Jurnal Teknik Industri, Vol. 27, No. 1, June 2025
ISSN 1411-2485 print / ISSN 2087-7439 online
DOI: https://doi.org/10.9744/jti.27.1.1-20

Optimizing Vehicle Routing with Time Window Constraints: A Case Study on Bread Distribution

Rifqi Fauzi[1], Adi Priansyah[1], Paskalis Krisna Puspadewa[1], Syifa Maulvi Zainun Awal[1], Nguyen Huu Tho[2], Achmad Pratama Rifai[1*]

[1)] Department of Mechanical and Industrial Engineering, Universitas Gadjah Mada,
Jl. Grafika no 2. Yogyakarta 55281, Indonesia
[2)] Faculty of Engineering and Technology, Nguyen Tat Thanh University,
300A Nguyen Tat Thanh Street, Ward 13, District 4, Ho Chi Minh city, Viet Nam
Email: achmad.p.rifai@ugm.ac.id[1]
[*]Corresponding author

Abstract: This research investigates the application of optimization methods to the Capacitated Vehicle Routing Problem with Time Windows in the context of bread distribution for the efficiency of different approach for managing large-scale goods delivery. Managing this distribution requires considering complexities such as travel distance, vehicle capacity, and time windows. Specifically, it compares the performance of ALNS, SA, and GA in minimizing total travel distance while adhering to strict delivery windows. The research is conducted across different cases, each distinguished by varying levels of demand, nodes, and time windows for each case. Based on four cases, ALNS is the most effective method among the three methods in optimizing bread distribution. It was averagely 33.02% more efficient than SA and 57.21% than GA for minimizing travel distance and offering a robust solution, improving delivery efficiency across different case scenarios.

Keywords: Capacitated vehicle routing problem with time windows, bread distribution, simulated annealing, adaptive large neighborhood search, genetic algorithm.

Introduction

Demand for primary products represents basic societal needs that must be efficiently distributed to various destinations. As a perishable product, bread faces distribution challenges due to limited delivery times related to schedules and expiration dates. Bread deliveries cannot be made at any time; they must adhere to specific predetermined time windows [1].

Optimal route planning is essential to ensure perishable products like bread reach their destinations before expiration [2], [3]. Poor route planning can result in products arriving in unsellable condition, leading to financial losses and a damaged reputation for the distributor [1]. Additionally, untimely deliveries can cause product buildup at depots, increasing waste and reducing operational efficiency [3].

A real-world problem faced by a convenience store company with a distribution network across Bali Province involves distributing bread, a product with a limited shelf life. The company must also efficiently use resources and navigate facility constraints, such as the number of vehicles, vehicle capacities, the number of employees, and operational hours at each store. With 150 stores to service in a single day, it is crucial to meet the demand at each store using minimal distribution routes. This study compares several approximation methods, including Simulated Annealing, Adaptive Large Neighborhood Search, and Genetic Algorithm, to make timely and efficient decisions.

In the context of distributing perishable products, such as bread, several key factors must be considered, including the distance between depots and destinations, travel time, vehicle capacity, and the time windows set by each destination. The bread distribution problem in this study belong to the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW), in which the considerations of various vehicle capacity and time window that aligns with the operational hours of each destination as well as the expiration tolerance of the goods being delivered add complexity to distribution planning, requiring a systematic approach to achieve

optimal efficiency [4]. To address these challenges, incorporating additional constraints for vehicle capacity and strict time windows that are critical for maintaining product freshness are essential for the developed model.

This research aims to develop an optimization solution tailored specifically to the bread distribution process under the CVRPTW framework. Unlike general distribution models, this study targets the unique challenges associated with perishable goods, where minimizing total travel distance and delivery time is critical to maintaining product quality and reducing costs. By comparing various approximation methods like Simulated Annealing (SA), Adaptive Large Neighborhood Search (ALNS), and Genetic Algorithm (GA), the study seeks to identify the most effective approach for meeting all store demands within the specified time windows in the complex CVRPTW problem. Furthermore, the robustness of these methods will be tested under various scenarios, such as varying demand, variations of nodes and time windows. The ultimate objective of this research is to offer actionable insights and practical guidelines for minimizing travel distance with time windows, enabling them to optimize delivery routes, uphold product quality, and boost customer satisfaction in the context of bread distribution.

Literature Review

One of the primary challenges in distribution is meeting the demand at multiple destinations. Bujel *et al.* [4] utilized the Recursive-DBSCAN Algorithm to determine the minimum travel time for 5000 destinations, considering time windows. This method clusters data at nodes, significantly reducing the program's computation time and effectively handling 5000 destinations, compared to traditional methods which manage only 2000 destinations [4]. The unique challenge of distributing semi-perishable goods, such as medications with varying expiration dates, requires minimizing travel time while considering vehicle capacity and demand at each destination [3].

Redi *et al.* [3] compared the Capacitated Vehicle Routing Problem (CVRP) solutions using Simulated Annealing (SA) and Nearest Neighbors (NN) methods, finding that SA yielded more optimal travel times. Mangwanya and Masache [1] addressed the CVRP by comparing the Saving Based Algorithm and NN, demonstrating that these methods could significantly reduce distribution costs compared to existing conditions. Meanwhile, Putri *et al.* [5] proposed Genetic Algorithm with Cluster-first Route-second to solve the CVRP. Besides the limitation of vehicle capacity, the study also considered the issue of time windows in the VRP. Similarly, Utama *et al.* [6] also focused on VRP with time windows. In addition, the study considered the issue of green VRP by adding the minimization of fuel consumption as one of the objective functions to be solved by the proposed Camel Algorithm (CA).

Estrada-Moreno *et al.* [2] minimized transportation and opportunity costs for perishable products with multiple depots (MDVRP) using the Randomized Biased Algorithm, considering deadlines and penalty costs.

Londoño *et al.* [7] developed an MDVRP solution to minimize route distance and balance routing using the Chu Beasley Genetic Algorithm with Variable Neighborhood Search (CBGA-VNS), a hybrid method applied to extensive and complex case studies.

In another case, Wang *et al.* [8] employed a two-echelon approach considering multi-modal, multi-commodity optimization to minimize generalized costs in VRP, known as a two-echelon VRP Collaboration Point (2E-VRP-CP) scenario to minimize total distribution costs [8]. The NSGA-II algorithm was applied to optimize makespan and carbon emissions. Pingale *et al.* [9] highlighted the importance of integrating multiple aspects in delivery to provide simultaneous decision-making solutions. They compared exact methods with metaheuristics (ALNS) for last-mile delivery in the Multi-Modal Multi-Commodity Vehicle Routing Problem (2E-MCM-VRP). Pingale *et al.* [9] indicated that ALNS is able to reach excellence optimal routing to minimize generalized cost. Kuo *et al.* [10] proposed an algorithm that produces high-quality solutions, measured through hypervolume and spacing, and tested the significant benefits of using drones to reduce makespan and carbon emissions [10].

In terms of minimizing travel distance and maximizing vehicle capacity utilization for CVRP and CVRPTW, Natalia *et al.* [11] implemented the Bee Algorithm for optimal waste transportation routes, achieving more efficient and effective routes than the current government system [11]. Mutar *et al.* [12] examined solving the CVRP using the Ant Colony System (ACS) algorithm, which leverages sub-path experiences from previous iterations to find the best solution and achieve the shortest distance within a reasonable time. Table 1 summarizes previous related studies on vehicle routing problems.

Table 1. Related studies

Author	Problem	Objective function	Methods	Note
[1]	CVRP	Minimize Distribution cost	Saving Based Algorithm, NN	comparing manual and software's result
[2]	MDVRP	Minimize total transport cost and opportunity cost	Randomized-Biased Algorithm	redistribution of perishable products
[3]	CVRP	Minimize travel time	SA, NN	pharmaceutical distribution
[4]	CVRPTW	Minimize total travelled distance	Recursive-DBSCAN Algorithm	big nodes (5000)
[5]	CVRPTW	Minimize total distribution costs	GA	cluster-first route-second
[6]	VRPTW	Minimize cost of fuel consumption and the cost of late delivery	CA	Green VRP
[7]	MDVRP	Minimize routing distance and routing balancing	CBGA-VNS	hybrid method
[8]	2E-VRP-CP	Minimize total distribution cost	Exact, Greedy Algorithm	collaborative last mile delivery
[9]	MCM-2E-VRP	Minimize generalized cost	ALNS	multi model and multi commodity simultaneously
[10]	VRPTW	Minimize makespan and total carbon emissions	NSGA-II	integrating trucks and drones
[11]	CVRP and CVRPTW	Optimal routes to minimize distance	Bee algorithm	waste transportation routes
[12]	CVRP	Minimize the distance or total cost of the route	ACS	symmetric and asymmetric TSP
This research	CVRPTW	Minimize total travel distance	SA, ALNS, GA	robustness analysis

Pingale *et al.* [9] conducted the excellent implementation of ALNS to minimize generalized cost. As a result, comparing with other metaheuristic approach is needed to test it. This research fills a gap in the existing literature by comparing the methods of SA, ALNS, and GA. The three metaheuristics are selected due to simplicity and adept as escaping local optima, besides of that comparing among population based dan trajectory based importantly needed to prove which method is near with optimal solution. Further, this study also tests the robustness of each method's results, ensuring reliable managerial recommendations even under varying demand, vehicle numbers, capacities, and other company constraints.

