Nusantara Journal of Artificial Intelligence and Information Systems
Vol. 1, No. 2, December 2025, pp 67-76 journal homepage: https://journal.unusia.ac.id/nuai
Research Article ISSN: 3090-8876 DOI: 10.47776/nuai.v1i2.1679

Optimizing Staple Food Distribution Efficiency using the Traveling Salesman Problem in Local Government

Moh Jufri[1*], Nasrullah[2], Asyatir Rodiyah[3]
[1,2,3]Department of Informatics Engineering, Faculty of Engineering, Universitas Islam Madura, Pamekasan, Indonesia
[1]mohjufri129@gmail.com, [2]nasrullah170602@gmail.com, [3]sa0595363@gmail.com

INFORMATION

Article history:

Received Jun 04, 2025
Revised Aug 13, 2025
Accepted Oct 01, 2025

Keywords:

Traveling salesman problem
Distribution route optimization
Python programming
Distance matrix
Route search algorithm

ABSTRACT

The equitable distribution of essential goods in Indonesia's remote and archipelagic regions faces significant logistical challenges, primarily due to geographical complexity, limited transportation infrastructure, and unpredictable weather conditions. This study adopts the Travelling Salesman Problem (TSP) model to optimize inter-island aid distribution routes, with delivery locations modeled as a complete weighted graph. Inter-location distances are calculated using the Haversine formula based on actual geographical coordinates. The Path Cheapest Arc heuristic algorithm, implemented via Google OR-Tools, is employed to efficiently generate near-optimal routes. The optimization results show a reduction in travel distance by 22.7% (from 256.7 km to 198.4 km), accompanied by a decrease in estimated travel time from 10.27 hours to 7.94 hours, and a reduction in fuel consumption from 102.7 liters to 79.4 liters. These findings empirically demonstrate that TSP optimization can significantly reduce travel distance and distribution time in the context of maritime logistics. The integration of graph theory, optimization algorithms, and interactive route visualization not only enhances computational efficiency but also provides practical benefits for policymakers in the public sector. This methodological framework has the potential to be replicated for distribution scenarios in other remote regions, while also strengthening data-driven decision-making in governmental logistics planning.

To cite this article:

Jufri, M., Nasrullah., & Rodiyah, A. (2025). Optimizing Staple Food Distribution Efficiency using the Traveling Salesman Problem in Local Government. *Nusantara Journal of Artificial Intelligence and Information Systems*, 1(2), 67–76. https://doi.org/10.47776/nuai.v1i2.1679

Corresponding Author:

Moh Jufri
Informatics Engineering Department, Faculty of Engineering, Universitas Islam Madura
Jl. Pondok Pesantren Miftahul Ulum Bettet, Pamekasan, Jawa Timur, Indonesia (69317)
Email: mohjufri129@gmail.com

1. INTRODUCTION

In the context of sustainable development and equitable social welfare, the government holds a fundamental responsibility to ensure the availability of basic logistics, particularly in remote areas such as outer islands. The distribution of social assistance in the form of basic food supplies (*sembako*) serves as a tangible manifestation of the state's intervention in fulfilling the basic rights of its citizens. However, inter-island logistics distribution is far from a simple matter [1]. Complex geographical terrain, limited transport fleets, minimal infrastructure connectivity, and unpredictable weather conditions often pose major challenges to the efficient implementation of logistics distribution.

A key challenge frequently encountered is how to design an optimal delivery route—one that is efficient in terms of distance, time, and resource consumption. Inefficiencies in route planning can lead to delivery delays, fuel wastage, and increased operational costs, ultimately reducing the effectiveness of the social assistance program itself [2]. In scientific studies, this routing problem can be systematically modeled as the Travelling Salesman Problem (TSP), a classic combinatorial optimization model in operations research that aims to find the shortest possible path to visit a set of locations exactly once and return to the point of origin.

The TSP is classified as an NP-Hard problem, meaning that the computation time required to find the optimal solution increases exponentially with the number of nodes to be visited. In cases involving eight distribution locations, as in this study, a brute-force approach is inefficient. Therefore, a more adaptive, efficient, and near-optimal algorithmic approach is needed to address this challenge realistically. In this research, a deterministic heuristic algorithm is employed—namely, the Path Cheapest Arc method, implemented using the open-source platform Google OR-Tools. This algorithm was chosen for its advantage in solving problems with a relatively small number of nodes, while offering significantly more efficient computation time and practical applicability for real-world scenarios.

