martinez, [4]Muslim Ansori & [5,*)]Wamiliana

[1,2,3,4,5] Department of Mathematics, Universitas Lampung Jl. Prof. Soemantri Brojonegoro No. 1, Bandar Lampung, Indonesia

Abstract — *Determining the optimal route is one of the important aspects in planning the distribution of health services, especially in emergency conditions in Bandar Lampung City. This study compares three algorithms for solving tour problems, namely Cheapest Insertion Heuristic, Nearest Neighbor (NN), and Christofides Algorithm, in determining the fastest tour to a number of hospitals. Calculations were performed manually and also implemented using the Python programming language. The results obtained show that manually and using Python programming, the Cheapest Insertion Heuristic algorithm produced 152 minutes, the Nearest Neighbour algorithm 142 minutes, and the Christofides Algorithm 147 minutes.*

Keywords: Cheapest Insertion Heuristic Algorithm; Christofides Algorithm; Nearest Neighbor Algorithm; Travelling Salesman Problem

** Corresponding author :*
Wamiliana
Dept. of Mathematics, FMIPA Universitas Lampung
Jl. Prof. Dr. Soemantri Brojokegoro No. 1,
Bandarlampung Email:
Wamiliana.1963@fmipa.unila.ac.id

1. INTRODUCTION

One of the most frequently visited healthcare facilities is the hospital. A hospital is a medical facility that offers all types of personal healthcare services, such as emergency, outpatient, and inpatient treatment [1]. Pharmaceutical services in hospitals are an integral part of the hospital healthcare system, which is patient- centered and focuses on providing quality and affordable pharmaceuticals, medical devices, and disposable medical supplies for all segments of society [2]. The need for medical devices and medications must be met daily to ensure uninterrupted healthcare services, necessitating speed and accuracy in their delivery.

To ensure that logistics and medication needs are met, it is necessary to optimize the distribution routes or tour for logistics and medications to minimize distances and reduce delivery times. Delivery times serve as a benchmark for the hospital's success in improving the quality of healthcare services through the timely and accurate availability of medical devices [3]. In this case study, the logistics and medication distribution routes to be optimized are for all hospitals in Bandar Lampung, thereby minimizing the time required for logistics distribution from the central hospital to each of the other hospitals in Bandar Lampung.

Determining the fastest route by visiting all points exactly once is known as the Traveling Salesmen Problem (TSP), where a salesman starts from one city, travels precisely to each of his destination cities, and then returns to his original city. The TSP is considered an NP-hard problem, thus, determining the optimal solution (the shortest route or tour) becomes a significant computational challenge, especially as the number of cities increases, although such solutions can be tested relatively quickly [4], [5], [6]. Since determining the shortest route among many alternative routes becomes increasingly computationally expensive as the number of cities increases, the TSP is fundamentally an NP-hard problem. The primary goal of the TSP is to find the minimum length of a Hamilton circuit in a graph [7].

The TSP has been extensively studied, and various methods have been developed, both exact and heuristic. Among the methods developed are Brute Force, Nearest Neighbor, Genetic Algorithms, Particle Swarm Optimization, Branch and Bound, the cheapest insertion heuristic algorithm, the Christofides algorithm, and the artificial bee colony algorithm, among others. The Nearest Neighbour heuristic was used by [8, 9, 10] to find the best distribution route, and Hougardy & Wilde

[11] used the Nearest Neighbour heuristic for metric TSP. Cheapest Insertion Heuristics (CIH) are used to determine the best route for product distribution [12, 13, 14] and to find the tourist location [15]. Some researchers that used Christofides algorithm include [16, 17, 18, 19, 20, 21]. In solving TSP, the concept of graph theory, which is a branch of mathematics, is used. The TSP graph represents a weighted network, where each vertex in the graph represents a city, each edge represents a road connecting those cities, and the weight on each edge represents the distance between those cities that needs to be traveled.

In this study, the focus is on determining the shortest route for the distribution of logistics and medicines 19 hospitals in Bandar Lampung. The data used is time (travel time), and comparing the CIH, the Christofides algorithm, and the nearest neighbor algorithms..

2. METHOD

1.
2.