Methods

This study employs a comparative analysis of several methods to identify the best results for running the algorithm. The data processing is conducted using MATLAB 2024a, supplemented by Excel and Python for data extraction and processing.

This research begins with a literature review and data collection to identify real-world problems and gain a comprehensive understanding of the bread distribution problem. The research begins with a literature review and field study to identify real-world problems such as bread distribution problems. The literature review focuses on topics related to the Capacitated Vehicle Routing Problem (CVRP) and commonly used methods for solving it, such as Simulated Annealing, Adaptive Large Neighborhood Search, and Genetic Algorithm. The field study involves adjusting to actual conditions and collecting the necessary data, including average demand data, distance data, and store location data.

A mathematical model is developed to minimize travel distance, considering constraints like vehicle limitations, time windows, and capacities. The model is solved and evaluated with three different metaheuristic algorithms: Simulated Annealing (SA), Adaptive Large Neighborhood Search (ALNS), and Genetic Algorithm (GA). Parameter tuning is determined using Design of Experiments (DOE), replicating parameter combinations five times.

Data processing is subsequently carried out with MATLAB 2023a for each method. Following the application of these algorithms, a comparative analysis is conducted to evaluate their performance and identify the most

suitable method for the problem at hand. A robustness analysis is then performed to assess how well the obtained solutions remain effective under varying conditions, such as changes in demand or vehicle capacity.

The final steps include conducting robustness analysis using statistical comparisons and analyzing the results. Drawing conclusions and making recommendations constitute the last steps of this research. This study employs a comparative analysis of several methods to identify the best results for running the algorithm. The data processing is conducted using MATLAB 2023a, supplemented by Excel and Open Street Map (OSMNX) Python for data extraction and processing.

Mathematical Model

This section provides a detailed description of the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) for perishable products. The CVRPTW involves selecting routes for a limited number of vehicles to distribute products within specific time frames. The constraints of this problem include a homogeneous fleet of vehicles, a central depot or distribution center (DC), a set of geographically distributed stores considered as customers, and a network connecting the DC to the stores.

Customers are identified as $i = \{1, 2, 3, \ldots, N\}$. For notation convenience, the distribution center (DC) is denoted as a special customer corresponding to node 0. DC serves as both the distribution hub and the collection center. Each store/customer has a fixed delivery demand denoted as q_i. This notation indicates the amount of goods that must be delivered from the DC to the store/customer. All products are assumed to be compatible and can be loaded into the same vehicle. The DC does not have any delivery requirements. The unit of demand is measured in crates or boxes.

The vehicles used are box trucks with a capacity of 200 crates or boxes, all having the same capacity and assumed to have the same delivery cost. They are initially stationed at the DC and are dispatched according to the delivery schedule within the time windows. In this case, there are two vehicles, and each vehicle starts and ends its route at the DC. GIS from Open Street Map data was used to create a distance matrix, providing accurate representations of the actual distances between customers.

Table 2. Notations of mathematical model

Symbol	Description
M	Number of vehicles
N	Number of nodes
i, j	Index of nodes $i, j \in \{1,2,3, \ldots, N\}$
k	Index of vehicles used $k \in \{1,2,3, \ldots, M\}$
d_{ij}	Distance between node i dan j
q_j	Size of delivery for each node/store
Q_k	Vehicle capacity to transport goods
e_i	Earliest time delivery can be made
l_i	Latest time delivery must be completed
t_i	Travel time at node/store i
s_i	Service time at node/store i
d_k	Departure time of vehicle k to the depot/DC
a_i	Vehicle arrival time at node/store i
p_k	Arrival time of vehicle k to the depot/DC
T_k	Total travel time for vehicle k
w_{ij}^k	Waiting time or delay on the route from i to j for vehicle k
x_{ij}^k	A binary variable indicating whether vehicle k travels directly from node i to node j, 1 if vehicle k travels from node i to node j, 0 otherwise

Both the DC and the stores/customers have specific time windows. The DC has a time window during which vehicles cannot depart before a specified time and must return when the stock is depleted. Each store/customer has a specific service time window defining the earliest and latest times a vehicle can start servicing the store. There are four cases in this research. Case 1 consist of 20 stores, case 2 consists of 50 stores, case 3 consists of 100 stores dan cases 4 consists of 150 stores. Each case has several stores operating 24 hours a day and several stores operating for 15 hours a day from 07:00 to 22:00. The distribution workday starts at 07:00 at the DC, assuming the vehicles are pre-loaded. Each customer (store) has a service time representing the duration the vehicle spends unloading goods at that store, assumed to be 10 minutes per unloading. Routes start at the DC, visit multiple stores/customers in a single trip, and return to the DC. The following assumptions are made for

this study: (1) Distribution is conducted twice a week. (2) The vehicles used for distribution are box trucks. (3) All vehicles have the same capacity. (4) Six vehicles are used. (5) Constant demand. (6) The unloading time is assumed to be the same for every demand. (7) Each node/store is visited only once.

The following is the mathematical model developed for the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) for perishable products. This model scratches dan refers to Bujel *et al.* [4]. The notation for indices, sets, parameters, and decision variables used in the mathematical model is presented in Table 2.

Objective function:

$$\text{Min } F = \sum_{k=1}^{M}\sum_{i=0}^{N}\sum_{i=0}^{N} d_{ij}^{k} x_{ij}^{k} \tag{1}$$

Subject to:

$$\sum_{k=1}^{M}\sum_{i=0}^{N} x_{ij}^{k} = 1 \qquad \forall\, j \in \{1,2,3,\ldots,N\} \tag{2}$$

$$\sum_{k=1}^{M}\sum_{j=1}^{N} x_{ij}^{k} = 1 \qquad \forall\, i \in \{0,1,2,3,\ldots,N\} \tag{3}$$

$$\sum_{j=0}^{N} x_{ij}^{k} = \sum_{j=0}^{N} x_{ji}^{k} \qquad \forall\, i \in \{1,2,3,\ldots,N\};\ k \in \{1,2,3,\ldots,M\} \tag{4}$$

$$\sum_{j=0}^{N} x_{0j}^{k} = 1 \qquad \forall\, k \in \{1,2,3,\ldots,M\} \tag{5}$$

$$\sum_{i=1}^{N} x_{i0}^{k} = 1 \qquad \forall\, k \in \{1,2,3,\ldots,M\} \tag{6}$$

$$\sum_{j=0}^{N} q_j \left(\sum_{i=0}^{N} x_{ij}^{k} \right) \le Q_k \qquad \forall\, k \in \{1,2,3,\ldots,M\} \tag{7}$$

$$u_i - u_j + (n-1)x_{ij}^{k} \le n-2 \qquad \forall\, i,j \in \{1,2,3,\ldots,N\};\ i \ne j;\ k \in \{1,2,3,\ldots,M\} \tag{8}$$

$$a_j = a_i + t_{ij} + s_i x_{ij}^{k} \qquad \forall\, i \in \{1,2,3,\ldots,N\};\ k \in \{1,2,3,\ldots,M\} \tag{9}$$

$$e_i \le a_i \le l_i \qquad \forall\, i \in \{1,2,3,\ldots,N\} \tag{10}$$

$$p_k = d_k + \sum_{i=1}^{n}(t_{ij} + s_i)\, x_{ij}^{k} \qquad \forall\, i \in \{1,2,3,\ldots,N\};\ k \in \{1,2,3,\ldots,M\} \tag{11}$$

$$e_0 \le p_k \le l_0 \qquad \forall\, k \in \{1,2,3,\ldots,M\} \tag{12}$$

$$T_k = \sum_{i,j=1}^{n}(t_{ij} + w_{ij}^{k})\, x_{ij}^{k} + \sum_{i=1}^{n} s_i \qquad \forall\, i \in \{1,2,3,\ldots,N\};\ k \in \{1,2,3,\ldots,M\} \tag{13}$$

$$x_{ij}^{k} \in \{0,1\} \qquad \forall\, i,j \in \{1,2,3,\ldots,N\}; k \in \{1,2,3,\ldots,M\} \tag{14}$$

$$a_i, p_k \ge 0 \qquad \forall\, i,j \in \{1,2,3,\ldots,N\}; k \in \{1,2,3,\ldots,M\} \tag{15}$$

Equation (1) describes the objective function which is to minimize the total travelled distance of all vehicles. Equations (2) and (3) ensure only one vehicle visits a node. Equation (4) ensures route continuity. Equations (5) and (6) ensure that the vehicle departs from and returns to depot. Equation (7) ensures that the total size of the goods transported by the vehicle does not exceed its vehicle capacity. Equation (8) is Miller-Tucker-Zemlin (MTZ) constraint for sub-tour elimination. u_i is additional variable which represents position of node i in the route. Equation (9) calculate arrival time at node/store i. Equation (11) calculate arrival time of vehicle k to the depot/DC. Equations (10) and (12) ensure that time windows are met. Equation 13 calculates the total travel time for vehicle k. Lastly, equations (14) and (15) limit the decision variables value.

