Solving Asymmetric Clustered Travelling Salesman Problem With Ant Colony Optimization in the Surabaya Gas Station Control Case

Abstract— **Effective management of distribution activities is crucial in supply chain management, as distribution costs influence product pricing. To reduce these costs, optimizing distribution routes is essential. This paper addresses the Asymmetric Clustered Travelling Salesman Problem (ACTSP), a TSP variant where cities are grouped into clusters, and visits must follow a sequential order within each cluster, with asymmetric travel costs. The Ant Colony Optimization (ACO) algorithm, inspired by ant foraging behavior, is applied to solve the ACTSP. We demonstrate its application in optimizing the control routes for gas stations in Surab**

Keywords— Ant Colony Optimization, Asymmetric Clustered Travelling Salesman Problem, Distribution, Opposition-Based Learning.

I. Introduction

In today's competitive markets, effective supply chain management is crucial for companies to reduce operational costs and maximize service value[1]. Distribution planning significantly impacts the final selling price of products, with costs comprising fixed and variable components such as fuel expenses and travel time [2]. Efficient distribution route planning is essential to optimize these costs [3].The Travelling Salesman Problem (TSP), a well-known combinatorial optimization problem, requires finding the shortest route for a salesperson to visit multiple cities. Krishna et al. [4] proposed the Spotted Hyena-based Rider Optimization Algorithm (S-ROA) to solve TSP. Arigliano et al. [5] suggested a branch-and-bound algorithm for the Asymmetric Travelling Salesman Problem (ATSP), iteratively selecting subproblems and calculating lower bounds until the optimal solution is found. Fuentes et al. [6] recommended clustering before solving TSP, followed by the NEH heuristic to obtain an initial solution, refined using Multi-Restart Iterated Local Search (MRSILS). Strodola [7] proposed the ant colony algorithm for TSP, demonstrating its effectiveness in applications like vehicle routing and logistics planning. Putra [8] addressed the Asymmetric Clustered Travelling Salesman Problem (ACTSP) using the firefly algorithm for gas station control in Surabaya.

Despite the prevalence of ACTSP, it remains under-researched. This problem typically occurs when route distances differ for outbound and return journeys between cities. Addressing ACTSP can significantly aid organizations in determining the shortest distribution routes. A pertinent application is route planning for inspecting and controlling gas stations in Surabaya, aimed at preventing fraud and protecting the public. This task involves officials visiting each gas station and performing inspections as per established procedures. The officials' routes can be modeled as ACTSP, with clusters representing different regions in Surabaya.

This study proposes using the Ant Colony Optimization (ACO) algorithm to solve the ACTSP model. ACO is a metaheuristic method that simulates the foraging behavior of ant colonies to find the shortest path to food. Previous research has shown that ACO outperforms other methods for TSP, making it suitable for ACTSP implementation. The algorithm will be applied to a case study involving route planning for gas station inspections in Surabaya.

II. Literature review

A. *Case Study: Gas Station Control Route Optimization*

Gas station control involves ensuring the accuracy of fuel dispensing units at various stations, which is mandated by Indonesian Law No. 2 of 1981 concerning Legal Metrology. Legal metrology is the science related to the accuracy of measurements concerning regulations based on laws aimed at protecting the public interest. Through this oversight, the government seeks to prevent actions that could harm consumers by gas station operators.

In gas station control, the officials visit gas stations repeatedly to test fuel dispensers [9]. Currently, the officials only use Google Maps to determine their visitation routes to each gas station. Additionally, there is lack of research on optimizing routes for officials in conducting gas station control activities.

B. *Travelling Salesman Problem*

The Travelling Salesman Problem (TSP) is a widely studied optimization challenge where a salesperson aims to find the shortest route visiting each city exactly once before returning to the starting point [4]. Traditionally, solving TSP involved brute-force methods, which explore all possible routes, making it costly for larger problems. TSP provides a more efficient approach, minimizing costs by identifying optimal routes [10]. Classified as NP-hard due to its exponential solution complexity with increasing cities [11], TSP finds applications in computer networking, vehicle routing, data clustering, and circuit board drilling. Recent advancements integrate artificial intelligence techniques like reinforcement learning, genetic algorithms, and ant colony optimization [11].

Mathematically, TSP is represented by a graph $\boldsymbol{G} = (\boldsymbol{V},\boldsymbol{A})$, where $\boldsymbol{V}$ represents cities as vertices and $\boldsymbol{A}$ represents edges connecting city pairs with associated costs $\boldsymbol{c_{ij}} \in \mathbb{R}^{+}$. Basic TSP assumes symmetric travel costs between cities $\boldsymbol{c_{ij}} = \boldsymbol{c_{ji}}$ [12]. Evolving from this, the asymmetric TSP (ATSP) considers varying travel costs in different directions [12]. This final project applies ATSP principles to optimize gas station control routes in Surabaya.

C. *Clustered Travelling Salesman Problem*

Define TSP has several variants, one of which is the Clustered Travelling Salesman Problem (CTSP). CTSP was first introduced by Chisman in 1975. In CTSP, nodes are divided into multiple clusters, and all nodes within the same cluster must be visited sequentially. However, there is no priority among clusters, and they can be visited in any order [13].

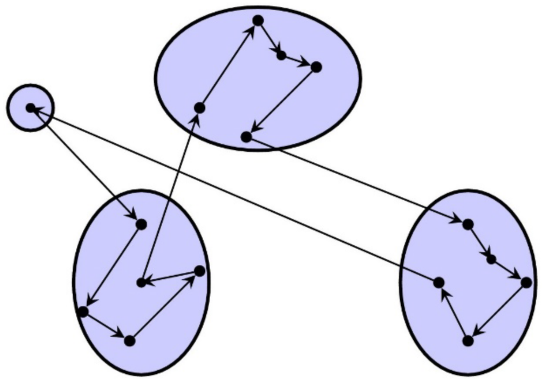


Fig. 1. Example of a CTSP Solution with 16 nodes and 4 clusters.

D. *Asymmetric Travelling Salesman Problem*

The Asymmetric Travelling Salesman Problem (ATSP) is a combinatorial problem involving a specific list of locations with different travel times and distances between locations for outbound and inbound routes. The goal of ATSP is to find a tour that visits each location while minimizing travel time and distance between locations [16].