2.1. Cheapest Insertion Heuristic (CIH)

In general, CIH is a simple algorithm that gradually inserts the places to be visited and calculates the distance traveled. It starts by making a simple tour between two vertices and then revising the tour by adding a new vertex to make a bigger tour, and repeating that step until all vertices are on the tour. In this study, we made a slight modification on CIH algorithm by making the tour of three vertices in the beginning. The CIH procedure is as follows:

Step 1: Make a tour of three vertices, and calculate the weight (distance) Step 2: Find candidate edges to be selected for inclusion in the new subtour.

Step 3: Calculate new total weight for each selected candidate edge using the formula:

Total weight of previous tour + weight of edges to be added total weight of edges to be removed + weight of edges connecting the removed point to the added point.

Step 4: Find the smallest weight value among the calculated candidate edges.

Step 5: Create a new subtour with the selected edges. Repeat these steps until all points are included in the subtour (for the next step, start from step 2 with the new total weight and subtour obtained from the previous step).

2.2 Nearest Neighbor (NN) Algorithm

This algorithm is simple and straightforward algorithm. Starting with an arbitrary city, the nearest neighbour method for TSP continually chooses the closest unexplored city to the current city until all cities have been visited. The following is the procedure of the NN algorithm [22].

Step 1: All vertices are set as unvisited.

Step 2: Select a vertex randomly as original vertex and make it as the current vertex i and set i as visited vertex.

Step 3: Find the shortest path between vertices in the unvisited set with current vertex i, suppose that vertex is j. Set j as visited and current vertex.

Step 4: Repeat Step iii) until all vertices are set visited and back to the original vertex.

According to [23], the nearest neighbor method only requires a short time, is simple and easy to understand, and is effective on large data sets.

2.3 Christofides Algorithm

The Christofides algorithm is used to generate approximate solutions in TSP on complete weighted undirected graphs. This algorithm finds the minimum weight value using the Minimum Spanning Tree (MST) to produce a graph with an optimal value. Next, the results of the MST algorithm are used to form an Euler circuit, which serves as an approximation of the TSP solution [18]. The following are the steps of the Christofides algorithm for solving TSP:

Step 1: Find the minimum spanning tree.
Step 2: Determine the odd degree vertices on the MST obtained.
Step 3: Form the Eulerian circuit by connecting the odd degree vertices.
Step 4: Reduces the degree of vertices whose degree more than two by removing one or more edges adjacent to it until gain a Hamiltonian circuit.

2.4 The Data

The data used in this study was obtained from Google Maps. Table 1 below shows the information about the travel time (in minutes) among19 hospitals in Bandar Lampung.

Vertex	The hospital's name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	RSUD Dr. H. Abdul Moeloek	0	12	20	9	5	16	16	10	14	2	9	16	10	6	7	12	5	8	15
2	RS. A dadi Tjokrodipo	12	0	22	5	13	23	20	18	22	11	14	4	8	14	9	5	10	17	9
3	RS Jiwa Bandar Lampung	20	22	0	22	21	26	12	26	12	21	28	26	29	25	26	28	24	23	32
4	RS Bumi Waras	9	5	22	0	10	20	20	16	20	9	11	8	7	11	6	7	7	13	11
5	RS Umum Advent Bandar Lampung	5	13	21	10	0	13	15	7	12	4	10	18	12	8	9	14	8	6	10
6	RS Umum Imanuel Way Halim	16	23	26	24	13	0	22	7	19	15	13	27	17	12	18	22	17	14	23
7	RS Pertamina Bintang Amin	16	20	12	20	15	22	0	20	4	16	26	24	26	19	24	25	20	15	29
8	RS Umum Urip Sumoharjo	10	18	26	16	7	7	20	0	18	10	11	27	15	5	13	19	13	11	21
9	RS Bhayangkara Polda Lampung	14	22	12	20	12	19	4	18	0	16	26	24	27	17	23	25	21	12	29
10	RS DKT Lampung	2	11	21	9	4	15	17	10	16	0	9	17	12	6	10	14	7	8	18
11	RS Graha Husada	9	14	28	11	10	13	26	11	26	9	0	15	6	10	8	13	7	14	16
12	RSIA Restu Bunda	16	4	26	8	18	27	24	27	24	17	15	0	12	19	13	7	13	22	9
13	RSIA Bunda Asy-Syifa	10	8	29	7	12	17	26	15	27	12	6	12	0	12	6	6	8	17	10
14	RSIA Puri Betik Hati	6	14	25	11	8	12	19	5	17	6	10	19	12	0	13	18	12	10	21
15	RSIA Mutiara Putri	7	9	26	6	9	18	24	13	23	10	8	13	6	13	0	8	3	16	12
16	RSIA Santa Anna	12	5	28	7	14	22	25	19	25	14	13	7	6	18	8	0	11	22	5
17	RS. Hermina Lampung	5	10	24	7	8	17	20	13	21	7	7	13	8	12	3	11	0	17	13
18	RSIA Delleza Kedaton	8	17	23	13	6	14	15	11	12	8	14	22	17	10	16	22	17	0	28
19	RS. Budi Medika	15	9	32	11	18	23	29	18	16	9	16	9	10	21	12	5	13	28	0