Simulated Annealing

Simulated Annealing (SA) is a metaheuristic algorithm inspired by the annealing process in metallurgy. In metallurgy, annealing involves heating a material to a high temperature and then gradually cooling it to reduce

defects and improve the crystal structure of the material [13]. This algorithm was first introduced by Kirkpatrick, Gelatt, and Vecchi (1983) and Černý (1985) and has proven effective compared to other metaheuristic methods for solving the Vehicle Routing Problem (VRP) [14].

The basic principle of SA is to optimize an objective function by exploring the solution space. It is commonly used for solving combinatorial optimization problems [14]. The algorithm searches for the best solution within the search space, retaining the best solutions found. It also accepts new solutions based on their relative cost, even if they are worse, with a probability determined by a temperature that decreases over time [13].

Simulated Annealing (SA) is widely used for solving combinatorial optimization problems like the Vehicle Routing Problem (VRP). SA has several advantages over other methods in solving VRP, particularly its ability to avoid local optima traps and find better global solutions [15], [16]. Additionally, the parameters of SA can be tuned to optimize specific characteristics of VRP problems, such as vehicle capacity and time windows. Sinaga [17] has shown that SA can solve multi-depot VRP by generating vehicle routes that minimize travel time. Thus, its application is not limited to standard VRP but extends to VRP with Time Windows, Multi Depot VRP, Heterogeneous Fleet VRP, and Capacitated Vehicle Routing Problem with Time Windows [15], [17], [18]. This research conducted only use routing and divide the route in balance for 6 vehicles.

Simulated Annealing uses exploration and exploitation operators to search for solutions. Three types of SA operators used in this research are swap, insert, and reverse sub-route operators. The operator used in each iteration is determined probabilistically [3]. The probabilities for accepting each operator are semi-subjective. If random value generated 0 < 0.333, the swap operator is used, which selects two nodes at random and swaps their positions. If 0.333 < 0.667, the insert operator is used, which selects a random node and inserts it at a different position to create a new solution. If 0.667 <1.000, the reverse operator is used, which selects two random nodes and reverses the order of nodes between them (flip process) to create a new solution.

The iterative process utilizing temperature reduction based on α, determined through Design of Experiments (DOE), can be scientifically formulated in the context of optimization of Simulated Annealing (SA) method. The search for an optimal solution involves a gradual decrease in a control parameter commonly referred to as temperature T. This temperature metaphorically controls the level of exploration within the solution space. This temperature decrease is regulated by the α, often termed the cooling rate. The α typically ranges between 0 and 1, and the new temperature in each iteration is calculated using below formula:

$$T_{new} = \alpha * T_{old} \tag{16}$$

In the context of CVRPTW, Simulated Annealing can be applied to find near-optimal solutions for vehicle routing that me*et all* vehicle capacity and time window constraints. In each iteration, the algorithm generates a new solution by randomly modifying the vehicle routes and then evaluates whether the new solution improves or worsens the current solution. By utilizing a cooling rate strategy, the algorithm effectively explores the solution space and eventually converges to better solutions [3].

In implementing Simulated Annealing for CVRPTW, solutions are represented as a set of vehicle routes, each containing several customers and meeting capacity and time window constraints [4]. The general pseudocode for SA and its application in this research is presented in Algorithm 1.

Algorithm 1 Pseudocode of SA

1	Input: $To, Tfinal, \alpha$
2	Generate an initial solution X
3	Calculate the fitness values of the solution $f(X)$
4	Initialize current temperature $T = To$
5	While $T > T\ final$
6	Perform solution modification $Xnew \leftarrow X$
7	Calculate the fitness $f(Xnew)$
8	Compute $\Delta = f\ Xnew - f(X)$
10	Metropolis criterion for acceptance
11	Adjust current temperature $T = \alpha T$
12	End while
13	Output: X and $f\ (X)$

Adaptive Large Neighborhood Search

The Adaptive Large Neighborhood Search (ALNS) method has a unique solution search process not found in other algorithms, specifically through its destroy and repair processes. Like other algorithms, each of these processes has specific operators. In the destroy process, operators include random removal (randomly selecting nodes) and worst removal (selecting the furthest nodes). In random removal, a certain number of components (i.e., often referred to as 'nodes' in the context of routing or scheduling problems) are randomly selected from the solution for removal. This method relies on stochastic processes, meaning it introduces a degree of randomness into the search, potentially helping to avoid local optima. Then the worst removal destruction involves identifying and removing the components of the solution that contribute least to its quality or that result in the most significant negative impact. For instance, in a vehicle routing problem, this might mean removing the stops that are furthest away or cause the route to be inefficient. The repair process features operators like random insert (repairing the destroyed solution with random alternative solutions) and greedy repair (using a greedy algorithm to replace destroyed solution nodes). Random Insert is a method where components removed from the solution during the destruction phase are replaced randomly. This involves selecting elements or nodes randomly from a candidate set and inserting them back into the solution at feasible random positions. Then, Greedy Repair utilizes a greedy algorithm to meticulously select the most beneficial or optimal elements for reintroduction from a pool of candidates. This selection is driven by specific metrics such travel distance, with a focus on comparing the shortest routes. Furthermore, after the destruction and repair processes, the routing will be adjusted to ensure that no locations are repeated within a single route for each vehicle.

Additionally, the destroy and repair processes in ALNS are weighted, and these weights are updated in each iteration. The destroy process removes several nodes randomly based on the degree of destruction (DoD) parameter. The number of nodes removed is calculated using the formula in Equation (13) [19].

$$\text{Number of removed nodes} = N\left(DoD_2 - \left(\left(DoD_2 - DoD_1\right)\left(\frac{t}{T}\right)\right)\right) \tag{17}$$

Where DoD_2 is the upper bound and DoD_1 is the lower bound of destruction parameter. t is the index of current iteration and T is total iterations. Afterward, the probability of selecting an operator $P(c)$ is calculated using the formula:

$$P(c) = \frac{w_c^{-/+}}{\sum_{d \in DM} w_d^{-/+}} \quad \forall\, c \in DM \tag{18}$$

Where $w_c^{-/+}$ is the weight of destroy or repair operator. Once an operator is selected, its weight is updated in the next iteration using the following equations:

$$w_{d,t+1}^{-} = \alpha\left(w_{d,t}^{-}\right) + (1-\alpha)\beta \tag{19}$$

$$w_{d,t+1}^{+} = \alpha\left(w_{d,t}^{+}\right) + (1-\alpha)\beta \tag{20}$$

$$\beta = \begin{cases} Z_1 \; if \; f(Xnew) \geq f(X) \\ Z_2 \; if \; f(Xnew) > f(X) \end{cases} \tag{21}$$

Algorithm 2 Pseudocode of ALNS

```
Input: T, D
Generate an initial solution X
Calculate the fitness values of the solution f(X)
Set Initial Weight w⁻ and w⁺
While t < Tfinal
    Select Destroy and Repair method based on its weight
    Perform destroy operator DX ← d(X)
    Perform repair operation Xnew ← r(DX)
    Perform adjustment to the new solution Xnew
    Calculate the fitness values of the solution f(Xnew)
    If f(Xnew)< f(X)then f(X) = f(Xnew) , X = Xnew
    Update the weights based on previous performance
End while
Output: X and f (X)
```

Here, $w^{+}_{d,t+1}/w^{-}_{d,t+1}$ is the weight of repair/destroy operator in the next iteration, while $w^{-}_{d,t}/w^{+}_{d,t}$ is the weight in previous operators. β is a decay parameter, controlling the sensitivity or weight to change. The higher the value, the more reluctant the weight is to change. Meanwhile, α is the score for changing the weights based on the operator's performance in the previous iteration [19]. Algorithm 2 presents the ALNS pseudocode in general and what is applied in this study.

Genetic Algorithm

Genetic algorithms (GA) are population-based algorithms, distinct from Simulated Annealing (SA) and Adaptive Large Neighborhood Search (ALNS). Like biological inheritance processes, GA involves selecting the best parents for crossover of their DNA and mutation of the parent DNA to produce offspring that have potential as alternative solutions [20]. The parent selection, crossover, and mutation processes depend on the initiated parameters, namely crossover rate and mutation rate.

Genetic algorithms (GA) have been successfully applied to various supply chain problems, such as the Vehicle Routing Problem (VRP). The effectiveness of GA in solving capacitated VRP highlights improvements in managerial decision-making processes and supply chain efficiency. Other studies indicate that while the combination of GA and Iterated Local Search (ILS) is effective for small-scale data, it is less efficient for large-scale data due to its susceptibility to local optima and long computation times [21]. Then, if the solution violates the constraints, especially in the time window, there is a big number penalty that causes the solution to be rejected and the selection, crossover, and mutation processes to be repeated and the algorithm will move towards selecting and evolving feasible solutions that respect the time windows. Algorithm 3 describes the pseudocode of GA.