The formulation of the Asymmetric Travelling Salesman Problem (ATSP) is as follows:

$$\text{Minimize} \sum_{i \in N} \sum_{j \in N, j \neq i} c_{ij} x_{ij} \tag{1}$$

Batasan :

$$\sum_{i \in N, j \neq i} x_{ij} = 1, \forall j \in N \tag{2}$$

$$\sum_{j \in N, j \neq i} x_{ij} = 1, \forall i \in N \tag{3}$$

$$\sum_{i \in S} \sum_{j \in \mathcal{N} \setminus [S\}} x_{ij} \geq 1, \forall S \subset N, |S| \geq 2 \tag{4}$$

$$\sum_{i \in S} \sum_{j \in S, j \neq i} x_{ij} \leq |S| - 1, \forall S \subset N, |S| \geq 2 \tag{5}$$

$$x_{ij} \in \{0,1\}, \forall i,j \in N, j \neq i. \tag{6}$$

The objective function (1) minimizes the total distance traveled. Constraints (2) and (3) ensure that each node is visited and departed exactly once. Constraints (4) and (5) are used to eliminate subtours, ensuring that the tour starts and ends at city $|S|$ and visits exactly $|S| - 1$ cities. Constraint (6) represents the domain of variable [10].

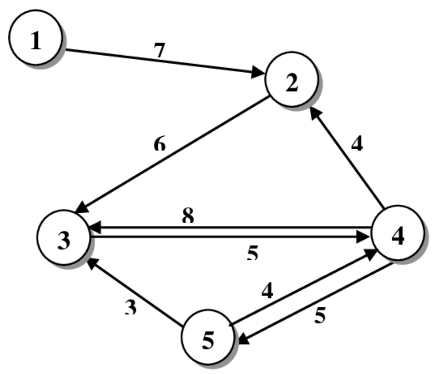


Fig. 2. Example of an ATSP solution with different weight between going and returning at node 3, 4 and 5.

E. *Ant Colony Optimization*

The Ant Colony Optimization (ACO) is an algorithm inspired by the natural behavior of ants searching for food. Since its introduction by Dorigo in 1991 and its first application to solving the TSP in 1997 [7], ACO has proven to be effective in solving various optimization problems.

ACO operates through two basic steps: route construction and pheromone update. In the first step, solutions are determined based on random proportional rule. m ants are randomly placed in n cities. In iteration t, the transition probability for ant k to travel from city i to j, termed as state transition probability, can be formulated as follows [18] as in:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum_{s \in T_k(i)} [\tau_{is}(t)]^{\alpha}[\eta_{is}(t)]^{\beta}}, & \text{if } j \in J_k(i) \\ 0, & \text{reversed} \end{cases} \tag{7}$$

τ_{ij} represents the pheromone trail and η_{ij} denotes the heuristic information. α and β are parameters that determine their respective influences. Typically, $\eta_{ij} = 1/d_{ij}$ where d_{ij} is the distance between cities (i,j). $J_k(i)$ denotes the feasible neighborhood for ant k at city i.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{8}$$

$\rho (0 < \rho \leq 1)$ is the rate of evaporation, $\Delta\tau_{ij}^{k}$ representing the pheromones remaining in the path (i,j) by the k-ants. $\Delta\tau_{ij}^{k}$ can be determined by:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k}, & \text{if ant k pass path } (i,j) \\ 0, & \text{reversed} \end{cases} \tag{9}$$

Q is the coefficient of increase in pheromones and .L_k is the total length of the ant's path k

F. *Opposition-Based Learning*

Opposition-Based Learning (OBL) was initially developed in 2005 by Tizhoosh [19]. OBL has proven to be a way to humanize optimal meta-heuristic search. This technique involves concurrently estimating opposite agent pairs to enhance the chances of finding agents with better fitness values. OBL is used to evaluate both the current solution and its opposite. Among these solutions, the optimal one is selected to enhance search capabilities.

Solution of TSP is represented as a sequence of city indices. For instance, a path through nnn cities is formulated as:

$$P = [1,2,\cdots,n] \tag{10}$$

P represents the order of cities visited by a salesman. Additionally, the clockwise (CW) opposite path can be formulated as :

$$P^{\mathrm{CW}} = \left[1,1+\frac{n}{2},2,2+\frac{n}{2},\cdots,\frac{n}{2}-1,n-1,\frac{n}{2},n\right] \\ ,when\ n\ is\ even \tag{11}$$

This equation defines the CW path for even n. For odd n, an additional city can be appended at the end of the reversed path. Once the opposite path is determined, the additional city can be removed.

G. Opposition-Based Ant Colony Optimization

Opposition-Based Ant Colony Optimization (OBL-ACO) is an optimization method that combines the ACO algorithm with OBL. In its application, there are three types of frameworks that can be used [18]:

1) *ACO-Index*

The initial step in ACO involves selecting the closest according to the rule of state transition. The reverse path is always longer than the original path, making it unable to update pheromones. To address this issue, ACO-Index is proposed based on a modified strategy for constructing reverse paths. The construction of reverse paths is divided into two steps: path selection and reverse path decision. If the reverse path for different paths but with similar cycle routes is found, these paths are considered the same [18].

2) *ACO-MaxIt*

Zhang et al. proposed an enhancement to the ACO method named ACO-MaxIt [18]. The mirroring point M is formulated as follows:

$$M = \left\lceil \frac{1+n}{2} \right\rceil \tag{12}$$

If n is odd, the mirrored city($\tilde{C}$) from the current city (C) can be defined as:

$$\tilde{C} = \begin{cases} C, & \text{if } C = M \\ C + M, & \text{if } C < M \\ C - M, & \text{if } C > M \end{cases} \tag{13}$$

If n is even, the mirrored city($\tilde{C}$) from the current city (C) can be defined as:

$$\tilde{C} = \begin{cases} C, & \text{if } C = n/2 \text{ or } (n/2+1) \\ C + M, & \text{else if } C < M \\ C - M, & \text{else if } C > M \end{cases} \tag{14}$$

3) *ACO-Rand*

In the previous two methods, the update of pheromones was decided based on predetermined time-related experience for calculating the reverse paths. In ACO-Rand, the time set for updating pheromones is determined by two variables, R_0 and R. R_0 is randomly chosen but fixed once established, while R is randomly selected during each iteration.