The Travelling Salesman Problem (TSP) is classified as an NP-Hard problem, which means that the computation time required to determine the optimal solution grows exponentially with the number of nodes involved. In scenarios involving eight distribution locations, as examined in this study, a brute-force approach becomes impractical. Consequently, a more adaptive, efficient, and near-optimal algorithmic strategy is required to address this challenge realistically. This research adopts a deterministic heuristic algorithm—specifically, the Path Cheapest Arc method—implemented through the open-source platform Google OR-Tools. This algorithm was selected due to its strength in solving problems with a relatively small number of nodes while delivering significantly faster computation times and practical applicability in real-world contexts.

Previous studies have demonstrated the relevance of the TSP across various sectors, ranging from manufacturing industries to public service distribution. A study by Irmansyah & Takaya [3] demonstrated the effectiveness of heuristic algorithms in planning delivery routes for essential goods in densely populated areas. Sumarsa & Widyastiti [4] showed that the application of Google OR-Tools in goods distribution systems can reduce logistics costs by up to 30%. Nurminarsih and Asih et al. [5][6] stated that deterministic strategies for solving the TSP in maritime regions provide significant efficiency in inter-island delivery route planning. Meanwhile, a study by Henri Fachrudin [7] emphasized that combinatorial programming can be effectively utilized in the distribution of social assistance by local governments.

Moreover, Chen et al. [8] developed an interactive visualization method to support transparency and traceability of routes within government logistics systems. A study by Syaputra [1] proposed an adaptive algorithm as a solution for complex archipelagic regions. Wardhana [9] added that logistics distribution planning systems in the public sector would be significantly more effective if combined with heuristic methods tailored to the local context. In addition, the dynamic visualization developed by Solehudin [10] helps decision-makers understand the spatial dynamics of delivery operations. Nazry et al. [11] demonstrated that the use of the Python programming language and the OR-Tools library opens up opportunities for technology adoption in regions with limited computing resources.

Nevertheless, there remains a limited number of studies that specifically combine deterministic heuristic strategies such as the Path Cheapest Arc method with route visualization in the maritime geographical context of Indonesia. This research is significant as it brings together the strengths of graph theory, optimization algorithms, and visualization technology to address real-world challenges in aid distribution across an archipelagic nation. Beyond contributing to the development of academic methodologies, this study also offers a replicable solution model for public logistics systems in other remote regions. The resulting efficiency not only leads to budget savings but also improves delivery time accuracy, operational reliability, and equity in distribution across regions.

By integrating theoretical and practical approaches grounded in technology, this study underscores the importance of adopting computational methods in the logistics decision-making processes of government institutions. The proposed model is expected to serve as a reference for developing more responsive, adaptive, and context-aware decision support systems. Through the synergy between algorithms, spatial data, and public policy, logistics distribution in remote regions can be elevated to a higher level of efficiency and equity.

2. THEORETICAL FOUNDATION

2.1. Fundamentals of graph theory

In both academic and practical domains, graph theory plays a fundamental role in modeling network-related problems particularly in the optimization of logistics delivery [12]. Formally, a graph is defined as an ordered pair:

$$G = (V, E) \tag{1}$$

where:

- V is the set of vertices representing locations, such as warehouses or delivery points.
- E is the set of edges that represent the paths connecting two vertices, which may reflect physical routes or logical relationships between distribution points.

In this context, each type of graph carries specific characteristics that contribute to how routing problems are structured and solved.

2.1.1. Weighted graph

In this graph, each edge$e \in E$ is assigned a weight $w(e)$ which represents the distance, time, or cost between locations. This representation is essential for algorithms such as Dijkstra's, which are designed to find the shortest path between two points [13][14]. As an illustration, imagine a delivery network represented by the following graph:

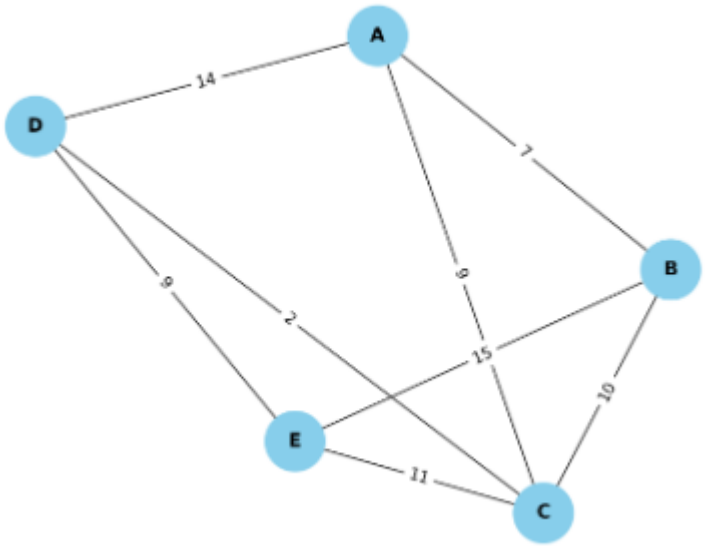


Figure 1. Weighted graph

The graph illustrates delivery locations (vertices) and routes (edges) with weights representing travel distances. This model helps logistics systems assess efficiency and operational costs.

