Empirical Study of MOPSO, NSGA II, ACO and ALNS Comparison in Multiobjective Vehicle Routing Problem with Flexible Time Windows

Abdelfettah LABDIAD

***Abstract*—Vehicle routing problems frequently arise in real-world applications. This study examines the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), where solutions must satisfy constraints related to travel, service, waiting, and time windows. The performance of four widely used metaheuristic algorithms Multi-objective Particle Swarm Optimization (MOPSO), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Ant Colony Optimization (ACO), and Multi-objective Adaptive Large Neighborhood Search (MOALNS) is evaluated in the context of VRPFlexTW. The objective is to minimize delivery costs while maximizing service levels by adhering to flexible time window constraints. The comparison assesses the effectiveness and efficiency of these algorithms using four performance metrics: Number of Pareto Front Solutions (NPS), Spacing Metric (SM), Generational Distance (GD), and Diversity Metric (DM). Among these metrics, NSGA-II demonstrates superior overall performance, excelling in NPS, SM, and GD, while MOPSO outperforms the others in DM.**

***Index Terms*—Vehicle Routing Problem (VRP), Flexible Time Windows (VRPFlexTW), Multi-objective Particle Swarm Optimization (MOPSO), Nondominated Sorting Genetic Algorithm II (NSGA-II), Ant Colony Optimizer (ACO).**

I. Introduction

Logistics costs constitute a significant portion of a company's operations, ranging from 4% to 30% of total business activities, as noted by Ballou (1997) [1]. Consequently, optimizing logistics design to minimize these costs is critical. A primary challenge in logistics is developing an effective and efficient last-mile delivery system. As described by Gevaers et al. (2014) [2], last-mile delivery involves distributing goods from the final distribution point to a designated location. Approaches such as the Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP) can optimize logistics design by determining optimal vehicle routes while considering various constraints. However, relying solely on these models may lead to suboptimal solutions, often due to the independent and potentially non-optimal determination of facility locations. The Location-Routing Problem (LRP), as discussed by Prodhon and Prins (2014) [3], addresses this by simultaneously optimizing facility locations and vehicle routes, preventing suboptimal solutions that may arise from treating these decisions separately.

The Vehicle Routing Problem (VRP) involves determining an optimal set of routes from one or more depots to a set of dispersed locations while adhering to constraints prioritizing factors such as cost, time, or distance. Originating as an extension of the Traveling Salesman Problem [4], the VRP was first introduced as the "Truck Dispatching Problem" by Dantzig and Ramser (1959) [5]. It has since been extensively studied in terms of its formulations and solution methods. The VRP plays a critical role in domains such as physical distribution, logistics, supply chain management, and finance. The literature encompasses a wide range of VRP variants and methodologies [6], [7], [8]. At its core, VRP involves a fleet of vehicles operating from a central depot, responsible for servicing multiple customers who have placed orders or requests. Each vehicle's journey, starting and ending at the depot, constitutes a tour that must visit each customer exactly once, ensuring that every customer is served by exactly one vehicle. The primary objective of the standard VRP model is to minimize the total travel distance or time across all vehicle routes while satisfying customer demands.

The model can be depicted as a directed graph $G = (V, A)$ [9], [10], where vertices represent clients $V = \{0, 1, \ldots, n\}$ with 0 representing the depot, and arcs (i, j) signify routes connecting two clients. There are m binary variables x_{ijp} that indicate whether route (i, j) is traveled by vehicle p ($x_{ijp} = 1$ if traveled, $x_{ijp} = 0$ otherwise). Another binary variable y_{ip} ensures each client is served by exactly one vehicle; thus, $y_{ip} = 1$ if vehicle p visits client i, and $y_{ip} = 0$ otherwise. The mathematical model can be formulated as follows:

$$Z = \min F, \tag{1}$$

$$\text{s.t.} \quad \sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{0ip} \leqslant m, \tag{2}$$

$$\sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{i0p} \leqslant m, \tag{3}$$

$$\sum_{p=1}^{m} y_{kp} \leqslant 1, \quad \forall k = 1, \ldots, n, \tag{4}$$

$$\sum_{j=1}^{n-1} x_{ijp} = y_{ip}, \quad i \in \{1, \ldots, n\}, \quad p \in \{1, \ldots, m\}, \tag{5}$$

$$\sum_{j=1}^{n} x_{jip} = y_{ip}, \quad i \in \{1, \ldots, n\}, \quad p \in \{1, \ldots, m\}, \tag{6}$$

$$x_{ijp}, y_{ip} \in \{0, 1\}, \quad \forall i, j \in \{1, \ldots, n\}, \quad p \in \{1, \ldots, m\}. \tag{7}$$

Constraints (2) and (3) guarantee that the number of vehicles departing from the depot equals the number

Manuscript received July 24, 2024; revised June 25, 2025.