3. RESULT AND DISCUSSION

3.

3.1. Solution using Cheapest Insertion Heuristics (CIH)

Based on the solution steps described in the research methodology, the following is a manual solution in the form of an iteration table shown in Table 3.1. However, because there are 17 iterations, the table only shows calculations for the first two iterations and the last two iterations.

Table 2**.** Iteration table for calculations using the Cheapest Insertion Heuristics

Iteration	Subtour	Total weight	$\|E\| = n$	Edge Candidate to be chosen	The Calculation	The chosen edge that to be added to the tour	The removed edge	The new total weight	The new subtour
1	[illegible]	11	No	e[illegible] = 5 e[illegible] = 6 e[illegible] = 6	11 − c[illegible] + c[illegible] + c[illegible] = 11 − 5 + 5 + 8 = 19 11 − c[illegible] + c[illegible] + c[illegible] = 11 − 5 + 6 + 8 = 20 11 − c[illegible] + c[illegible] + c[illegible] = 11 − 5 + 6 + 8 = 20	e[illegible] e[illegible]	e[illegible]	19	[illegible]
2	[illegible]	19	No	e[illegible] = 6 e[illegible] = 3 e[illegible] = 6 e[illegible] = 6	19 − c[illegible] + c[illegible] + c[illegible] = 19 − 5 + 6 + 12 = 32 19 − c[illegible] + c[illegible] + c[illegible] = 19 − 8 + 3 + 9 = 23 19 − c[illegible] + c[illegible] + c[illegible] = 19 − 4 + 6 + 8 = 29 19 − c[illegible] + c[illegible] + c[illegible] = 19 − 4 + 6 + 8 = 29	e[illegible] e[illegible]	e[illegible]	23	[illegible]
The 3rd – 15th iterations are performed here for efficiency									
16	[illegible]	127	No	e[illegible] = 20 e[illegible] = 24 e[illegible] = 26 e[illegible] = 28 e[illegible] = 29 e[illegible] = 32 e[illegible] = 28 e[illegible] = 22 e[illegible] = 26 e[illegible] = 22 e[illegible] = 25 e[illegible] = 26 e[illegible] = 26 e[illegible] = 12 e[illegible] = 12 e[illegible] = 21 e[illegible] = 23 e[illegible] = 21	127 − c[illegible] + c[illegible] + c[illegible] = 127 − 5 + 20 + 24 = 166 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 3 + 24 + 26 = 174 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 8 + 26 + 28 = 169 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 6 + 28 + 29 = 178 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 10 + 29 + 32 = 174 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 5 + 32 + 28 = 182 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 5 + 28 + 22 = 172 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 4 + 22 + 26 = 171 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 8 + 26 + 22 = 167 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 11 + 22 + 25 = 163 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 5 + 25 + 26 = 173 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 7 + 26 + 26 = 172 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 13 + 26 + 21 = 161 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 12 + 12 + 23 = 150 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 4 + 12 + 12 = 147 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 15 + 21 + 12 = 145 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 8 + 23 + 12 = 154 127 − c[illegible] + c[illegible] + c[illegible] = 127 − 2 + 21 + 20 = 147	e[illegible] e[illegible]	e[illegible]	150	[illegible]
17	[illegible]	145	Yes						