2.1.2. Complete graph

A complete graph is one in which every pair of vertices is connected by a direct edge[15]. In the context of delivery, this means that all locations are assumed to be directly reachable from one another—even though, in practice, not all points are connected in such a manner. However, this approach is often used to simplify initial modeling and to compute near-optimal solutions, particularly in problems such as the Travelling Salesman Problem (TSP).

2.1.3. Directed graph

This type of graph assigns a direction to each edge, which is relevant for deliveries in urban areas with one-way streets or specific traffic regulations. For example, in water distribution systems or city road networks, the direction of flow or travel becomes crucial. Using directed graphs allows us to avoid solutions that may be theoretically correct but impractical to implement in real-world scenarios [16].

By utilizing these various types of graphs, we can model real-world constraints and objectives more realistically. One mathematical representation is the weighted adjacency matrix, where:

- $A[i,j] = w(i,j)$: .if there is an edge from vertex i to vertex j.
- $A[i,j] = \infty$: if there is no edge.
- $A[i,j] = 0$: if i and j are the same vertex.

2.2. Traveling salesman problem (TSP)

The Traveling Salesman Problem (TSP) is one of the most classic and fundamental combinatorial optimization models in operations research [17]. TSP describes an agent (in this context, a logistics vessel) that

must visit a number of locations exactly once, and then return to the starting point, with the goal of minimizing the total travel distance.

Mathematically, given a set of locations $V = \{v_1, v_2 \dots, v_{n\}}$ and the distance between locations $\mathrm{d}(v_{i,}v_j)$, the goal is to find a permutation π of these locations that minimizes the total distance

$$\text{Minimize } \mathrm{D}(\pi) = \sum_{i=1}^{n-1} \mathrm{d}(v_{\pi(i)}, v_{\pi(i+1)}) + (v_{\pi(n)}, v_{\pi(1)}) \quad (2)$$

n the context of delivering basic food supplies to eight islands, the TSP can represent the movement of a vessel starting from the main port, visiting each island exactly once, and returning to the port. The primary objective is route efficiency, aiming to achieve savings in fuel consumption, operational time, and distribution costs.

2.3. Complexity and solution approaches to the TSP

The TSP belongs to the class of NP-Hard problems, meaning that while a solution can be verified quickly, finding the solution requires computational time that increases exponentially with the number of nodes[18]. For a case involving eight locations, as in this study, there are $\frac{(n-1)!}{2} = 2520$ possible routes to evaluate using a brute-force approach. Therefore, an efficient algorithmic approach is required.

2.3.1. Exact approach

Exact methods are designed to find optimal solutions with guaranteed accuracy, making them particularly suitable for small-scale problems such as in this study[19]. Among the commonly used exact methods are:

- Branch and Bound, which works by pruning the search space based on the lower bound of partial solutions;
- Dynamic Programming, which stores the results of subproblems to avoid redundant computations;
- Integer Linear Programming (ILP), which formulates the problem as a system of linear equations with integer constraints.

  For example : $\mathrm{C}(\mathrm{s}, \mathrm{i}) = \min \{C(S - \{j\}, j) + d(j, i)\}$ (3)

2.3.2. Heuristic and metaheuristic approaches

For large-scale cases, heuristic approaches are employed as alternatives to obtain near-optimal solutions within a more efficient time frame. Some popular techniques include:

- Nearest Neighbor, which locally selects the next closest point;
- 2-opt dan 3-opt, which improve routes by performing edge swaps;
- Simulated Annealing, which mimics the metal cooling process to avoid local optima.

However, in the context of eight locations, exact approaches remain highly relevant, as the complexity is still within reasonable bounds for modern computing. Nevertheless, in this implementation, a deterministic heuristic strategy Path Cheapest Arc from Google OR-Tools—is used due to its simplicity and rapid convergence, which makes it highly suitable for real-time applications [20]. This approach enables fast and efficient solutions without significantly compromising solution quality for problems of relatively small scale.

2.4. Simulation and visualization of optimal routes using python

The use of the Python programming language in this study aims to build a structured simulation for solving the TSP. Python offers a wide range of libraries and algorithmic approaches that are flexible for both exact and heuristic methods. The modeling process in Python includes:

- Representing locations and the distance matrix as the main input,
- Selecting the appropriate TSP solving method based on the scale of the problem,
- Visualizing route paths to facilitate intuitive result analysis and support operational interpretation in logistics.

Through this simulation, the outcomes produced are not limited to numerical outputs, but also include route maps that can be interpreted by decision-makers to determine real-world delivery scenarios [21].