H. Performance Evaluation

To evaluate the performance of the algorithm used, performance evaluation is necessary to demonstrate the quality of the algorithm. Performance evaluation can be conducted by calculating the average error to minimize its value [20].

$$\textbf{\textit{Average error}} = \frac{\textbf{\textit{Average result}} - \textbf{\textit{optimum}}}{\textbf{\textit{optimum}}} \times \mathbf{100\%} \tag{15}$$

III. Methodology

A. Literature Study

The initial stage of this thesis is literature review, where the researcher gathers and comprehends various references related to the research object, which is distribution route planning in supply chain management. This is followed by a literature review focusing on the main issues in this research, including TSPLIB data, optimization cases for gas station control, Travelling Salesman Problem (TSP), Clustered Travelling Salesman Problem (CTSP), Asymmetric Travelling Salesman Problem (ATSP), Ant Colony Optimization (ACO), Opposition-Based Learning (OBL), Opposition-Based Ant Colony Optimization (OBL-ACO), evaluation of route performance, and selected case studies.

B. Data Collection

The data sources used in this thesis include TSPLIB data published by Gerhard Reinelt. This data is openly accessible on its official website and contains various datasets related to TSP and other optimization problems from various sources and types. Additionally, data is collected for gas station (SPBU) and Pertamina office locations in Surabaya using Google Maps.

C. Data Preprocessing

The data preprocessing stage involves transforming the TSPLIB dataset so that it can be used as input for OBL-ACO. Additionally, raw data from SPBU and Pertamina office locations is processed into usable formats.

D. Method Implementation

This stage focuses on implementing the OBL-ACO method on the TSPLIB dataset. It includes adjusting parameters and other configurations to prepare OBL-ACO for use in optimizing routes for ACTSP.

E. Method Evaluation

The purpose of this stage is to assess the performance of the developed method. This is done by comparing the route costs obtained through experiments with known route costs from previous research.

F. *Method Implementation in Case Study*

In this stage, the method developed and tested on previous datasets is implemented to determine optimal routes in the case of gas station control by technicians.

IV. Result and Discussion

A. *Equal-Size Spectral Clustering Design*

1) *Initial Cluster Initialization*

In the first stage, clusters are initialized using the Spectral Clustering function. To assess cluster distribution, the standard deviation of distances between points within the same cluster is calculated. The results of this clustering will serve as a reference to balance the number of cluster members.

2) *Calculation of Neighbors for Each Cluster*

Following the cluster initialization process, the next step is to calculate the neighbors for each cluster. This is done by determining the mode of the cluster labels of the nearest neighbors for each data point. Calculating the neighbors for each cluster is crucial as cluster balancing is achieved by exchanging points between neighboring clusters.

3) *Balancing the Number of Cluster Members*

Based on the input values for the minimum and maximum number of cluster members, the program will determine the optimal cluster size. Clusters that exceed or fall short of the optimal size will be balanced by transferring points based on the nearest distance to neighboring clusters. Clusters exceeding the optimal member count will release members, while clusters below the optimal count will receive members from neighboring clusters.

B. *Model Design*

This section describes the problem model to be solved. Before designing the algorithm, it is essential to define several constraints or limitations specific to the gas station (SPBU) case. To enhance the algorithm's results, the SPBU data will be clustered first.

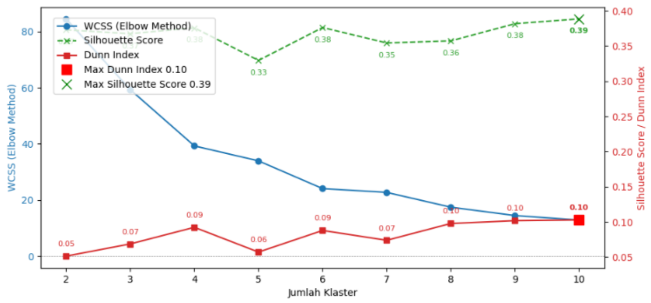


Fig. 3. Result of cluster number finding with 3 method.

Based on Figure 2, testing with the Elbow Method shows a plateauing interval starting at 8 clusters. In contrast, the Silhouette Score and Dunn Index indicate the highest value at 10 clusters. Therefore, the optimal cluster division for gas stations in Surabaya is eight and ten clusters. This clustering process is conducted using the Equal-Size Spectral Clustering method. The results from both cluster numbers will be compared.

The visit sequence for 8 clusters is determined based on the cluster order, sequentially from cluster 1 to cluster 8. Node 0 serves as the start and end point for each cluster. Distance data for each gas station $C = [cij]$ is divided into 8 clusters with cluster 1 ($n1$) = 8, cluster 2 ($n2$) = 9, cluster 3 ($n3$) = 11, cluster 4 ($n4$) = 7, cluster 5 ($n5$) = 5, cluster 6 ($n6$) = 9, cluster 7 ($n7$) = 9, and cluster 8 ($n8$) = 12. Then, matrix C is modified by assigning a maximum value M = 9999 to the distance of each edge (i,j) where $i = j$.

The visit sequence for 10 clusters is determined based on the cluster order, sequentially from cluster 1 to cluster 10. Node 0 serves as the start and end point for each cluster. Distance data for each gas station $C = [cij]$ is divided into 10 clusters with cluster 1 ($n1$) = 6, cluster 2 ($n2$) = 11, cluster 3 ($n3$) = 6, cluster 4 ($n4$) = 8, cluster 5 ($n5$) = 8, cluster 6 ($n6$) = 6, cluster 7 ($n7$) = 6, cluster 8 ($n8$) = 5, cluster 9 ($n9$) = 6, dan cluster 10 ($n10$) = 7. Then, matrix C is modified by assigning a maximum value M = 9999 to the distance of each edge (i,j) where $i = j$.

C. *Algorithm Design*

1) *Initializing the Ant Colony*

The initialization of the ant colony begins with setting up key parameters such as the number of nodes, distance matrix, starting point, number of ants, alpha and beta values, pheromone evaporation coefficient, pheromone constant, and the number of iterations. The pheromone map is initialized with a small initial value for all node pairs. Then, ant objects are created using these parameters, with each ant having a starting location, a list of potential locations to visit, the pheromone map, distance function, alpha and beta values, and an indicator for the first iteration.

2) *Route Formation by Ants*

Each ant forms its route by selecting the next path based on probabilities determined by the amount of pheromone and the distance between nodes. Ants tend to choose paths with higher pheromone levels and shorter distances. After selecting a path, the ant moves to the next node, updating its route and the distance traveled. This process continues until all locations have been visited, and the ant returns to the starting location, completing its route.

3) *Pheromone Calculation*

After all ants complete their routes, the pheromone map is updated by evaporating some of the old pheromone and adding new pheromone based on the routes traveled by the ants.