3.2. Solution with Nearest Neighbor (NN)

By using the data from Table 1 we do the following:

1. Start with vertex $v1$ as the depot (departure point).
2. Then select the node with the shortest distance from vertex $v1$, which is vertex $v10$. Thus, the temporary route is $v1 - v10$.
3. Continue by selecting the node that has the shortest distance to vertex $v10$, which is vertex $v5$, the temporary route is $v1 - v10 - v5$.
4. Continue by selecting the node that has the shortest distance to vertex $v5$, which is vertex $v18$, the temporary route is $v1 - v10 - v5 - v18$.
5. The steps are repeated until all unselected nodes are included in the route by selecting the vertices whose smallest edge to the previous vertex.
6. The final route is $v1 - v10 - v5 - v18 - v14 - v8 - v6 - v11 - v13 - v15 - v17 - v4 - v2 - v12 - v16$ $- v19 - v9 - v7 - v3 - v1$ with a total travel time of 142 minutes. Figure 3 shows the tour.

Figure 3. The tour obtained using the NNH

In addition to manual methods, NNH solutions are also implemented using the Python programming language. Source code 2 shows a part of the source code for determining tours from 19 hospitals in Bandar Lampung City using NNH.

```
        if nearest is not None:
            route.append(nearest)
            visited[nearest] = True
            total_distance += min_dist
            current = nearest
total_distance += matrix[current][start]
    route.append(start)
    return route, total_distance
route_indices, total_distance = nearest_neighbour(distance_matrix, start=0)
route_names = [locations[i] for i in route_indices]
print("Rute yang dilalui:")
print(" → ".join(route_names))
print(f"Total jarak: {total_distance:.2f}")
```

Source code 2. Screenshot part of the source code for NNH

The program produces output with a tour $v1 - v10 - v5 - v18 - v14 - v8 - v6 - v11 - v13 - v15 - v17 - v4 - v2 - v12 - v16 - v19 - v9 - v7 - v3 - v1$ with a weight of 142 minutes.

3.3. Solution with the Christofides Algorithm

The Christofides algorithm has many possible solutions, because there are many possible ways to connect odd-degree vertex. The Christofides algorithm is performed by first determining the MST of 19 vertices using the Kruskal algorithm. Figure 4 below shows the results obtained.

Figure 4. The Minimum Spanning Tree of the 19 hospital locations in Bandar Lampung city.

The next step is to form Eulerian tour by connecting odd-degree graph points with other odd-degree vertices, resulting in even-degree vertices. As previously stated, there are many ways to form Eulerian tour because there are some vertices with odd degrees. To form a Hamiltonian tour, some edges that incident to vertices with degree greater than 2 are removed so that the degree of the vertex becomes two.

Figure 5. The formation of Eulerian tour.

Based on Figure 5, there are 4 vertices have degree more than 2. The vertex $v1$ has a degree of 4, so we need to remove 2 edges to make its degree becomes 2. However, we must pay attention to the end vertex that incident to the remove vertex, because, by removing the edges its degree reduced. Thus, we need to add an edge that connect both end vertices of the removed edges. There are four edges incident to $v1$ which are $e1,10$, $e1,17$,, $e1,14$ and $e1,5$. Suppose that edges $e1,14$ and $e1,5$ are selected for removal, thus we must add edge $e5,14$. Next, vertex $v2$ has degree 4, so we need to remove 2 edges to make its degree becomes 2. There are four edges to be considered: $e2,12$, $e2,4$,, $e2,16$ and $e2,3$. Suppose that edges $e2,16$ and $e2,3$ are selected for deletion, thus we must add edge $e16,3$. Next, vertex $v5$ has degree 4, so we need to remove 2 edges to make its degree becomes 2. There are four edges to be chosen: $e5,14$, $e5,10$, $e5,9$ and $e5,18$.