2.5. Relevance of TSP in government logistics systems

The distribution of basic food assistance (*sembako*) by local governments is a tangible example of how mathematical approaches can provide solutions to social and logistical challenges. In Indonesia's archipelagic geographical context, route planning based on the TSP model holds strategic value. It helps the government to:

- Reduce the risk of budget and fuel waste,
- Improve the timeliness of aid deliveries,
- Provide a foundation for data-driven and scientifically grounded decision-making.

General Mathematical Model

$$\min \sum_{i=1}^{n} \quad \sum_{j=1}^{n} c_{ij}.\, i_j \tag{4}$$

Subject to the following constraints:

1. Each location must be visited exactly once:

$$\sum_{i=1}^{n} x_{ij} = 1, \forall j \tag{5}$$

2. Each location must be departed from exactly once:

$$\sum_{i=1}^{n} x_{ij} = 1, \forall i \tag{6}$$

3. Subtour elimination (partial cycle prevention)

$$u_i - u_j + n.\, x_{ij} \leq n - 1 \tag{7}$$

where:

- c_{ij} represents the distance, time, or cost from point i to j,
- $x_{ij} \in \{0,1\}$ is a binary decision variable,
- u_{ij} is the position of point i in the delivery sequence.

This approach is not only applicable in academic contexts, but also adaptive to the complexity of terrain and resource constraints encountered in the field. The efficiency gained through the application of the TSP provides a concrete example of how algorithms and graph theory can bridge the gap between scientific knowledge and societal needs.

3. IMPLEMENTATION

3.1. Theoretical Approach to the TSP

In the realm of combinatorial optimization, the Travelling Salesman Problem (TSP) holds a central position as one of the classical problems that continues to draw attention across various scientific disciplines. Formally, the problem is modeled as a complete weighted graph denoted by $G = (V, E)$, whare:

- The set of vertices V represents delivery points, including a designated depot node that serves as both the starting and ending point of the logistics route.
- The set of edges E connects every pair of vertices, with weights representing the Euclidean distances between locations.

The primary goal of the TSP is to find a Hamiltonian cycle a closed loop that visits each vertex exactly once with the minimum total weight [15]. This weighted graph representation enables the problem to be mathematically formulated and solved using structured algorithmic approaches.

For small-scale instances, exact methods such as brute-force, dynamic programming, and integer linear programming can be employed to obtain an optimal solution [22]. However, due to its factorial complexity $(O(n!))$, such approaches become impractical for medium to large-scale datasets.

Although exact methods remain applicable for a limited number of locations, this implementation adopts a deterministic heuristic strategy Path Cheapest Arc from Google OR-Tools—due to its simplicity and fast convergence, making it highly suitable for real-time applications. This strategy enables the search for near-optimal solutions with significantly more efficient computation time, without compromising the interpretability of the results in an operational logistics context.

3.2. Practical Approach: Code-Based Implementation

3.2.1. Data preparation

Destination islands for aid distribution. Each point is defined by its geographical coordinates (latitude and longitude) as well as the maritime distances between locations, measured in nautical kilometers. These distances were calculated using the Haversine formula based on actual geographical coordinates obtained from Google Maps and validated by the Geospatial Information Agency (BIG). The distances are presented in the form of an 8×8 distance matrix.

The data structure includes the following elements:

- Location Names and Island Representation: P0: Depot – Tanjung Saronggi Port (main port). P1: Giligenting Island, P2: Giliraja Island, P3: Raas Island, P4: Kangean Island, P5: Masalembu Island, P6: Pajangan Island, P7: Talango Air Island.
- Distance Matrix: The inter-location distances, expressed in nautical kilometers, encompass both outbound and return routes to and from the depot.
- Visualization Coordinates: The projected 2D coordinates of each location were utilized for visualization in Python via the Matplotlib library. These coordinates do not reflect the actual geographic positions; rather, they have been scaled to enhance graphical clarity.

The dataset structure serves as the foundation for constructing a complete weighted graph, representing the Travelling Salesman Problem (TSP). In this model, each pair of locations is interconnected by a weighted edge corresponding to the maritime distance between them, facilitating the resolution of the TSP to identify the optimal distribution route. Visualization of the dataset structure and location coordinates:

```
# ===== 1. DATA KOORDINAT ASLI =====
# Format: (kode, nama, lat, lon)
node_data = [
    ("P0", "Depot (Pelabuhan Tanjung Saronggi)", -7.125302210656127, 113.89395017190174),
    ("P1", "Pulau Giligenting", -7.178925206677771, 113.89796527013415),
    ("P2", "Pulau Giliraja", -7.208431439583993, 113.79984960909061),
    ("P3", "Pulau Raas", -7.133578069102832, 114.49610214090914),
    ("P4", "Pulau Kangean", -6.841079583611733, 115.22817872105686),
    ("P5", "Pulau Masalembu", -5.576763740396142, 114.41472537071537),
    ("P6", "Pulau Pajangan", -6.9741303288263214, 114.42854609531491),
    ("P7", "Pulau Talango Air", -7.060906519480061, 114.64901056132487),
]

location_names = [n[0] for n in node_data]  # Kode titik
```