Algorithm 3 Pseudocode of GA

```
Input: P, g, co, mu
Generate an initial Population
While i < g
    Select Parents from population
    Produce offsprings from selected parents using crossover
    Mutate the offsprings
    Extend the population adding offsprings to it
    Reduce the extended population
End
Output: X and f (X)
```

Comparison Testing

At this stage, the optimal parameters for each method will be tested 10 times. Statistical analysis will be conducted to understand the characteristics of the results from each trial. This will involve analyzing the smallest and largest data points, the average data, and the standard deviation of the trial data [20].

Results and Discussions

The SA, ALNS, and GA methods are influenced by various parameters that can significantly impact solution quality. To optimize these parameters for the specific problem, a Design of Experiment (DOE) approach was employed. By systematically varying parameters and analyzing the resulting solutions, DOE identified optimal parameter settings for each method. This study utilized a fractional factorial design with three levels (low, medium, and high) for each parameter, enabling efficient exploration of the parameter space while minimizing the number of experiments required

Parameter Tuning

Parameter tuning or the best parameter selection is conducted using DOE for each method. Each parameter combination is replicated five times. The results of each combination are then compared to obtain the tuning parameters that yield the best solution for each method.

SA Parameters

For the SA method, the parameters that need tuning through DOE are the initial temperature (T0) and alpha (α). The T0 parameter determines the starting point of the iteration process, while the α parameter controls the

rate of temperature decrease. Higher values of T0 and α result in more iterations. The final temperature parameter (Tfinal) for the SA method is already set at Tfinal = 0.001 that can prevent premature convergence. And other parameter set with combination of high value (T0= 2000 and α=0.99) dan small value (T=1000 and α=0.95) to create search space on this research.

There are four parameter combinations subjected to DOE in the SA method, as shown in Table 3. The DOE results indicate that T0 = 2000 and α = 0.99 provide the best results. These parameters will be used as tuning parameters for comparison with the ALNS and GA methods.

The results in Table 3 demonstrates that α = 0.99 yields better outcomes compared to α = 0.95. A higher α value results in a slower temperature decrease, thus allowing more iterations. This indicates that for this problem, the SA method requires more iterations to find the best or near-optimal solution. More iterations allow the SA method to perform more extensive exploration and exploitation. Although more iterations increase the running time, they provide significantly better or closer-to-optimal solutions.

Table 3. DOE results for SA parameters

Parameters	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
	616.74	2116.74	12.98
T0 = 1000	640.08	2140.08	11.50
α = 0.95	708.89	2208.89	11.69
	736.53	2236.53	11.47
	660.11	2160.11	11.47
Average	672.47	2172.47	11.82
	508.71	2008.71	60.49
T0 = 1000	540.10	2040.10	59.18
α = 0.99	516.56	2016.56	55.41
	516.56	2016.56	60.39
	516.56	2016.56	57.63
Average	519.70	2019.70	58.62
	683.70	2183.70	13.01
T0 = 2000	683.70	2183.70	17.48
α = 0.95	660.67	2160.67	18.78
	660.67	2160.67	19.34
	664.46	2164.46	15.02
Average	670.64	2170.64	16.73
	531.98	2031.98	75.99
T0 = 2000	499.71	1999.71	70.22
α = 0.99	489.55	1989.55	75.84
	464.60	1964.60	93.44
	537.77	2037.77	96.05
Average	**504.72**	**2004.72**	**82.31**

ALNS Parameters

For the ALNS method, the tuning parameters that require DOE are the total iterations T and the Degree of Destruction (DoD) parameter. These parameters are set based on trial error in small case then concluded the categories of each parameter. The T parameter controls the number of iterations, while the DoD parameter sets the percentage of nodes that need to be destroyed and repaired. The larger the value of T, the more iterations the process will undergo. The DoD parameter has both lower and upper limit values that determine the number of nodes to be destroyed and repaired. A greater difference between the lower and upper limit values results in a higher percentage of nodes being destroyed and repaired. In this study, the DOE stage for the ALNS method only determines the upper limit value, while the lower limit value is set at 0.1.

As shown in Table 4, the DOE for the ALNS method tests four parameter combinations. The DOE results indicate that T = 1000 and the upper limit value DoD = 0.5 provide the best results. These parameters will be used for tuning to compare with the SA and GA methods.

The DOE results in Table 4 reveal that a higher number of iterations and a wider DoD range offer a better solution. More iterations allow for more extensive exploration and exploitation processes. A wider range of DoD values provides a broader exploration space at the beginning of the iteration because a higher percentage of

nodes are destroyed and repaired. While an increased number of iterations extends the running time, a larger DoD range results in faster times with the same number of iterations compared to a smaller DoD range.

Table 4. DOE results for ALNS parameters

Parameters	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
T = 500 DoD = [0.1 0.4]	444.05	1944.05	7.12
	432.59	1932.59	4.93
	439.63	1939.63	5.12
	424.53	1924.53	4.39
	454.80	1954.80	4.33
Average	439.12	1939.12	5.18
T = 500 DoD = [0.1 0.5]	440.27	1940.27	4.39
	449.18	1949.18	3.67
	445.97	1945.97	3.61
	431.93	1931.93	3.64
	436.14	1936.14	3.64
Average	440.70	1940.70	3.79
T = 1000 DoD = [0.1 0.4]	435.88	1935.88	8.39
	420.18	1920.18	10.37
	424.05	1924.05	8.31
	426.51	1926.51	8.92
	438.63	1938.63	8.72
Average	429.05	1929.05	8.94
T = 1000 **DoD = [0.1 0.5]**	418.48	1918.48	9.16
	417.87	1917.87	8.43
	422.44	1922.44	8.43
	433.29	1933.29	8.38
	424.87	1924.87	8.23
Average	**423.39**	**1923.39**	**8.53**

GA Parameters

For the Genetic Algorithm (GA) method, the tuning parameters requiring DOE are the population size (P), the number of generations or iterations (*g*), the crossover rate (*co*), and the mutation rate (*mu*). The parameter P determines the population size or the number of solutions to be processed, and g sets the number of generations or iterations for the GA process. The parameters co and mu are related to the exploration and exploitation processes; a higher co value increases the likelihood of crossover, while a higher mu value increases the likelihood of mutation. A larger co value expands the search space or exploration process, whereas a larger mu value enhances the exploitation of a single solution.

There are 16 parameter combinations tested through DOE for the GA method, as shown in Table 5. The DOE results indicate that the combination of P = 1000, $g = 500$, $co = 0.8$, and $mu = 0.2$ yields the best results compared to other parameter combinations. These parameters will be used for tuning and comparison with the SA and ALNS methods.

The DOE results in Table 5 show that population size plays a crucial role in the GA method. The larger the population size, the more solution combinations can occur for exploration. When P = 100 or P = 1000, changes in the *g, co*, and *mu* parameters do not significantly affect the results. Altering the P value increases the likelihood of finding the best solution in each iteration, so a larger P value produces better solutions.

The sensitivity of *co* and *mu* values is significant in producing optimal solutions. Changes in co and *mu* values, when P and *g* are constant, yield different solutions. An increase in co must be balanced with an increase in mu. A higher co value combined with a higher *mu* value provides better solutions than when co or mu values are opposed. As shown in Table 4, when $co = 0.7$, the solution diverges from the optimal, indicating that the number of iterations and a more extensive DoD range provide better solutions. More iterations allow for more extensive exploration and exploitation processes. A larger DoD range offers a broader exploration space at the beginning of the iteration due to the higher percentage of nodes being destroyed and repaired. While increasing the number of iterations lengthens the running time, a larger DoD range results in faster times with the same number of iterations compared to a smaller DoD range.