4) *Route Update for Ants*

After updating the pheromone map, ants are reinitialized for the next iteration. This involves resetting the routes, the distances traveled, and the starting locations of the ants. The pheromone map updated by the ants is also reset for the next iteration, ensuring that old pheromone does not affect the new iteration. This step ensures that ants can explore new paths in the next iteration.

5) *Calculation of Opposite Solutions (Opposition-Based Learning)*

Opposition-Based Learning (OBL) is applied to enhance the exploration of solutions. This involves calculating the opposite location for each node in the ant's route and computing the total distance for the opposite route. If the opposite route has a shorter distance compared to the route found by the ant, the opposite route is considered a better solution. If the opposite route is better, the shortest route found and the shortest distance are updated with the values from the opposite route.

6) *Solution Evaluation*

After all iterations are completed, the shortest routes and distances found by the ants are collected. Then, the solutions with and without the application of OBL for each cluster are compared. An analysis is performed to assess how OBL affects the algorithm's performance in finding the optimal solution, and adjustments are made if needed for subsequent iterations or applications to other problems.

D. *Environment*

The testing environment used in this research implementation involves Microsoft Excel for processing cluster data and distance matrices. Additionally, Visual Studio Code with Python version 3.9.13 is used to execute the program code.

E. *Test Parameters & Scenarios*

To execute the ant colony optimization algorithm, several input parameters are necessary. The determination of the best combination of these parameters will yield the shortest total distance. The parameters used are as follows:

1) *Number of ants (m)*
2) *Influence of pheromone on path selection by ants (α)*
3) *Influence of distance on path selection by ants (β)*
4) *Pheromone evaporation rate (ρ)*
5) *Number of iterations*

Various scenarios are tested to determine the most optimal results. These scenarios are implemented on the datasets ftv33, ftv70, and ftv70 to assess the performance of the ant colony algorithm by comparing the trial results with the optimal values from TSPLIB. The best scenario derived from the dataset tests will then be applied to the gas station (SPBU) data in Surabaya. The scenarios used in this research are as follows:

1) *Comparison of total distance based on the parameter m set to 33, 70, and 70 (corresponding to the number of nodes in each dataset) for each trial.*

2) *Comparison of total distance based on the parameter α set to 0.5 for each trial.*

3) *Comparison of total distance based on the parameter β set to 3, 5, and 10 for each trial.*

4) *Comparison of total distance based on the parameter ρ set to 0.1 for each trial.*

5) *Comparison of total distance based on the number of iterations set to 1000, 5000, and 10000 for each trial.*

F. *Algorithm Implementation*

From the results of testing the algorithm on the ATSP datasets ftv33, ftv64, and ftv70 from the TSPLIB website with various parameter scenarios, different routes were obtained, each with varying total distances.

For the ftv33 dataset, the Ant Colony Optimization (ACO) algorithm with parameter scenarios of iteration count = 1000–10000; *m*=33; α=0.5; β=3,5,10; and ρ=0.1; produced the shortest route with a total distance of 1316, using β=5 and 10000 iterations. odification of Opposition-Based Learning (OBL) in Table 2, the shortest route achieved a total distance of 1311 using the same parameter settings of β=5 and 10000 iterations. The same parameter scenario for both algorithms resulted in the most optimal distance, demonstrating that the OBL modification produced a more optimal distance than the standard ACO algorithm.

TABLE I. FTV33 TOP FIVE RESULT USE ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
10000	5	1316	1286	2.333%
5000	5	1324	1286	2.955%
1000	3	1341	1286	4.277%
5000	3	1341	1286	4.277%
10000	3	1344	1286	4.510%

TABLE II. FTV33 TOP FIVE RESULT USE MODIFICATION OBL-ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
10000	5	1311	1286	1.944%
10000	3	1315	1286	2.255%
5000	5	1316	1286	2.333%
1000	3	1351	1286	5.054%
5000	10	1355	1286	5.365%

For the ftv64 dataset, the Ant Colony Optimization (ACO) algorithm with parameter scenarios of iteration count = 1000–10000; *m*=64; α=0.5; β=3,5,10; and ρ=0.1; produced the shortest route with a total distance of 1978, using β=5 and 10000 iterations. From modification of Opposition-Based Learning (OBL) in Table 4, the shortest route achieved a total distance of 1973 using β=5 and 5000 iterations. With fewer iterations, the modification of Opposition-Based Learning (OBL) achieved a more optimal route distance.

TABLE III. FTV64 TOP FIVE RESULT USE MODIFICATION OBL-ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
10000	5	1978	1839	7.558%
5000	5	1979	1839	7.613%
5000	10	2003	1839	8.918%
10000	10	2017	1839	9.679%
1000	5	2020	1839	9.842%

TABLE IV. FTV64 TOP FIVE RESULT USE MODIFICATION OBL-ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
5000	5	1973	1839	7.287%
1000	10	1979	1839	7.613%
10000	5	1995	1839	8.483%
5000	3	1996	1839	8.537%
10000	3	1999	1839	8.700%

For the ftv70 dataset, the Ant Colony Optimization (ACO) algorithm with parameter scenarios of iteration count = 1000–10000; *m*=70; α=0.5; β=3,5,10; and ρ=0.1; produced the shortest route with a total distance of 2127, using β=5 and 1000 iterations. From modification of Opposition-Based Learning (OBL) in Table 6, the shortest route achieved a total distance of 2120 using β=5 and 5000 iterations. The modification of Opposition-Based Learning (OBL) achieved a more optimal route distance with a larger number of iterations.

TABLE V. FTV70 TOP FIVE RESULT USE MODIFICATION OBL-ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
1000	5	2127	1950	9.077%
5000	5	2148	1950	10.154%
10000	5	2163	1950	10.923%
1000	10	2187	1950	12.154%
5000	10	2216	1950	13.641%

TABLE VI. FTV70 TOP FIVE RESULT USE MODIFICATION OBL-ACO ALGORITHM

Iteration	β	Distance	Optimum	Gap
5000	5	2120	1950	8.718%
1000	5	2158	1950	10.667%
1000	10	2170	1950	11.282%
10000	5	2170	1950	11.282%
5000	3	2212	1950	13.436%

Based on all the scenarios tested, β = 5 consistently yielded the best results. For ftv33, 10,000 iterations produced the most optimal solution. In ftv64, 10,000 iterations were optimal without using Opposition-Based Learning modification. Moving to ftv70, Opposition-Based Learning modification resulted in a better solution with 5,000 iterations, making 10,000 iterations the most productive for achieving optimal solutions.