Figure 2. Direct distance matrix between delivery locations based on case data

Figure 2 illustrates the data structure comprising eight distribution points used in this study. Each row contains the location code, island name, and the latitude and longitude coordinates obtained from the actual geographical conditions. This data forms the basis for calculating the inter-location distances using the Haversine formula and is applied within the Travelling Salesman Problem model to determine the most efficient route for distributing basic commodities.The coordinates are organized as follows for the purpose of 2D visualization:

```
# ===== 2. FUNGSI HITUNG JARAK (HAVERSINE) =====
def haversine(lat1, lon1, lat2, lon2):
    R = 6371.0  # radius bumi dalam km
    phi1, phi2 = math.radians(lat1), math.radians(lat2)
    dphi = math.radians(lat2 - lat1)
    dlmb = math.radians(lon2 - lon1)
    a = math.sin(dphi / 2) ** 2 + math.cos(phi1) * math.cos(phi2) * math.sin(dlmb / 2) ** 2
    return 2 * R * math.asin(math.sqrt(a))

# ===== 3. BANGUN MATRICS JARAK =====
coords_latlon = [(n[2], n[3]) for n in node_data]  # list (lat, lon)
distance_matrix = []
for i in range(len(coords_latlon)):
    row = []
    for j in range(len(coords_latlon)):
        row.append(round(haversine(coords_latlon[i][0], coords_latlon[i][1],
                                   coords_latlon[j][0], coords_latlon[j][1]), 2))
    distance_matrix.append(row)

# ===== 4. KOORDINAT UNTUK VISUALISASI (x=lon, y=lat) =====
coords = np.array([[n[3], n[2]] for n in node_data])
```

Figure 3. Simplified 2D coordinates for visualizing the relative positions of locations

Figure 3 presents three main components in the distance data processing of this study. First, the Haversine function is employed to calculate the shortest distance between two points on the Earth's surface based on latitude and longitude coordinates, taking into account the Earth's curvature to ensure accurate results in kilometers. Second, the construction of the distance matrix processes all coordinates from node_data and generates an 8×8 matrix containing the distances between each pair of locations, including the return distance to the starting point. Third, the visualization coordinates convert latitude–longitude data into an (x, y) format, ready for use in maps or graphs, solely for route display purposes and not for distance calculation. Accordingly, the code in Figure 3 serves as a bridge between the raw coordinate data and the distance information, which is then utilized for modeling the Travelling Salesman Problem and visualizing the distribution routes.

3.2.2. Route optimization

The Path Cheapest Arc algorithm is used to obtain a near-optimal solution for the Travelling Salesman Problem (TSP). The process begins at the starting node (depot) and iteratively selects the edge (arc) with the smallest weight that connects to the nearest unvisited node. This selection continues until all nodes have been

visited exactly once. Once all locations have been reached, the algorithm closes the route by returning to the starting node, thus forming a complete cycle. This strategy is known for its speed in generating an initial solution, although it is not guaranteed to be globally optimal, and is often employed as a starting point before applying more advanced optimization methods. Code Implementation:

```python
# ===== 5. IMPLEMENTASI SOLVER TSP =====
def solve_tsp():
    manager = pywrapcp.RoutingIndexManager(len(distance_matrix), 1, 0)  # 1 kendaraan, start di 0
    routing = pywrapcp.RoutingModel(manager)

    def distance_callback(from_index, to_index):
        return int(distance_matrix[manager.IndexToNode(from_index)][manager.IndexToNode(to_index)] * 1000)  # meter -> int

    transit_callback_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

    search_params = pywrapcp.DefaultRoutingSearchParameters()
    search_params.first_solution_strategy = routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC

    solution = routing.SolveWithParameters(search_params)
    tsp_route = []
    if solution:
        index = routing.Start(0)
        while not routing.IsEnd(index):
            tsp_route.append(manager.IndexToNode(index))
            index = solution.Value(routing.NextVar(index))
        tsp_route.append(manager.IndexToNode(index))
    return tsp_route

tsp_route = solve_tsp()
```

Figure 4. Python code implementation of the Path Cheapest Arc algorithm

This approach utilizes the Path Cheapest Arc strategy as an initial solution method, where the system iteratively selects the edge with the lowest weight from the currently visited node. This approach is deterministic and well-suited for small to medium-scale scenarios, such as the one presented in this study.