Table 5. DOE results for GA parameters

Parameters	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
P = 100 g = 500 co = 0.7 mu = 0.1	1210.63	6060.63	2.95
	1193.42	5988.42	2.56
	1070.39	5980.39	2.30
	1113.48	5753.48	2.48
	1089.35	5784.35	2.35
Average	1135.45	5913.45	2.53
P = 100 g = 500 co = 0.7 mu = 0.2	1126.59	6066.59	2.74
	1113.64	6003.64	2.53
	1102.75	5762.75	2.41
	1101.68	5921.68	2.42
	1022.78	5837.78	2.40
Average	1093.49	5918.49	2.50
P = 100 g = 500 co = 0.8 mu = 0.1	1127.76	5842.76	2.94
	1034.12	5484.12	2.55
	1158.54	5973.54	2.46
	1124.08	6009.08	2.43
	1135.08	5825.08	2.46
Average	1115.91	5826.91	2.57
P = 100 g = 500 co = 0.8 mu = 0.2	1044.69	5824.69	2.64
	996.31	5771.31	2.42
	1075.96	5860.96	2.37
	1106.95	5796.95	2.36
	1044.68	5984.68	2.53
Average	1053.72	5847.73	2.46
P = 100 g = 1000 co = 0.7 mu = 0.1	995.57	5540.57	4.79
	1028.66	5653.66	4.63
	1046.29	5831.29	4.63
	1111.10	5776.10	4.33
	959.29	5589.29	4.36
Average	1028.18	5678.18	4.55
P = 100 g = 1000 co = 0.7 mu = 0.2	1014.74	5769.74	5.47
	1049.73	5924.73	5.12
	1036.28	5871.28	4.56
	1074.94	5999.94	4.68
	973.38	6123.38	4.54
Average	1029.81	5937.81	4.87
P = 100 g = 1000 co = 0.8 mu = 0.1	1042.84	5792.84	7.79
	1035.12	5940.12	8.51
	1070.30	5705.30	7.61
	1098.39	5743.39	8.23
	1036.56	5726.56	7.02
Average	1056.64	5781.64	7.83
P = 100 g = 1000 co = 0.8 mu = 0.2	1043.20	6078.20	8.53
	1001.09	5696.09	8.97
	1051.55	5766.55	7.74
	1078.32	5723.32	6.86
	934.52	5789.52	7.43
Average	1021.73	5810.73	7.91
P = 1000 g = 500 co = 0.7 mu = 0.1	889.28	5774.28	22.06
	979.62	5619.62	21.80
	853.40	5488.40	20.47
	1005.70	5780.70	18.70
	870.54	5705.54	19.18
Average	919.71	5673.71	20.44
P = 1000 g = 500 co = 0.7 mu = 0.2	1024.00	5849.00	22.57
	865.28	5830.28	23.12
	908.69	5618.69	19.53
	975.59	5690.59	20.79
	916.67	5811.67	19.50
Average	938.04	5760.04	21.10
P = 1000 g = 500 co = 0.8 mu = 0.1	896.23	5686.23	23.38
	929.13	5949.13	32.45
	932.60	5842.60	31.59
	1011.24	5701.24	29.96
	913.94	5788.94	27.07
Average	936.63	5793.63	28.89

Parameters	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
P = 1000 g = 500 co = 0.8 mu = 0.2	859.56	5819.56	27.39
	883.21	5788.21	27.99
	938.21	5818.21	27.46
	971.49	5771.49	27.30
	851.01	5981.01	27.33
Average	**900.70**	**5835.70**	**27.49**
P = 1000 g = 1000 co = 0.7 mu = 0.1	953.82	5738.82	67.21
	916.69	5706.69	56.24
	1008.81	5878.81	51.24
	1001.42	5636.42	50.16
	943.62	5793.62	57.39
Average	964.88	5750.88	56.45
P = 1000 g = 1000 co = 0.7 mu = 0.2	930.64	5685.64	53.08
	934.64	5679.64	51.73
	913.61	5628.61	53.90
	917.84	5577.84	52.81
	1018.97	5748.97	51.49
Average	943.14	5664.14	52.60
P = 1000 g = 1000 co = 0.8 mu = 0.1	866.48	5716.48	57.91
	943.25	5698.25	56.68
	914.88	5694.88	55.51
	924.22	5864.22	57.69
	944.31	5814.31	57.46
Average	918.63	5757.63	57.05
P = 1000 g = 1000 co = 0.8 mu = 0.2	880.85	5570.85	56.91
	860.57	5845.57	55.96
	907.78	5572.78	55.34
	964.34	5934.34	55.79
	930.20	5740.20	57.99
Average	908.75	5732.75	56.40

SA Results

Based on the most optimal parameter tuning for determining the minimum distribution distance, the results of processing the CVRPTW using SA are presented in Table 6.

Based on 10 experiments each case, the minimum travelled distance achieved using the Simulated Annealing (SA) method in experiment 4 on case 1 was 296.99 km, the total travel time was 556.99 minutes, with a computation time of 58.65 seconds. Then, case 2 was 300.58 km, the total travel time was 860.58 minutes, with a computation time of 34.78 seconds. The case 3 was 403.48 km, the total travel time was 1463.48 minutes, with a computation time of 63.40 seconds. Then, case 4 was 522.98 km, the total travel time was 2022.98 minutes, with a computation time of 80.72 seconds. The process of running the SA method in experiment 4 case 4 to minimize the total travel distance is illustrated in Figure 2. The division of nodes for each vehicle from the solution of case 4 is detailed in Table 7.

Table 6. SA results in average

Case	Number of vehicles	Number of destinations	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
1	6	20	296.99	556.99	58.65
2	6	50	300.58	860.58	34.78
3	6	100	403.48	1463.48	63.40
4	6	150	522.98	2022.98	80.72

Figure 2 illustrates that at temperatures approaching 1000, the minimum distance increases slightly up to a temperature of 1200. This indicates that the exploration and exploitation process is generally running well and does not show signs of premature convergence. As the temperature decreases, the SA method effectively explores and exploits the solution space, maintaining a balance that prevents early convergence and ensures a thorough search for the optimal solution. In general, the more cases there are, the longer the computing time will be.

Based on the data processing results for bread distribution using the Simulated Annealing (SA) method, it can provide optimal distribution route suggestions. However, there is an imbalance in the assignment of tasks to each vehicle, resulting in an unequal workload distribution among the vehicles.

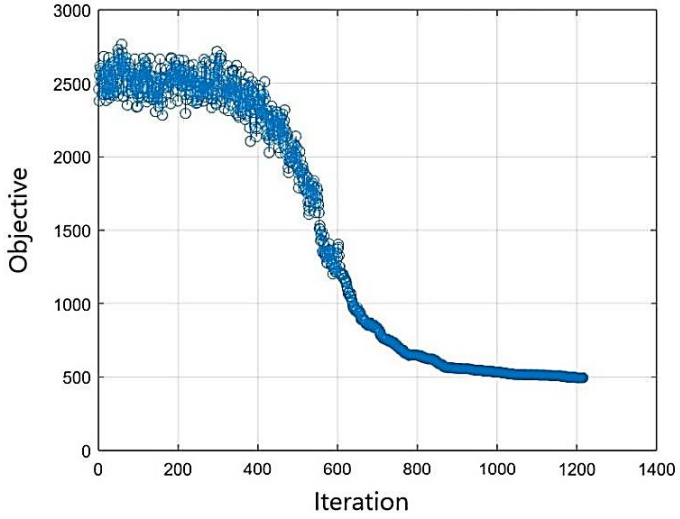

Figure 2. Fourth case iteration process of SA

Table 7. ALNS results in average

Case	Number of vehicles	Number of destinations	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
1	6	20	128.03	338.03	3.42
2	6	50	177.30	645.30	4.90
3	6	100	331.11	1371.11	7.50
4	6	150	424.56	1924.56	7.95

ALNS Results

Based on the most optimal parameter tuning for determining the minimum travelled distance, the results of processing the CVRPTW using ALNS are presented in Table 7.

Based on 10 experiments in each case, the minimum travelled distance, minimum travel time dan computation time achieved using the Adaptive Large Neighborhood Search (ALNS) method show the different values in Table 7.

According to Figure 3 as iteration process of ALNS from case 4, it shows that up to iteration 800, the minimum distance remains relatively stable, but after iteration 800, there is a significant reduction in the minimum distance until convergence is achieved around iteration 1000. Overall, the exploration and exploitation processes are running well, indicating an initial tendency towards premature convergence that is effectively prevented by a sufficiently high number of iterations, which is 1000 iterations.

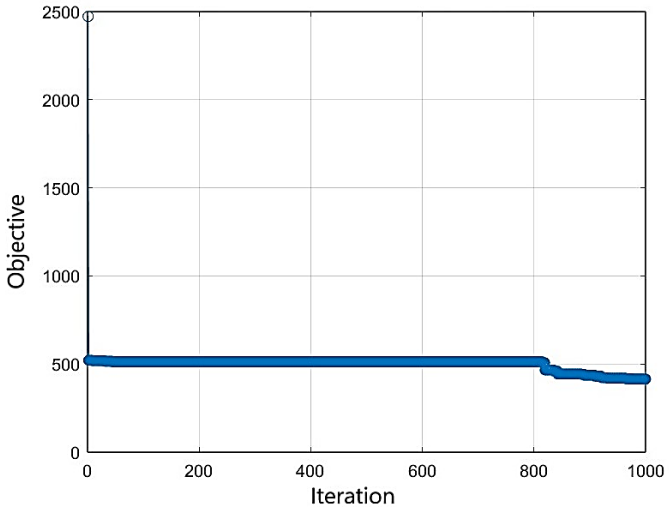

Figure 3. Fourth case iteration process of ALNS

Based on the data processing, the ALNS can allocate tasks in a more balanced manner, ensuring that only five vehicles are needed. This optimization can offer significant benefits for the company by improving fuel efficiency and allowing the reallocation of resources to other tasks.

GA Results

Based on the most optimal parameter tuning for determining the minimum distribution distance, the results of processing the CVRPTW using GA are presented in Table 8.

Table 8. GA results in average

Case	Number of vehicles	Number of destinations	Total travelled distance (Km)	Total travel time (minutes)	Computation time (seconds)
1	6	20	298.36	548.36	20.47
2	6	50	502.76	1062.76	15.56
3	6	100	731.32	1791.32	10.83
4	6	150	916.66	2416.66	9.86

Based on 10 experiments in each case, the minimum travelled distance, minimum travel time dan computation time achieved using the Adaptive Large Neighborhood Search (ALNS) method show the different values in table 7. The process of running the GA method in case 4 to minimize the total travelled distance is illustrated in Figure 4.