Suppose that edges $e5,14$ and $e15,9$ are chosen to be removed, then we add edge $e9,14$. Next, vertex $v15$ has degree 4, the degree will be reduced by removing 2 edges. There are four edges as candidate to be removed: $e15,4$, $e15,17$,, $e15,13$ and $e15,11$. Suppose that edges $e15,4$ and $e15,11$ are selected for deletion, then we add edge $e4,11$. Figure 6 shows that it has become a Hamiltonian circuit with tour results $v1 - v17 - v15 - v13 - v11 - v4 - v2 - v12 - v19 - v16 - v3 - v7 - v9 - v14 - v8 - v6 - v18 - v5 - v10 - v1$ with a weight of 152 minutes.

Figure 6. The tour obtained after reducing the degree of vertices v_1, v_2, v_2, and v_{15}

In addition to manual methods, the Christofides algorithm is also implemented using the Python programming language. Source code 3 below shows a part of source code of Christofides algorithm for determining a tour of 19 hospitals in Bandar Lampung City.

```
euler_circuit = list(eulerian_circuit(multigraph, source=next(iter(G.nodes))))
visited = set()
tsp_path = []
for u, v in euler_circuit:
    if u not in visited:
        tsp_path.append(u)
        visited.add(u)
```

Source code 3. Screenshot part of the source code for the Chistofides Algorithm

The source code generates output with tour $v1 - v14 - v8 - v6 - v11 - v13 - v15 - v19 - v16 - v2 - v12 - v4 - v17 - v3 - v7 - v9 - v5 - v18 - v10 - v1$ with 152 minutes, and Figure 7 shows the tour obtained.

4. CONCLUSIONS

Based on discussion above, manual calculations and implementation using the Python programming language, it was found that the Nearest Neighbor algorithm produced a total travel time of 142 minutes, while the Christofides algorithm produced 152 minutes. Meanwhile, the Cheapest Insertion Heuristic algorithm shows a difference in results, namely 145 minutes in manual calculations and 140 minutes in Python calculations. This difference may be due to differences in the order of insertion processes performed manually and automatically by the program. In general, the Cheapest Insertion Heuristic algorithm provides a more efficient solution than the other two algorithms.

ACKNOWLEDGMENTS

The authors wish to thanks Research Laboratory of Mathematics and Statistics, Universitas Lampung for the support given

REFERENCES

[1] S. Farlinda, R. Nurul, and S. A. Rahmadani. "Pembuatan Aplikasi Filling Rekam Medis Rumah Sakit ISSN : 2354-5852," Kesehatan, vol. 5, no. 1, pp. 8–13, 2017.

[2] A. Primadiamanti, G. A. R. Saputri, and D. L. Sari. "Evaluasi Penyimpanan Dan Pendistribusian Obat Di Instalasi Farmasi Rumah Sakit Mutiara Bunda Tulang Bawang,"J. Farm. Malahayati, Vol. 4(2), pp. 205–215, 2022, doi: 10.33024/jfm.v4i2.5315

[3] J. Jutriano and A. Abidin, "Integrasi Servqual, Kano Model, Dan Kansei Engineering Untuk Peningkatan Kualitas Layanan Distribusi Alat Kesehatan Urologi," *J. Compr. Sci.*, Vol. 2(12), pp. 1458–1466, 2023, doi: 10.59188/jcs.v2i12.561.

[4] G. Reinelt., *The Traveling Salesman: Computational Solutions for TSP Applications*. Springer, 2003.

[5] Ashish Gupta and Shipra Khurana, "Study of Traveling Salesman Problem Using Genetic Algorithm," *Int. J. Manag. IT& Engineering,* Vol. 3(3), pp. 183–190, 2012.

[6] D. B Essay, "An evolutionary approach to the traveling salesman problem," *Biol. Cybern.,* Vol. 3(3), pp. 139– 144, 1988.

[7] A. W. Aranski, "Optimization of The Smallest Road Using The Traveling Salesman Problem (TSP) Method," *Int. J. Inf. Syst. Technol. Akreditasi*, Vol. 6 (158), pp. 159–166, 2022.

[8] W. A. F. B and S. Rosnafi, "Product Distribution Route using Nearest Neighbor Algorithm Rute Pendistribusian Barang dengan Algoritma Nearest Neighbor," Vol. 4, pp. 894–900, 2024.

[9] I. Sutoyo, "Penerapan Algoritma Nearest Neighbour untuk Menyelesaikan Travelling Salesman Problem," J. Paradig., Vol 20 (1), p. 101, 2018.