3.2.3. Interactive visualization in the form of animation

One of the advantages of this implementation is the visualization of the optimized route in the form of an interactive animation. The visualization is built using the `matplotlib.animation` library with a frame-by-frame rendering approach, where each step of the vessel's journey is displayed incrementally [10].

Objectives of the Visualization:

- To intuitively convey the sequence of visits made by the vessel.
- To assist policymakers in visually understanding distribution patterns.
- To link computational results with spatial graphic representation.

Stages in the Animation Process:

- Initialization: Display all location points with labels.
- Frame Update: Each frame adds one travel route to the graph..
- Finalization: Once all locations have been visited, the return route to the depot is drawn.

Animation Code Snippet:

```python
# ===== 6. VISUALISASI DENGAN ANIMASI =====
fig, ax = plt.subplots(figsize=(8, 6))
colors = 'tab:blue'
def init():
    ax.clear()
    ax.set_title("Visualisasi Rute TSP - Distribusi Sembako")
    ax.set_xlabel("Longitude")
    ax.set_ylabel("Latitude")
    for i, (x, y) in enumerate(coords):
        ax.text(x, y + 0.03, location_names[i], ha='center', fontsize=9)
        ax.plot(x, y, 'o', color='tab:gray')
    return []
def update(frame):
    init()
    path = tsp_route[:min(frame + 2, len(tsp_route))]
    xs = [coords[i][0] for i in path]
    ys = [coords[i][1] for i in path]
    ax.plot(xs, ys, '-o', color=colors)
    return []
ani = FuncAnimation(fig, update, frames=len(tsp_route), init_func=init,
                    blit=False, interval=800, repeat=False)
plt.tight_layout()
plt.show()
```

Figure 5. Animated visualization of the TSP optimization route for basic food distribution

With this animation, readers are not only able to understand the results in numerical form, but can also visually observe the process of route construction—from the initial departure at the port to the completion of all visits and the return to the starting point.

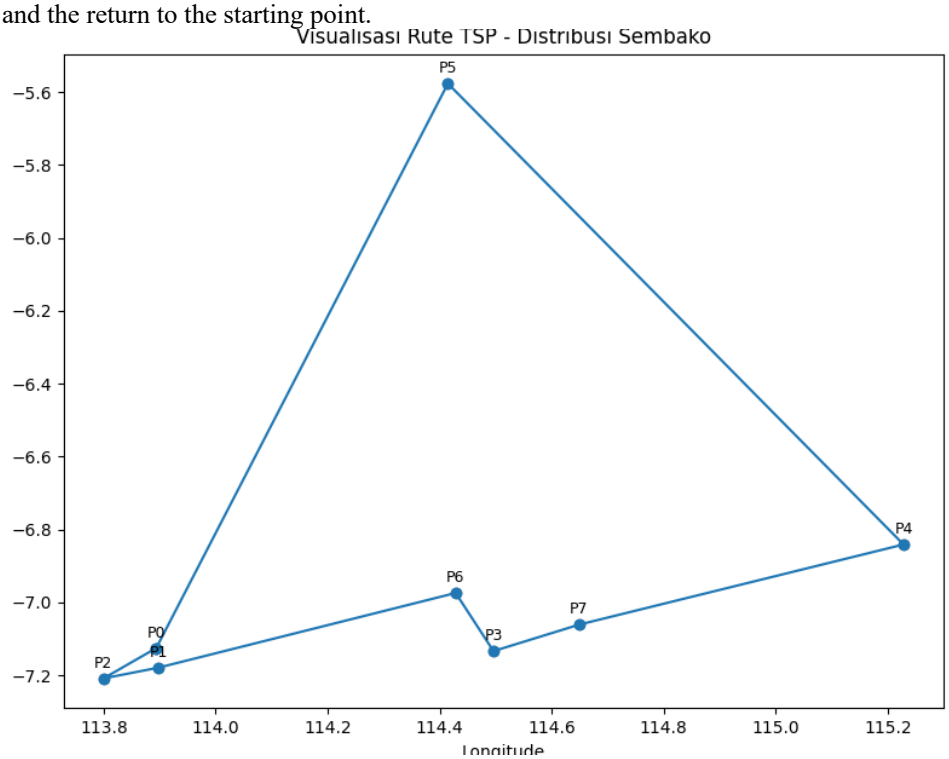


Figure 6. Route visualization generated by solving the Travelling Salesman Problem (TSP) for optimizing delivery paths with one logistics ship

3.2.4. Result visualization

The route optimization process in this study concludes with the creation of an interactive visualization using the matplotlib.animation library. This visualization depicts the movement of the vessel from the starting point to each island in the sequence determined by the optimized TSP route. Each frame in the animation represents a single travel step, where the connecting lines between points are updated progressively until the complete route is formed.

The coordinate data used in the visualization are actual data obtained from Google Maps for each island location. Accordingly, the calculated distances (256.7 km for the initial route and 198.4 km for the optimized route) reflect the real geographical conditions.