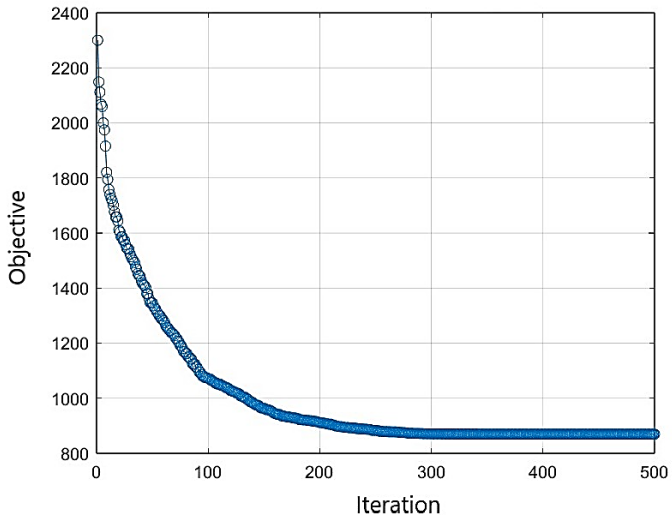

Figure 4. Fourth case iteration process of GA

Based on Figure 4, it shows that at generations approaching 300, the minimum distance stabilizes and remains consistent until generation 500. Overall, the exploration and exploitation processes are running well and do not indicate any signs of premature convergence.

Like SA, the results of GA yielded an imbalance in the assignment of tasks to each vehicle, resulting in an unequal workload distribution among the vehicles. Additionally, the minimum distance results obtained with the GA method cannot be recommended. This is due to the significantly higher minimum distance value compared to other methods.

Algorithm's Comparison

Comparison analysis is conducted to determine the reliability and stability of the algorithms in producing optimal solutions. As shown in Figures 5, 6, and 7, the processing results using the SA, ALNS, and GA methods indicate that the ALNS method produces the most optimal solutions compared to SA and GA. The ALNS method has the most optimal solution in terms of total distance traveled and total travel time. Therefore, the distribution routes generated by the ALNS method are selected as the recommended solution.

Compared to the GA method, which is a population-based method, the SA and ALNS methods, which are trajectory-based, provide better results. As shown in Table 8, the total distance and travel time produced by the

GA method are significantly different from those produced by the SA and ALNS methods. Several factors could explain why the GA method's results are less optimal, such as an insufficient number of iterations and a lower crossover rate (co), leading to local optima. However, increasing the number of iterations can result in longer running times.

The minimum distance comparison between the SA, ALNS, and GA methods in the previous sub-chapter shows that the ALNS method produces the shortest distance, followed by the SA method. The results of the two methods are close enough to validate the algorithms against each other. Interestingly, the results obtained with the GA method show almost double the minimum distribution distance compared to the SA and ALNS methods. This anomaly may be due to non-optimal GA parameters, indicating a need for further experimentation to achieve a minimum distance comparable to the SA and ALNS results. Similarly, the travel time results for the three methods correlate with the distribution distance and the loading/unloading service time.

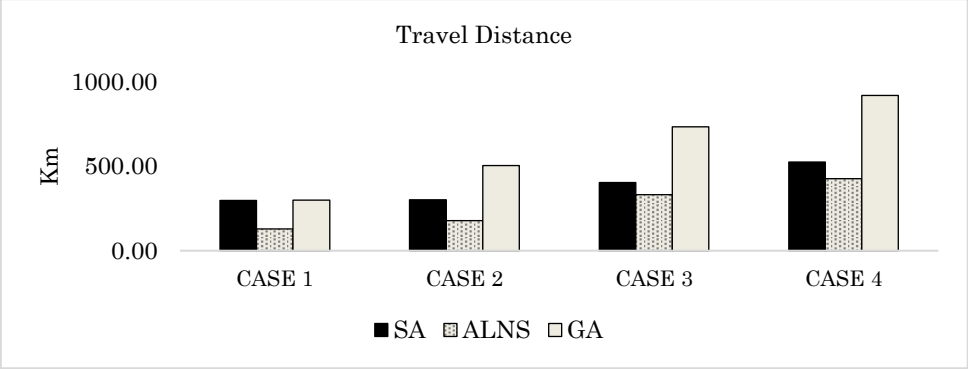

Figure 5. Comparison of minimum total travel distance

The bar chart presented in Figure 5 provides a comparative analysis of the minimum total travel distances for three optimization methods including Simulated Annealing (SA), Adaptive Large Neighborhood Search (ALNS), and Genetic Algorithm (GA) across four cases. It reveals distinct performance trends: ALNS generally excels, particularly in all cases, by achieving significantly lower travel distances compared to SA and GA, highlighting its robust capability in diverse and complex scenarios. However, its performance drops in Case 4, where ALNS demonstrates superior efficiency, suggesting its effectiveness in dynamically adapting search strategies to complex challenges. SA, while consistently providing moderate results, does not achieve the lowest travel distances in any case, indicating its utility as a stable but not always optimal choice

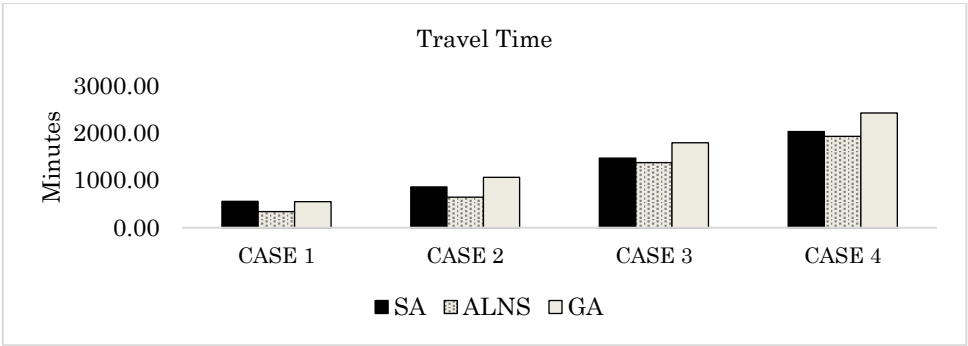

Figure 6. Comparison of minimum total travel time

Figure 6 explains that travel time is obtained from the arrival time of each vehicle so that it aligns time windows or operational times at the destination point. Compared with the time windows among destinations, case 1 and case 2 did not have routing that violated the expiration and operational time for each destination since some destinations have 15 and 24 hours per day. However, case 3 and 4 had a high travel time, over 24 hours, but still tolerance and did not have more than 3 days of expiration time on average. That issue might be caused by a limited number of vehicles for case 3 and 4. Based on figure 6 the results reveal that ALNS generally maintains consistently lower travel times in all cases, suggesting its efficiency in optimizing travel time in these scenarios. This indicates that ALNS often provide more optimal solutions for reducing travel times.

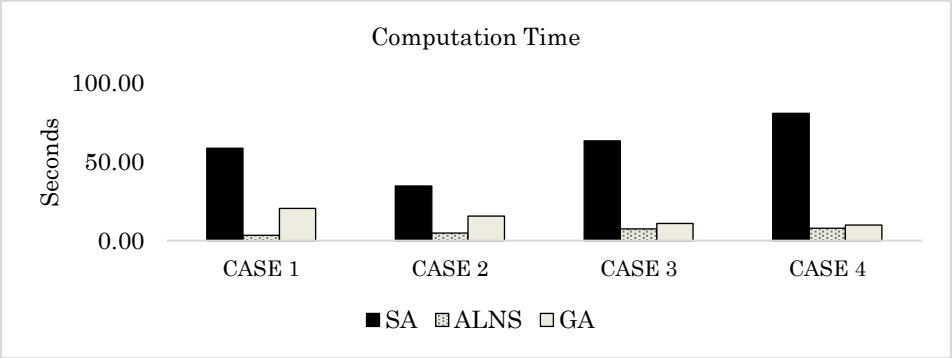

Figure 7. Comparison of minimum total computation time

Based on Figure 7, the bar chart clearly illustrates varying efficiencies of the methods in computational performance. In Case 4, GA and ALNS show similar, low computation times, while GA significantly exceeds these, indicating a possibly less efficient computational process for this case. Moving to all cases, ALNS displays the lowest computation times, suggesting its algorithms may be particularly optimized for efficient computation in these scenarios. GA, while showing improvement, still requires longer computation times. In Case 4, however, ALNS and GA are almost equal in their computation times, which are considerably less than that of SA.

The minimum distance comparison between the SA, ALNS, and GA methods in the previous sub-chapter shows that the ALNS method produces the shortest distance, followed by the SA method. The results of the two methods are close enough to validate the algorithms against each other. Interestingly, the results obtained with the GA method show almost double the minimum distribution distance compared to the SA and ALNS methods. This anomaly may be due to non-optimal GA parameters, indicating a need for further experimentation to achieve a minimum distance comparable to the SA and ALNS results. Similarly, the travel time results for the three methods correlate with the distribution distance and the loading/unloading service time.