The experiments indicate that Ant Colony Optimization with Opposition-Based Learning modification consistently outperforms the basic Ant Colony Optimization algorithm. This aligns with previous research findings favoring this modification for achieving optimal solutions. Additionally, higher iteration counts significantly contribute to the algorithm's success in achieving optimal values.

After testing with the best parameters, the results obtained from both Ant Colony Optimization algorithm and Opposition-Based Learning modification did not show any difference at the same cluster size. This indicates that the number of iterations was too large for the size of the clustered SPBU data. Therefore, the number of iterations was reduced to 400 to achieve the best results and faster runtime.

Table 8 represents the improvement of algorithm after modification. The best result obtained is 249.3. The average data is 249.76 with a standard deviation of 0.474. It is evident that Opposition-Based Learning modification can produce the most optimal results, albeit with a larger data spread compared to Ant Colony Optimization alone. In terms of time, this modification achieves optimal distance with an average time faster by 17.1946 seconds. This is 1.432 seconds faster compared to without modification.

TABLE VII. RESULTS OF REPEATED EXPERIMENT ITERATION 400 AND 8 CLUSTERS WITH ANT COLONY OPTIMIZATION ALGORITHM

n	Iteration	β	Distance (km)	Runtime (s)
1	400	5	249.7	15.89
2	400	5	249.9	23.053
3	400	5	250.3	18.436
4	400	5	249.7	17.78
5	400	5	249.9	19.103
6	400	5	250	17.645
7	400	5	249.5	18.39
8	400	5	249.8	17.341
9	400	5	249.7	18.311
10	400	5	250.3	20.317
Min			249.5	15.89
Average			249.88	18.6266
Standard Deviation			0.261618892	1.934808472

TABLE VIII. RESULTS OF REPEATED EXPERIMENT ITERATION 400 AND 8 CLUSTERS WITH ANT COLONY OPTIMIZATION ALGORITHM

n	Iteration	β	Distance (km)	Runtime (s)
1	400	5	249.3	16.245
2	400	5	249.7	20.179
3	400	5	249.7	15.69
4	400	5	249.9	17.72
5	400	5	250.9	17.583
6	400	5	249.3	16.539
7	400	5	249.3	16.871
8	400	5	249.7	16.905
9	400	5	250	16.934
10	400	5	249.8	17.28
Min			249.3	15.69
Average			249.76	17.1946
Standard Deviation			0.474224513	1.2107255

Meanwhile, with 10 clusters, the Ant Colony Optimization algorithm achieved the best result of 290.6, with an average of 291.93 after 10 trials. The standard deviation for this data was 1.11. In contrast, with the Opposition-Based Learning modification, Table 10 shows that the best result obtained was 289.4. The average data was 291.24 with a standard deviation of 0.448. This indicates that Opposition-Based Learning modification can produce both minimal and average distances better than without modification. The required time was also faster, with an average of 15.818 seconds and relatively consistent, as evidenced by the low standard deviation of 0.448 seconds.

TABLE IX. RESULTS OF REPEATED EXPERIMENT ITERATION 400 AND 8 CLUSTERS WITH ANT COLONY OPTIMIZATION ALGORITHM

n	Iteration	β	Distance (km)	Runtime (s)
1	400	5	291.5	15.44
2	400	5	291.7	15.86
3	400	5	290.6	15.32
4	400	5	291.2	15.42
5	400	5	290.8	16.33
6	400	5	292.5	15.74
7	400	5	291.9	15.33
8	400	5	291.9	16.26
9	400	5	290.9	16.56
10	400	5	289.4	15.92
Min			289.4	15.32
Average			291.24	15.818
Standard Deviation			0.872034658	0.448919691

TABLE X. RESULTS OF REPEATED EXPERIMENT ITERATION 400 AND 8 CLUSTERS WITH ANT COLONY OPTIMIZATION ALGORITHM

n	Iteration	β	Distance (km)	Runtime (s)
1	400	5	291.5	15.44
2	400	5	291.7	15.86
3	400	5	290.6	15.32
4	400	5	291.2	15.42
5	400	5	290.8	16.33
6	400	5	292.5	15.74
7	400	5	291.9	15.33
8	400	5	291.9	16.26
9	400	5	290.9	16.56
10	400	5	289.4	15.92
Min			289.4	15.32
Average			291.24	15.818
Standard Deviation			0.872034658	0.448919691

From these results, a comparison was made between the best outcomes of the Ant Colony Optimization algorithm and the Opposition-Based Learning modification. The findings show that the use of Opposition-Based Learning modification can achieve a more optimal distance compared to the Ant Colony Optimization algorithm, despite obtaining a higher standard deviation from 10 trials. Table 11 shows that 8 clusters can find the shortest distance with or without modification.

TABLE XI. BEST RESULTS COMPARISON

Method	Cluster Number	Distance (km)
Ant Colony Optimization	8	249.5
Ant Colony Optimization + Opposition-Based Learning	8	249.3

The best route results of the entire optimized cluster can be visualized in Figure 3. Considering the working hours of the personnel, these routes can be executed separately over eight days based on cluster divisions. For the visualization of each cluster, refer to Figures 4-11. Table 12 represents the route and distance for every cluster.

TABLE XII. ROUTE AND DISTANCE FOR EVERY CLUSTERS

Cluster	Route	Distance (km)
1	0-19-14-13-11-9-10-2-0	31.1
2	0-68-32-40-34-60-63-69-67-0	24.4
3	0-49-47-20-45-37-46-41-44-29-0	36.5
4	0-25-23-21-24-30-22-28-16-26-0	36.7
5	0-58-55-56-59-52-54-57-53-1-65-0	23
6	0-42-39-50-36-35-48-43-31-51-33-0	33.1
7	0-5-7-8-12-6-4-3-61-0	46.9
8	0-38-17-64-66-18-15-27-62-70-0	17.6
Total		249.3