From the visualization, it is evident that the optimized route is more efficient, achieving a distance reduction of 22.7% compared to the initial route. Assuming a vessel speed of 25 km/h and a fuel consumption rate of 0.4 liters/km, the estimated travel time decreases from 10.27 hours to 7.94 hours, and fuel consumption decreases from 102.7 liters to 79.4 liters. It should be noted that these estimates are based on general assumptions, as the actual operational parameters of the vessel have not yet been obtained from field measurements.

4. CONCLUSION

This study demonstrates that logistics route planning based on the Travelling Salesman Problem (TSP) is an effective and measurable approach to addressing the challenges of inter-island aid distribution. By modeling delivery locations as a complete weighted graph and applying route-finding algorithms based on constraint programming, optimal solutions can be achieved within efficient computational time. The resulting animated visualization further enhances understanding of the route patterns and provides intuitive interpretability for policymakers. The optimization results show a significant reduction in total travel distance, which implicitly contributes to savings in fuel consumption, operational time, and distribution costs. These findings affirm that approaches grounded in graph theory and optimization algorithms are not only academically relevant but also practically applicable in public service contexts-particularly in geographically complex regions such as archipelagos. Thus, this research not only offers a technical solution to logistical problems but also emphasizes the integration of computational science with real societal needs. The broader implications of this study point

toward opportunities for developing data-driven logistics decision support systems that can be replicated in similar distribution scenarios in other remote regions.

REFERENCES

[1] P. E. Syaputra, "Kajian Integrasi Transportasi Multi Moda Untuk Menekan Biaya Logistik Pada Wilayah Kepulauan: Studi Kasus Pada Pulau Bawean," *J. Transp.*, vol. 24, no. 1, pp. 49–61, 2024.

[2] E. Nurjati, "Peran dan tantangan e-commerce sebagai media akselerasi manajemen rantai nilai produk pertanian," in *Forum Penelitian Agro Ekonomi*, 2021, pp. 115–133.

[3] M. I. Irmansyah and R. Takaya, "Optimalisasi Pengiriman Last-Mile Di Rantai Pasokan Perkotaan Menggunakan Algoritma Heuristik," *J. Ilm. Multidisiplin Terpadu*, vol. 8, no. 7, 2024.

[4] A. Sumarsa and M. Widyastiti, "Penerapan Traveling Salesman Problem with Time Windows dalam Pendistribusian Produk," in *Prosiding Seminar Nasional Pendidikan Matematika*, 2022, pp. 1–6.

[5] S. Nurminarsih, "Pengembangan Model dan Algoritma Dynamic-Inventory Ship Routing Problem (D-ISRP) dengan Mempertimbangkan Tingkat Kesibukan Pelabuhan." Surabaya: Institut Teknologi Sepuluh Nopember, 2015.

[6] H. M. Asih, R. A. C. Leuveano, A. Rahman, and M. Faishal, "Traveling Salesman Problem With Prioritization for Perishable Products in Yogyakarta, Indonesia," *J. Adv. Manuf. Technol.*, vol. 16, no. 3, 2022.

[7] H. Fachrudin, "Optimasi Penentuan Rute Perjalanan Sales pada UD. Aster." Universitas Islam Majapahit, 2019.

[8] C. Chen *et al.*, "Perbandingan Implementasi Evolutionary Algorithm (EPO, FHO, dan CFA) pada Kasus Travelling Salesman Problem untuk Tempat Pariwisata di Surabaya," *INSYST J. Intell. Syst. Comput.*, vol. 5, no. 1, pp. 23–38, 2023.

[9] I. N. Wardhana, "Optimasi Rute Distribusi Gas Transport Module (Gtm) Menggunakan Vehicle Routing Problem (Vrp)." Institut Teknologi Sepuluh Nopember, 2018.

[10] R. H. Solehudin, *Business Intelligence dan Analisis Kuantitatif-Damera Press*. Damera Press, 2025.

[11] H. W. N. S. Nazry, F. Riza, F. Rizky, Z. A. Gultom, M. Haris, and M. D. B. Barus, "Model Optimasi Model Optimasi Rute Transportasi Berbasis Pemrograman Linear," *J. Sist. Inf. Triguna Dharma (JURSI TGD)*, vol. 4, no. 1, pp. 75–81, 2025.

[12] A. P. Peranginangin, "Analysis of Graph Theory in Transport Network Optimisation : A Mathematical Approach and Its Applications," vol. 10, no. 4, pp. 1431–1439, 2025.

[13] A. Amin and B. Hendrik, "Analisis Penerapan Algoritma Dijkstra dalam Optimasi Penentuan Rute: Sebuah Kajian Literatur Sistematis," *J. Educ. Res.*, vol. 6, no. 1, pp. 100–106, 2025.