The solutions obtained using the SA and ALNS methods differ significantly from the GA method but not drastically from each other. On average, the total distance using the SA and ALNS methods shows a difference of nearly 100 km, and the total travel time differs by about 100 minutes. This significant difference impacts the distribution of goods by reducing operational costs. However, the computation time comparison between the SA and ALNS methods shows that ALNS is far superior. The SA method requires an average computation time, demonstrating that the ALNS method is more efficient in producing better solutions in a shorter time. Figure 8 presents an example of the route obtained by ALNS in case 1.

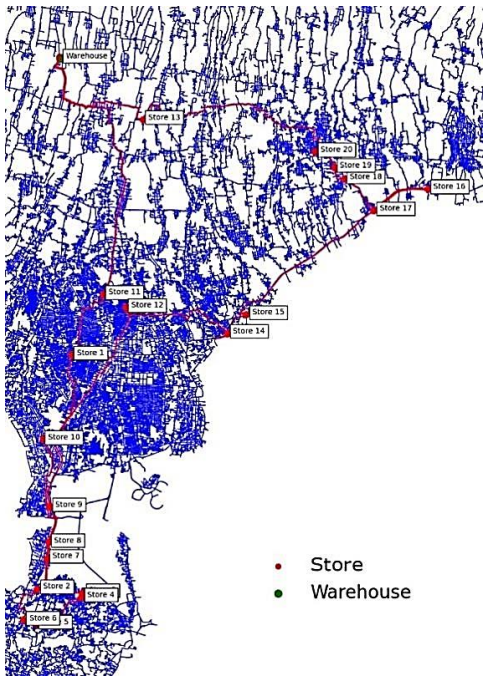

Figure 8. Example routing of case 1 with 20 stores

The analysis of variance (One-Way ANOVA) and Post Hoc Tukey tests conducted on four different cases reveals significant differences among the three tested method including Simulated Annealing (SA), Adaptive Large Neighborhood Search (ALNS), and Genetic Algorithm (GA) across three metrics consist of Travel Distance, Travel Time, and Computation Time. Based on the ANOVA result the exceedingly small p-values (0.000) in each case indicate that the differences between these methods are statistically significant. Tables 9-12 present the results of one-way ANOVA and post hoc Tukey for case 1, 2, 3, and 4, respectively.

Table 9. Case 1 result of one-way ANOVA and post hoc Tukey

	Case	1 (Travel distance)	Case	1 (Travel time)	Case	1 (Computation time)
	p-value	0.000	p-value	0.000	p-value	0.000
Method	Mean	Grouping	Mean	Grouping	Mean	Grouping
SA	296.99	A	556.99	A	58.650	A
ALNS	128.03	B	338.03	C	3.418	C
GA	298.36	A	548.36	B	20.475	B

According to the results of case 1, both SA and GA showed similar superiority over ALNS in terms of travel distance, both being categorized in group A. However, in travel time, SA significantly excelled, recording the fastest mean time, and placing in a separate group (A), while GA and ALNS were categorized in groups B and C. Based on the computational time, ALNS again stood out with the shortest computation time, followed by GA and SA.

Table 10. Case 2 result of one-way ANOVA and post hoc Tukey

	Case	2 (Travel distance)	Case	2 (Travel time)	Case	2 (Computation time)
	P-value	0.000	P-value	0.000	P-value	0.000
Method	Mean	Grouping	Mean	Grouping	Mean	Grouping
SA	300.58	B	860.58	B	34.776	A
ALNS	177.30	C	645.30	C	4.901	C
GA	502.76	A	1062.76	A	15.560	B

Almost similar to the previous case, case 2 depicts a similar pattern where GA consistently demonstrated less efficiency, being ranked in the top group (A) for both Travel Distance and Travel Time, indicating the longest distance and duration. Conversely, SA and ALNS were placed in more favorable groups, with ALNS showing the highest efficiency. For the computational time, ALNS maintained the lead with the quickest time.

Table 11. Case 3 result of one-way ANOVA and post hoc Tukey

	Case	3 (Travel distance)	Case	3 (Travel time)	Case	3 (Computation time)
	p-value	0.000	p-value	0.000	p-value	0.000
Method	Mean	Grouping	Mean	Grouping	Mean	Grouping
SA	403.48	B	1463.48	B	63.400	A
ALNS	331.11	C	1371.11	C	7.498	B
GA	731.30	A	1791.30	A	10.829	B

Based on Table 11, GA again showed the furthest Travel Distance and the longest Travel Time, indicating lower performance compared to SA and ALNS who exhibited more efficient metrics. Nevertheless, in Computational Time, ALNS presented the fastest time, showcasing higher computational efficiency than the other two methods.

Table 12. Case 4 result of one-way ANOVA and post hoc Tukey

	Case	4 (Travel distance)	Case	4 (Travel time)	Case	4 (Computation time)
	p-value	0.000	p-value	0.000	p-value	0.000
Method	Mean	Grouping	Mean	Grouping	Mean	Grouping
SA	522.98	B	2022.98	B	80.720	A
ALNS	424.56	C	1924.56	C	7.946	B
GA	916.66	A	2416.66	A	9.864	B

Case 4 reconfirmed this trend with GA requiring the shortest distance and longest travel time. Although SA recorded higher values in Computational Time, ALNS continued to exhibit the best computing time, followed

by GA in the same group. It means that the computational time between ALNS and GA is not statistically significant. These indicates that all three methods show robustness in generating consistent solutions. However, the tuning parameters for the SA and GA methods need to be refined to produce even better solutions.

The ANOVA results indicate significant differences among the three methods across all measured metrics, as evidenced by p-values all below 0.05. This means that the null hypothesis (H0), which states that there are no significant differences in the average performance of these methods, is rejected for all three metrics. Regarding Travel Distance, the ALNS method significantly differs from SA and GA, with a lower average travel distance, indicating more efficient performance. In contrast, SA and GA do not show significant differences in this metric and are grouped together in Group A. For Travel Time, the three methods exhibit significant differences from each other. GA has the highest travel time, followed by SA, while ALNS shows the lowest travel time, meaning this method is significantly faster in terms of travel time.

Finally, in the Computation Time metric, the results show that ALNS has a much lower computation time compared to SA and GA, which are grouped into Groups B and C, respectively. ALNS proves to be more efficient in terms of computation time, demonstrating the method's capability to complete calculations quickly. Overall, the rejection of the null hypothesis across all three metrics demonstrates that ALNS has significant advantages over SA and GA in two of the three metrics analyzed. However, all three methods show significant differences when compared to each other.

According to the results of One-Way ANOVA and Tukey post-hoc tests, Adaptive Large Neighborhood Search (ALNS) showed superior performance compared to Simulated Annealing (SA) and Genetic Algorithm (GA) in all cases analyzed. ALNS consistently produced lower average values for distance traveled and time traveled, while having a much faster computation time compared to the other two methods. This shows that ALNS can provide a more efficient solution in terms of saving distance and time traveled, while solving the problem with a shorter computation time, making it the best choice in this optimization.

In contrast, SA, and GA, although sometimes showing good results in some categories, tend to be slower and less efficient in time and distance traveled. GA often came out on top in terms of distance and time, while SA excelled in some cases but was still slower in terms of computation compared to ALNS. Thus, ALNS is the most recommended method to use, as it provides optimal results with high efficiency in all aspects analyzed.

Conclusions

This study tackles these the CVRPTW using a case study of bread distribution. The main objectives are to minimize the total distance traveled by all vehicles of four different cases and each cases provide 10 experiments, ensure timely fulfillment of all delivery demands, and offer recommendations for companies to optimize their delivery routes and improve delivery times. Three methods are developed: SA, ALNS, and GA. Parameter selection was conducted using the Design of Experiment (DOE). There are four cases that have different number of stores, the first case is 20 stores, the second case is 50 stores, the third case is 100 stores, and the fourth case is 150 stores. The parameter selection process with DOE was conducted using the fourth case.

Results from the DOE for the SA method, the best parameters were determined to be $T0 = 2000$ and $\alpha = 0.99$. For the ALNS method, the optimal parameters were $T = 1000$ and an upper DoD limit of 0.5. The best parameters for the GA method were a population size of 1000, 500 generations, a crossover rate of 0.8, and a mutation rate of 0.2. Using these parameters yielded the smallest average minimum distribution distances.

ALNS tends to offer the most efficient solutions in terms of distance and travel time in most cases, but at the cost of longer computation times, which may be a consideration in applications requiring rapid response. SA provides a balance between computational time efficiency and reasonably good outcomes in travel distance and time, making it a considerable option depending on specific usage needs. GA, meanwhile, might be less favored due to consistently lower performance in travel efficiency. The choice of method will heavily depend on the priorities between time, distance, and available computational resources.