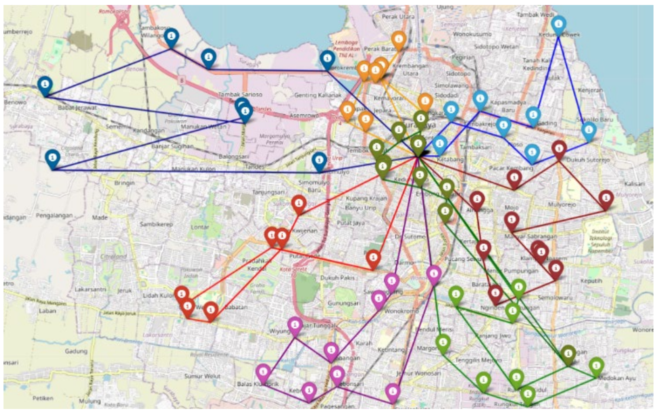
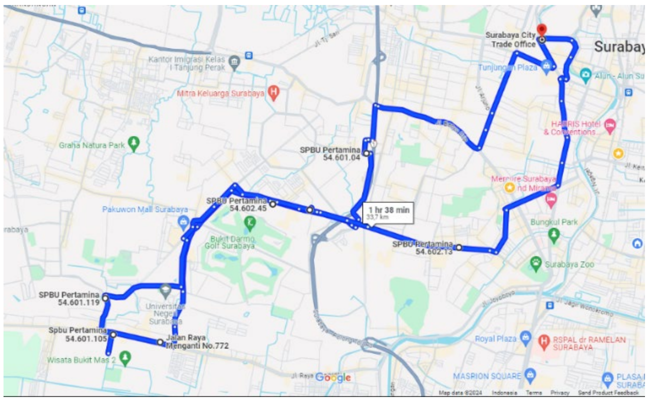
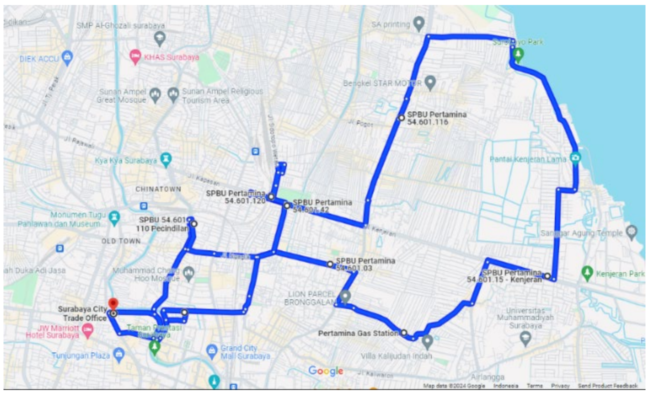


Fig. 4. Visualization after optimization for every clusters


Surabaya City Trade Office
SPBU Pertamina 54.601.04
SPBU Pertamina 54.602.45
SPBU Pertamina 54.602.13
SPBU Pertamina 54.601.119
Spbu Pertamina 54.601.109
Jalan Raya Menganti No.772
1 hr 38 min
Fig. 5. Visualization of Cluster 1 Routes after Ant Colony Optimization with Opposition-Based Learning


SPBU Pertamina 54.601.116
SPBU Pertamina 54.601.120
SPBU Pertamina 54.602.42
SPBU 54.601.110 Pecindilan
SPBU Pertamina 54.601.03
SPBU Pertamina 54.601.15 - Kenjeran
Surabaya City Trade Office
Pertamina Gas Station
Fig. 6. Visualization of Cluster 2 Routes after Ant Colony Optimization with Opposition-Based Learning

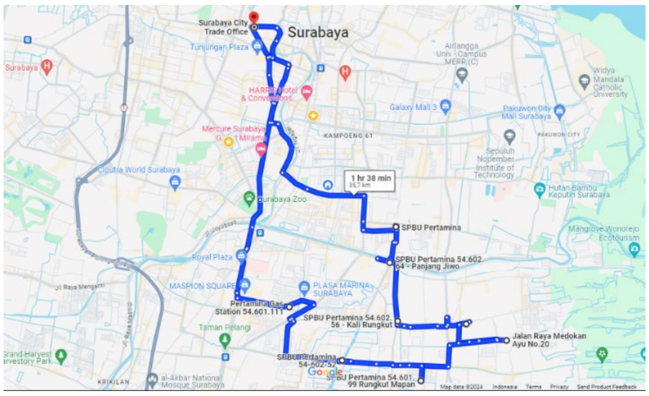

Surabaya City Trade Office
Surabaya
1 hr 38 min
SPBU Pertamina
SPBU Pertamina 54.602.64 - Panjang Jiwo
Pertamina Gas Station 54.601.111
SPBU Pertamina 54.602.56 - Kali Rungkut
Jalan Raya Medokan Ayu No.20
SPBU Pertamina 54.602.52
SPBU Pertamina 54.601.99 Rungkut Mapan

Fig. 7. Visualization of Cluster 3 Routes after Ant Colony Optimization with Opposition-Based Learning

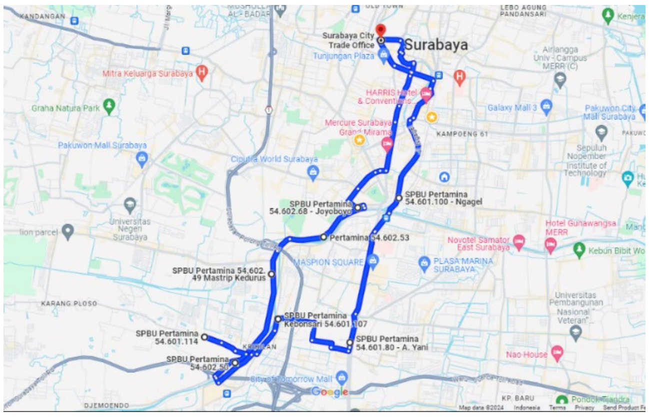

Surabaya City Trade Office
Surabaya
SPBU Pertamina 54.601.100 - Ngagel
SPBU Pertamina 54.602.68 - Joyoboyo
Pertamina 54.602.53
SPBU Pertamina 54.602.49 Mastrip Kedurus
SPBU Pertamina Kebonsari 54.601.107
SPBU Pertamina 54.601.80 - A. Yani
SPBU Pertamina 54.601.114
SPBU Pertamina 54.602.50
Google

Fig. 8. Visualization of Cluster 4 Routes after Ant Colony Optimization with Opposition-Based Learning

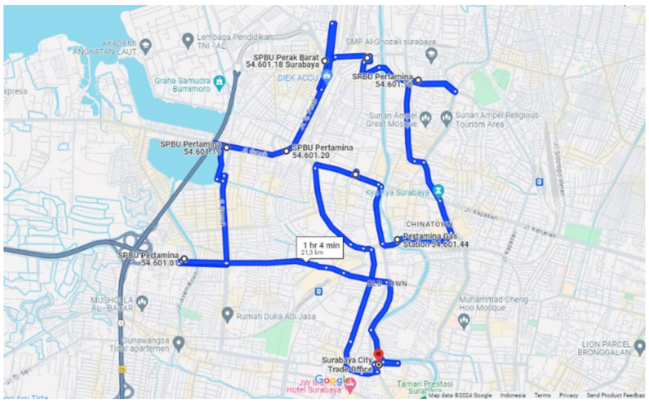


Fig. 9. Visualization of Cluster 5 Routes after Ant Colony Optimization with Opposition-Based Learning