[10] A. Lusiani, S. S. Purwaningsih, and E. Sartika, "TSP Method Using Nearest Neighbor Algorithm at PT. J&T Express in Bandung," J. Lebesgue J. Ilm. Pendidik. Mat. Mat. dan Stat., Vol. 4(3), pp. 1560–1568, 2023, doi: ggh10.46306/lb.v4i3.449.

[11] M. W. Stefan Hougardy, "On the nearest neighbor rule for the metric traveling salesman problem," Discret. Appl. Math., pp. 101–103, 2015.

[12] L. V. Hignasari and E. D. Mahira, "Optimization of Goods Distribution Route Assisted By Google Map With Cheapest Insertion Heuristic Algorithm (Cih)," Sinergi, Vol. 22(2), p. 132, 2018, doi: 10.22441/sinergi.2018.2.010.

[13] K. Meliantari, D. Putra Githa, and N. K. Ayu Wirdiani, "Optimasi Distribusi Produk Menggunakan Metode Cheapest Insertion Heuristic Berbasis Web," J. Ilm. Merpati (Menara Penelit. Akad. Teknol. Informasi), Vol. 6(3), p. 204, 2018, doi:10.24843/jim.2018.v06.i03.p07.

[14] R. G. Utomo, D. S. Maylawati, and C. N. Alam, "Implementasi Algoritma Cheapest Insertion Heuristic (CIH) dalam Penyelesaian Travelling Salesman Problem (TSP)," J. Online Inform., Vol. 3(1), p. 61, 2018, doi: 10.15575/join.v3i1.218.

[15] A. R. Dinata, Wamiliana, M. Ansori, Fitriani, and Notiragayu. Determining the Shortest Tour Location of Tourist Attractions in Bandar Lampung Using Cheapest Insertion Heuristic (CIH) and Modified Sollin Algorithm. Jurnal Pepadun, Vol. 6 (1), pp. 92-102, 2025.

[16] N.W. Hadi, R. Nurfabella, Wamiliana, and M. Mustika. "Comparative Analysis of CIH and Christofides Algorithms for Optimal Tourist Route Planning in West Java", Integra: Journal of Integrated Mathematics and Computer Science, Vol 2 (2):56 -62, 2024.

[17] M. Y. Aswin and M. Ansori, "Perbandingan Cheapest Insertion Heuristic dan Algoritma Christofides untuk Menentukan Tour Pasar Tradisional di Kota Bandar Lampung" Jurnal Pepadun, Vol. 5(2), pp. 182–194, 2024.

[18] F. J. Tjoea and S. Halim, “Evaluasi Rute Pelayaran Kapal dengan Pendekatan Modified Minimum Spanning Tree,” Jurnal Titra, Vol. 11 (2), pp. 265–272, 2023.

[19] I. K. Omar Cheikhrouhou, “A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy, Computer Science Review,” Comput. Sci. Rev., Vol. 40, 2021.

[20] D. R.A. Putri, N.A. Oktavia, S. L. Chasanah, R. Sawitri, and F. A. Paskalia.

[21] “Implementation of Christofides Algorithm to Determine the Shortest Tour of Some Hospitals in Palembang City” Integra : Journal of Integrated Mathematics and Computer Science, Vol. 2 (1), pp. 15–19.

[22] Hadi N. W., R. Nurfabella, Wamiliana, and M. Mustika. Comparative Analysis of CIH and Christofides Algorithms for Optimal Tourist Route Planning in West Java. Integra: Journal of Integrated Mathematics and Computer Science, Vol 2 (2):56 -62, 2024.

[23] M. Z. Rahman, S. R. Sheikh, A. Islam, and M. A. Rahman. Improvement of the Nearest Neighbor Heuristic Search Algorithm for Traveling Salesman Problem. Journal of Engineering Advancements, Vol 5(1), pp.19– 26, 2024.

[24] S. Mutrofin, A. Mu'alif, R. V. H. Ginardi, and C. Fatichah. Solution of class imbalance of k-nearest neighbor for data of new student admission selection. International Journal of Artificial Intelligence Research, 3(2), pp. 47- 55, 2019.