The comparison between the SA, ALNS, and GA models in this study shows that the ALNS method provides the smallest distance and shortest travel time in all four cases. Using the four cases in order (e.g. case 1, 2, 3, and 4), the ALNS method produces the average smallest distance with the minimum travel distance of 128.03 km, 177.30 km, 331.11 km, and 414.86 km, compared to the SA method with a minimum travel distance (i.e. in

average) of 296.99 km, 300.58 km, 403.48 km, and 495.29 km and the GA method with a minimum travel distance (i.e. in average) of 298.36 km, 502.76 km, 731.32 km, and 869.51 km. According to the minimum travel distance result of three approximation methods, on average, ALNS had a more efficient 30.02% than SA and 57.21% than GA. Statistical analysis to measure robustness for the fourth case reveals that the ALNS method significantly outperforms SA and GA in both Travel Distance and Computation Time, demonstrating greater efficiency. For Travel Time, all three methods differ significantly, with ALNS showing the fastest performance. Overall, ALNS is the most effective method among the three evaluated.

The superior performance of the ALNS method can be attributed to its better diversification capability compared to SA and GA. This means that ALNS can explore the search space more broadly and effectively, making it more likely to find optimal solutions. Additionally, ALNS can easily adapt to solve various types of problems due to its more extensive parameter set compared to the other two methods. However, the comparison of the SA, ALNS, and GA methods in this study needs to be tested with different data to determine if ALNS consistently outperforms SA and GA. Similar problems involving route determination with additional variables and different data might yield different results. Therefore, although the ALNS model in this study outperforms SA and GA, the results may change with different data or additional variables.

Based on the discussion and analysis, the suggestion for further research is to improve the process of the algorithm so that it does not get trapped in local optima, as in evaluating fitness, especially in GA. Fitness evaluation can be done adaptively so that the exploration and exploitation process of finding solutions becomes more effective.

Acknowledgment

This study is funded by Badan Riset dan Inovasi Nasional (BRIN) and Lembaga Pengelola Dana Pendidikan (LPDP) Republic of Indonesia, grant Riset dan Inovasi untuk Indonesia Maju (RIIM) No. 172/IV/KS/11/2023 and 6815/UN1/DITLIT/Dit-Lit/KP.01.03/2023.

References

[1] A. Masache and F. Mangwanya, "Application of vehicle routing in a bread delivery system at Bakers Inn 15th Avenue Factory in Bulawayo," *International Journal of Engineering & Scientific Research*, Vol. 1, No. 2, pp. 1-17, Dec. 2013. https://www.ijmra.us/project%20doc/IJESR_DECEMBER2013/IJMRA-4339.pdf

[2] A. Estrada-Moreno, C. Fikar, A. A. Juan, and P. Hirsch, "A biased-randomized algorithm for redistribution of perishable food inventories in supermarket chains," *International Transactions in Operational Research*, vol. 26, no. 6, pp. 2077–2095, Apr. 2019, doi: https://doi.org/10.1111/itor.12668.

[3] A. A. N. P. Redi *et al.*., "Simulated annealing algorithm for solving the capacitated vehicle routing problem: a case study of pharmaceutical distribution," *Jurnal Sistem dan Manajemen Industri,* vol. 4, no. 1, pp. 41–49, Jul. 2020, doi: https://doi.org/10.30656/jsmi.v4i1.2215.

[4] K. Bujel, F. Lai, M. Szczecinski, W. So, and M. Fernandez, "Solving high volume capacitated vehicle routing problem with time windows using recursive-DBSCAN clustering algorithm," arXiv preprint,1812.02300. Dec. 2018, doi: https://doi.org/10.48550/arXiv.1812.02300

[5] K. A. Putri, N. L. Rachmawati, M. Lusiani, and A. A. N. P. Redi, "Genetic algorithm with cluster-first route second to solve the capacitated vehicle routing problem with time windows: A case study", *Jurnal Teknik Industri: Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 23, no. 1, pp. 75–82, May 2021. doi: https://doi.org/10.9744/jti.23.1.75-82

[6] D. M. Utama, W. O. N. Safitri, and A. K. Garside, "A modified camel algorithm for optimizing green vehicle routing problem with time windows", *Jurnal Teknik Industri: Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 24, no. 1, pp. 23–36, May 2022. doi: https://doi.org/10.9744/jti.24.1.23-36.

[7] A. A. Londoño, W. G. González, O. D. Montoya, and J. W. Escobar, "A hybrid heuristic approach for the multi-objective multi depot vehicle routing problem," *International Journal of Industrial Engineering Computations*, vol. 15, no. 1, pp. 337–354, Jan. 2024, doi: https://doi.org/10.5267/j.ijiec.2023.9.006.

[8] S. Wang, X. Zhu, P. Shang, X. Lin, L. Yang, and Lóránt Tavasszy, "Two-echelon multi-commodity multimodal vehicle routing problem considering user heterogeneity in city logistics," *Expert Systems with Applications*, vol. 252, pp. 124141–124141, Oct. 2024, doi: https://doi.org/10.1016/j.eswa.2024.124141.

[9] S. Pingale, A. Kaur, and R. Agarwal, "Collaborative last mile delivery: A two-echelon vehicle routing model with collaboration points," *Expert Systems with Applications*, vol. 252, pp. 124164–124164, Oct. 2024, doi: https://doi.org/10.1016/j.eswa.2024.124164.

[10] R. J. Kuo, E. Edbert, F. E. Zulvia, and S.-H. Lu, "Applying NSGA-II to vehicle routing problem with drones considering makespan and carbon emission," *Expert Systems with Applications*, vol. 221, p. 119777, Jul. 2023, doi: https://doi.org/10.1016/j.eswa.2023.119777.

[11] C. Natalia, V. Triyanti, G. Setiawan, and M. Haryanto, "Completion of capacitated vehicle routing problem (CVRP) and capacitated vehicle routing problem with time windows (CVRPTW) using bee algorithm approach to optimize waste picking transportation problem," *Journal of Modern Manufacturing Systems and Technology*, vol. 5, no. 2, pp. 69–77, Aug. 2021, doi: https://doi.org/10.15282/jmmst.v5i2.6855.

[12] M. L. Mutar, M. A. Burhanuddin, A. S. Hameed, N. Yusof, and H. J. Mutashar, "An efficient improvement of ant colony system algorithm for handling capacity vehicle routing problem," *International Journal of Industrial Engineering Computations*, pp. 549–564, 2020, doi: https://doi.org/10.5267/j.ijiec.2020.4.006.

[13] C. Wang, D. Mu, F. Zhao, and J. W. Sutherland, "A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup–delivery and time windows," *Computers & Industrial Engineering*, vol. 83, pp. 111–122, May 2015, doi: https://doi.org/10.1016/j.cie.2015.02.005.

[14] L. Wei, Z. Zhang, D. Zhang, and S. C. H. Leung, "A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints," *European Journal of Operational Research*, vol. 265, no. 3, pp. 843–859, Mar. 2018, doi: https://doi.org/10.1016/j.ejor.2017.08.035.

[15] A. Andriansyah, R. Novatama, and P. D. Sentia, "algoritma simulated annealing untuk menentukan rute kendaraan heterogen (studi kasus)," *Jurnal Teknologi Informasi dan Ilmu Komputer*, Oct. 2020, doi: https://doi.org/10.25126/jtiik.2020752018.

[16] S. D. Juniarto, E. S. Martiana, P. Fariza, and Ira, "Optimasi distribusi barang berdasarkan rute dan daya tampung menggunakan metode simulated annealing," *EEPIS Final Project Politeknik Elektronika Negeri Surabaya,* 2011. http://repo.pens.ac.id/id/eprint/1133

[17] R. L. Sinaga, "Algoritma simulated annealing untuk menyelesaikan multi depot vehicle routing problem dengan variabel travel time," Doctoral dissertation, Institut Technology Sepuluh Nopember, 2021.

[18] E. Wirdianto, and B. S. Jonrinaldi, "Penerapan algoritma simulated annealing pada penjadwalan distribusi produk," *Jurnal Optimasi Sistem Industri*, vol. 7, no. 1, pp. 7-20, 2007. http://repo.unand.ac.id/3453/1/2_Jurnal-TIUA-_Eri-Jon-Betris__Edisi_13_hal-7-20.pdf

[19] S. T. Windras Mara, R. Norcahyo, P. Jodiawan, L. Lusiantoro, and A. P. Rifai, "A survey of adaptive large neighborhood search algorithms and applications," *Computers & Operations Research*, vol. 146, p. 105903, Oct. 2022, doi: https://doi.org/10.1016/j.cor.2022.105903.

[20] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992, doi: 10.2307/24939139.

[21] I. D. Safira, A. Pradjaningsih, and A. Riski, "Penerapan kombinasi genetic algorithm dan iterated local search pada multi-depot capacitated vehicle routing problem," *Jurnal Matematika*, vol. 11, no. 1, p. 41, Aug. 2021, doi: 10.24843/jmat.2021.v11.i01.p135.

[22] B. Tuffin, and P. L'Écuyer, *Monte Carlo and Quasi-Monte Carlo Methods*, Springer International Publishing, Vol. 324, p. 539, 2020.