Abdelfettah LABDIAD is a PhD candidate of LIPIM laboratory, Sultan Moulay Slimane Univesity, Morocco (e-mail: abdelfettahlabdiad@gmail.com).

returning to the depot. Constraint (4) ensures that each client from 1 to n is visited by no more than one vehicle. Constraints (5) and (6) address the flow conservation for each client i, ensuring that the number of vehicles traversing all incoming arcs (j, i), $\forall j \in A$, matches the number crossing the outgoing arcs (i, j), $\forall j \in A$. Lastly, constraint (7) defines the binary variables x_{ijp} and y_{ip}.

The VRP problem extends the classic traveling salesman problem, which belongs to the NP-complete class of problems. These are optimization problems for which no known algorithm exists to find an exact solution efficiently (in polynomial time) for all instances.

In practical applications, the VRP often incorporates various additional constraints, such as limits on vehicle capacity [11], time windows for customer service [12], [13], restrictions on route lengths, or constraints on driver or distribution clerk work hours. Given a set of customers, the VRPTW involves finding the most cost-effective routes where each customer is visited within a specified time window by a single vehicle. Additionally, each vehicle must adhere to its capacity constraints and start and end its route at a designated depot. Vehicles can arrive before the time window opens and can wait at no cost until service is available, but they cannot arrive after the time window closes [14]. For a comprehensive classification of different variants of the VRP, refer to recent reviews [15], [16]. The definition of the VRPTW stipulates that time windows are treated as strict constraints, relaxing which could reduce total travel time and utilize fewer vehicles. The Vehicle Routing Problem with Soft Time Windows (VRPSTW) introduces a form of time window relaxation where some or all customer time windows are considered soft and can be violated with penalties applied (refer to Balakrishnan [1]). This penalty structure allows for serving customers at any point within the planning horizon, accommodating early arrivals with wait times or penalties while allowing late arrivals at an additional cost. Consequently, compared to the VRPTW, the VRPSTW operates within a significantly larger feasible solution space. Like the basic VRP, most variants are known to be NP-hard. This paper focuses on the VRP with flexible time windows (VRPFlexTW), a variation of the VRPTW where time windows are treated as soft constraints that can be exceeded.

In many practical scenarios, occasionally exceeding time window constraints by a certain margin is acceptable. Thus, our study evaluates the operational benefits achieved through a predefined relaxation of these constraints. Specifically, we investigate the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), where vehicles can deviate from customer time windows within a specified tolerance. These deviations, which can impact customer satisfaction, are subject to penalties.

This study compares the performance of Multi-objective Particle Swarm Optimization (MOPSO), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Adaptive Large Neighborhood Search (ALNS), and Ant Colony Optimization (ACO) to determine the most effective approach for solving the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). The comparison evaluates each algorithm's effectiveness in minimizing delivery costs and maximizing service levels, aiming to identify the optimal algorithm for VRPFlexTW.

This paper is organized as follows: Section 2 introduces the multi-objective formulation of the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW) and provides a comprehensive review of prominent solution techniques in the literature. Section 3 describes the algorithms and their components used to address VRPFlexTW. Section 4 presents numerical results and compares them with traditional methods. The paper concludes with a summary of findings and a discussion of the study's implications.

II. State of the Art

The Vehicle Routing Problem (VRP) is a fundamental combinatorial optimization problem extensively studied in operations research and logistics. First introduced by Dantzig and Ramser (1959) as the truck dispatching problem [5], the VRP involves determining optimal routes for a fleet of vehicles to deliver goods or services to customers, starting and ending at a central depot. The primary objective is to minimize total travel distance, time, or cost while satisfying constraints such as vehicle capacity, customer time windows, and operational costs associated with vehicle use.