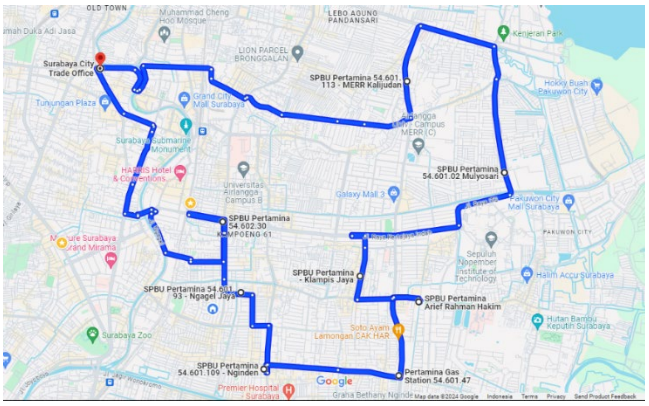

Surabaya City Trade Office
SPBU Pertamina 54.601.113 - MERR Kalijudan
SPBU Pertamina 54.601.02 Mulyosari
SPBU Pertamina 54.602.30
SPBU Pertamina - Klampis Jaya
SPBU Pertamina 54.601.93 - Ngagel Jaya
SPBU Pertamina Arief Rahman Hakim
SPBU Pertamina 54.601.109 - Nginden
Pertamina Gas Station 54.601.47
Google

Fig. 10. Visualization of Cluster 6 Routes after Ant Colony Optimization with Opposition-Based Learning

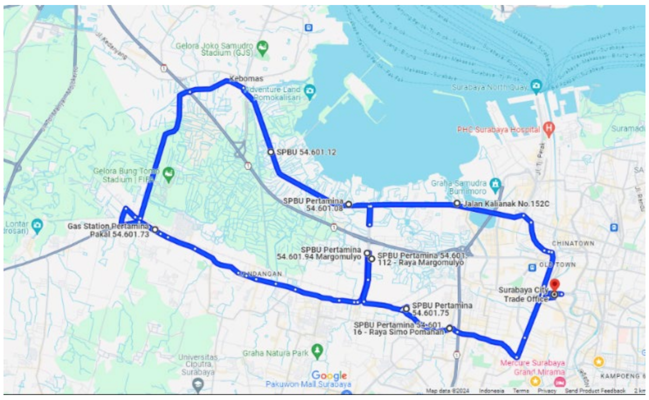

SPBU 54.601.12
SPBU Pertamina 54.601.08
Jalan Kalianak No.152C
Gas Station Pertamina Pakal 54.601.73
SPBU Pertamina 54.601.94 Margomulyo
SPBU Pertamina 54.601.112 - Raya Margomulyo
Surabaya City Trade Office
SPBU Pertamina 54.601.75
SPBU Pertamina 54.601.16 - Raya Simo Pomahan
Google

Fig. 11. Visualization of Cluster 7 Routes after Ant Colony Optimization with Opposition-Based Learning

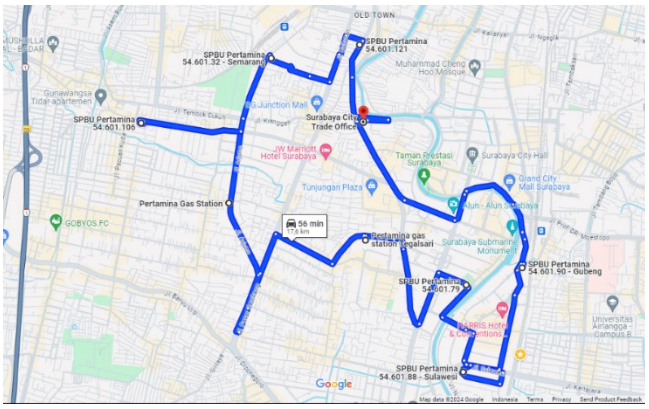

SPBU Pertamina 54.601.121
SPBU Pertamina 54.601.32 - Semarang
SPBU Pertamina 54.601.106
Surabaya City Trade Office
Pertamina Gas Station
56 min
17,6 km
Pertamina gas station Pegalsari
SPBU Pertamina 54.601.79
SPBU Pertamina 54.601.90 - Gubeng
SPBU Pertamina 54.601.88 - Sulawesi
Google

Fig. 12. Visualization of Cluster 8 Routes after Ant Colony Optimization with Opposition-Based Learning

In the current implementation, 3 officials from the Surabaya Office of Trade and Industry visit 3-5 SPBUs per day for inspections using a single minibus. The selection of SPBUs to visit each day is based on the order of inspection requests from Pertamina, not by proximity. The minibus used has a fuel consumption of 10.5 km per liter of Pertamax fuel, costing Rp. 12,950 per liter. It takes 18 days to visit all SPBUs in Surabaya.

By utilizing the routes presented in this study, the Surabaya Office of Trade and Industry can optimize time and estimate the costs required for inspections across all SPBUs in Surabaya. With clustering methods, the selection of SPBUs to visit daily becomes more effective due to their proximity. The total time required to visit all SPBUs in Surabaya is reduced to 8 days based on cluster order, with a total fuel cost estimated at Rp. 307,470.

V. Conclusion

Based on the results of experiments to solve the Asymmetric Clustered Traveling Salesman Problem (ACTSP) using the Ant Colony Optimization (ACO) with predetermined parameters, the following conclusions can be drawn:

1) The selection of beta value and number of iterations plays a crucial role in optimization using the Ant Colony Optimization algorithm.

2) A beta value of 5, representing the ants' visibility to distance, can provide optimal results in the asymmetric clustering travelling salesman problem.

3) The number of iterations that can yield optimal results in the asymmetric clustering travelling salesman problem is 400.

4) Opposition-Based Learning modification provides better results compared to Ant Colony Optimization without modification, both in dataset and SPBU data.

5) The minimal total distance achieved is 249.3 km.

6) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 1 is 0-19-14-13-11-9-10-2-0.

7) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 2 is 0-68-32-40-34-60-63-69-67-0.

8) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 3 is 0-49-47-20-45-37-46-41-44-29-0.

9) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 4 is 0-25-23-21-24-30-22-28-16-26-0.

10) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 5 is 0-58-55-56-59-52-54-57-53-1-65-0.

11) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 6 is 0-42-39-50-36-35-48-43-31-51-33-0.

12) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 7 is 0-5-7-8-12-6-4-3-61-0.

13) The optimal route generated using the Ant Colony Optimization algorithm with Opposition-Based Learning for Cluster 8 is 0-38-17-64-66-18-15-27-62-70-0.

Acknowledgment

The author would like to thank the Data Engineering and Business Intelligence Laboratory (RDIB) Information Systems Department FTEIC ITS for helping and providing facilities that support the implementation of this research.

References

[1] R. A. Palhares and M. C. B. Araújo, "Vehicle Routing: Application of Travelling Salesman Problem in a Dairy," in IEEE International Conference on Industrial Engineering and Engineering Management, 2019. doi: 10.1109/IEEM.2018.8607472.

[2] R. Dondo, C. A. Méndez, and J. Cerdá, "The multi-echelon vehicle routing problem with cross docking in supply chain management," Comput Chem Eng, vol. 35, no. 12, 2011, doi: 10.1016/j.compchemeng.2011.03.028.

[3] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: a literature review and classification," Operational Research, vol. 22, no. 3, 2022, doi: 10.1007/s12351-020-00600-7.

[4] M. M. Krishna, N. Panda, and S. K. Majhi, "Solving traveling salesman problem using hybridization of rider optimization and spotted hyena optimization algorithm," Expert Syst Appl, vol. 183, Nov. 2021, doi: 10.1016/j.eswa.2021.115353.

[5] A. Arigliano, G. Ghiani, A. Grieco, E. Guerriero, and I. Plana, "Time-dependent asymmetric traveling salesman problem with time windows: Properties and an exact algorithm," Discrete Appl Math (1979), vol. 261, pp. 28–39, May 2019, doi: 10.1016/j.dam.2018.09.017.

[6] G. E. A. Fuentes, E. S. H. Gress, J. C. S. T. Mora, and J. M. Marín, "Solution to travelling salesman problem by clusters and a modified multi-restart iterated local search metaheuristic," PLoS One, vol. 13, no. 8, 2018, doi: 10.1371/journal.pone.0201868.

[7] P. Stodola, P. Otřísal, and K. Hasilová, "Adaptive Ant Colony Optimization with node clustering applied to the Travelling Salesman Problem," Swarm Evol Comput, vol. 70, 2022, doi: 10.1016/j.swevo.2022.101056.

[8] S. A. Putra, "PENYELESAIAN ASYMMETRIC CLUSTERED TRAVELLING SALESMAN PROBLEM DENGAN ALGORITMA EVOLUTIONARY DISCRETE FIREFLY ALGORITHM PADA KASUS KONTROL SPBU SURABAYA," Surabaya, Jun. 2023.

[9] G. Sri Guntoro and M. Rukmini, "PENEGAKAN HUKUM PIDANA OLEH PENGAWAS KEMETROLOGIAN TERHADAP MANIPULASI POMPA UKUR BAHAN BAKAR MINYAK BERDASARKAN UNDANG-UNDANG NOMOR 2 TAHUN 1981 TENTANG METROLOGI LEGAL DALAM RANGKA PERLINDUNGAN KONSUMEN," Iustitia Omnibus, vol. I, no. 2, pp. 1–19, 2020.

[10] W. Li, C. Wang, Y. Huang, and Y. ming Cheung, "Heuristic smoothing ant colony optimization with differential information for the traveling salesman problem," Appl Soft Comput, vol. 133, 2023, doi: 10.1016/j.asoc.2022.109943.

[11] R. D. Lamperti and L. V. R. de Arruda, "A strategy based on Wave Swarm for the formation task inspired by the Traveling Salesman Problem," Eng Appl Artif Intell, vol. 126, Nov. 2023, doi: 10.1016/j.engappai.2023.106884.

[12] E. Osaba, X.-S. Yang, and J. Del Ser, "Traveling salesman problem: a perspective review of recent research and new results with bio-inspired metaheuristics," in Nature-Inspired Computation and Swarm Intelligence, Elsevier, 2020, pp. 135–164. doi: 10.1016/B978-0-12-819714-1.00020-8.

[13] K. V. Dasari and A. Singh, "Two heuristic approaches for clustered traveling salesman problem with d-relaxed priority rule," Expert Syst Appl, vol. 224, p. 120003, Aug. 2023, doi: 10.1016/j.eswa.2023.120003.

[14] Y. Lu, J.-K. Hao, and Q. Wu, "Solving the clustered traveling salesman problem via traveling salesman problem methods," PeerJ Comput Sci, vol. 8, p. e972, Jun. 2022, doi: 10.7717/peerj-cs.972.

[15] Z. H. Ahmed, "An exact algorithm for the clustered travelling salesman problem," OPSEARCH, vol. 50, no. 2, pp. 215–228, Jun. 2013, doi: 10.1007/s12597-012-0107-0.

[16] A. Gharehgozli, C. Xu, and W. Zhang, "High multiplicity asymmetric traveling salesman problem with feedback vertex set and its application to storage/retrieval system," Eur J Oper Res, vol. 289, no. 2, pp. 495–507, Mar. 2021, doi: 10.1016/J.EJOR.2020.07.038.

[17] G. Campuzano, C. Obreque, and M. M. Aguayo, "Accelerating the Miller–Tucker–Zemlin model for the asymmetric traveling salesman problem," Expert Syst Appl, vol. 148, p. 113229, Jun. 2020, doi: 10.1016/J.ESWA.2020.113229.

[18] Z. Zhang, Z. Xu, S. Luan, X. Li, and Y. Sun, "Opposition-Based Ant Colony Optimization Algorithm for the Traveling Salesman Problem," Mathematics, vol. 8, no. 10, p. 1650, Sep. 2020, doi: 10.3390/math8101650.

[19] D. Zhao et al., "Opposition-based ant colony optimization with all-dimension neighborhood search for engineering design," J Comput Des Eng, vol. 9, no. 3, pp. 1007–1044, Jun. 2022, doi: 10.1093/jcde/qwac038.

[20] U. Boryczka and K. Szwarc, "The Harmony Search algorithm with additional improvement of harmony memory for Asymmetric Traveling Salesman Problem," Expert Syst Appl, vol. 122, pp. 43–53, May 2019, doi: 10.1016/j.eswa.2018.12.044.*ial Intelligence*, 126. https://doi.org/10.1016/j.engappai.2023.106884