[14] J. Ahmad and M. Nadhir Ab Wahab, "Enhancing the safety and smoothness of path planning through an integration of Dijkstra's algorithm and piecewise cubic Bezier optimization," *Expert Syst. Appl.*, vol. 289, p. 128315, 2025, doi: https://doi.org/10.1016/j.eswa.2025.128315.

[15] M. Iqbal, A. Damayanti, N. T. Maulani, N. L. P. Wardhani, and M. G. Rahman, "Implementasi Graf Hamilton pada Travelling Salesman Problem dari Kantor Walikota ke Setiap Kantor Kecamatan di Kota Salatiga," *Realis. J. Educ. Math. Sci.*, vol. 2, no. 2, pp. 84–91, 2024.

[16] R. Nasiboglu, "Dijkstra solution algorithm considering fuzzy accessibility degree for patch optimization problem," *Appl. Soft Comput.*, vol. 130, p. 109674, 2022, doi: https://doi.org/10.1016/j.asoc.2022.109674.

[17] W. Y. Oeitama, W. Y. Oeitama, F. F. Sitandi, and S. Mas' ud, "Optimasi Rute Pendistribusian Barang Menggunakan Kombinasi Algoritma Branch and Bound dan Cheapest Insertion Heuristic," *Sq. J. Math. Math. Educ.*, vol. 6, no. 2, pp. 89–104, 2024.

[18] C. R. Gunawan, "Optimization of the Travelling Salesman Problem Based on Genetic Algorithm with Adaptive Crossover and Mutation Probabilitie," *J. Ilmu Komput. dan Sist. Inf.*, vol. 3, no. 3, pp. 234–242, 2024.

[19] A. Y. Nugroho, A. Suyitno, and R. Arifudin, "Perbandingan Algoritma Branch And Bound Dan Algoritma Genetika Untuk Mengatasi Travelling Salesman Problem (Tsp)(Studi Kasus Pt. Jne Semarang)," *UNNES J. Math.*, vol. 5, no. 2, pp. 135–143, 2016.

[20] R. Ramadhan, L. R. Kristiana, and H. K. Pambudi, "Perancangan Rute Distribusi Produk Pestisida Dengan Menggunakan Software or Tools Untuk Meminimasi Jarak Tempuh Dan Biaya Transportasi Di PT ABC," *eProceedings Eng.*, vol. 12, no. 2, pp. 3223–3230, 2025.

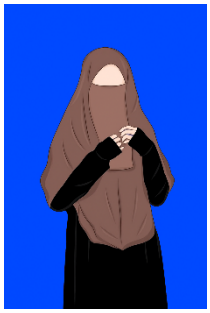
[21] T. D. Arsyad, M. D. C. Nababan, R. K. Imburi, and P. Harliana, "Implementasi Algoritma Dijkstra dalam Mencari Rute Terpendek dari Universitas Negeri Medan ke Museum Negeri Sumatera Utara," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 9, no. 1, pp. 235–242, 2025.

[22] D. B. Paillin and J. M. Tupan, "Model Integer Liniear Programming (ILP) dalam Pemecahan Traveling Salesman Problem (TSP)(Studi Kasus: PT. Paris Jaya Mandiri Ambon)," *ALE Proceeding*, vol. 3, pp. 40–47, 2020.

BIOGRAPHIES OF AUTHORS

Moh Jufri is a fourth-semester undergraduate student in the Informatics Engineering program at Universitas Islam Madura, Indonesia. He is actively involved in research activities in the field of computer science, with particular interests in software engineering, data analysis, and computational optimization. Jufri is highly passionate about developing practical technology-based solutions to address real-world challenges. His academic background and research involvement reflect a strong commitment to advancing digital innovation. He can be contacted via email at: mohjufri129@gmail.com

Nasrullah is a fourth-semester undergraduate student in the Informatics Engineering program at Universitas Islam Madura, Indonesia. He is actively involved in academic projects and mobile app development, with particular interests in React Native, software engineering, and user interface design. Nasrullah is highly passionate about building creative and efficient digital solutions to solve real-world problems. His technical experience, combined with a solid academic foundation, reflects a strong commitment to continuous learning and digital innovation. He can be contacted via email at: nasrullah123@gmail.com



Aisyatir Rodiyah is a fourth-semester student in the Informatics Engineering program at Universitas Islam Madura, Indonesia. Her academic interests include software engineering, data analysis, and computational optimization. She has been involved in student research and has experience using Python and supporting libraries to develop simple, efficient systems. Aisyatir frequently attends seminars and training programs to enhance her technical skills. She believes technology should create real impact, guiding her research toward practical and beneficial outcomes. She can be contacted via email: sa0595363@gmail.com