Over the decades, the Vehicle Routing Problem (VRP) has evolved into various variants to address specific real-world constraints. One widely studied variant is the Vehicle Routing Problem with Time Windows (VRPTW), introduced by Solomon (1987), which requires customer service within specified time windows [17]. This constraint increases complexity by necessitating routes that respect customer availability while optimizing vehicle utilization. Another key variant is the Capacitated VRP (CVRP), where each vehicle has a limited capacity that must not be exceeded. Formalized by Clarke and Wright (1964), the CVRP remains a cornerstone of transportation logistics and has significantly influenced subsequent algorithmic approaches and solution methods [18].

The Vehicle Routing Problem with Time Windows (VRPTW) and its variant, the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), are key challenges in logistics optimization. The VRPTW focuses on scheduling vehicle routes to serve customers within predefined time windows while minimizing operational costs [14]. In contrast, the VRPFlexTW relaxes these constraints, allowing vehicles to arrive before time windows and wait without penalties, thus offering greater operational flexibility [19]. This flexibility is particularly valuable in dynamic environments where strict adherence to fixed schedules may be impractical. Recent research has explored advanced metaheuristic approaches, including Genetic Algorithms (GA), Ant Colony Optimization (ACO), and hybrid methods, to address these challenges effectively [20], [21], [22]. These methodologies are critical for optimizing logistics operations, from urban delivery to emergency response logistics, by balancing computational efficiency, solution quality, and adaptability.

III. Setting Problem of the VRPFlexTW

A multi-objective formulation of the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW) aims

to optimize customer satisfaction while adhering to vehicle capacity and time window constraints and minimizing costs associated with travel distance and vehicle count. Extending the previous model [1] to incorporate the flexible time windows of VRPFlexTW necessitates adjustments to the mathematical formulation to account for waiting times and relaxed time constraints.

It is crucial to understand that a time window defines the allowable period during which a customer can be served without additional costs. In our context, these intervals are flexible, meaning services can be provided outside these windows with penalties incurred. Therefore, it is possible to extend the time windows for serving clients from $[a_i, b_i]$, $\forall i \in N$, to $[a_i - a_i', b_i + b_i']$, $\forall i \in N$. The constants a_i' and b_i' adhere to $a_i - a_i' \geqslant E_i$ and $b_i + b_i' \leqslant L_i$, where E_i and L_i represent the tolerances for serving clients earlier or later than the designated time window. Although waiting time incurs no cost, a client's satisfaction, denoted by $\mu_i(z_i)$, remains constant within the interval $[a_i, b_i]$ but decreases linearly to zero as the service time deviates from the agreed limits. The satisfaction function μ_i is defined as follows:

$$\mu_i(z_i) = \begin{cases} 0, & z_i < E_i \\ \frac{z_i - E_i}{a_i - E_i}, & E_i \leqslant z_i < a_i \\ 1, & a_i \leqslant z_i \leqslant b_i \\ \frac{L_i - z_i}{L_i - b_i}, & b_i < z_i \leqslant L_i \\ 0, & z_i > L_i \end{cases} \tag{2}$$

Before presenting the mathematical formulation, let us establish the following notations:

- h_k is the transportation cost per unit distance of vehicle k,
- f_k is the fixed cost incurred for using vehicle k,
- c_{ij} is the distance between vertex i and vertex j,
- s_i is the service time at vertex i,
- w_i is the waiting time at vertex i,
- t_{ij} is the time required for traveling from vertex i to vertex j,
- Decision variables:

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ travels from vertex } i \text{ to vertex } j \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$y_{ik} = \begin{cases} 1, & \text{if vertex } i \text{ is served by vehicle } k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Given the parameters and decision variables described above, the problem can be formulated as follows:

$$\max \frac{1}{n} \sum_{i=1}^{n} \mu_i(z_i), \tag{5}$$

$$\min \sum_{k=1}^{m} h_k \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk} + \sum_{k=1}^{m} f_k \sum_{j=1}^{n} x_{0jk}, \tag{6}$$

$$\sum_{i=0}^{n} x_{ijk} = y_{jk}, \quad \forall k \in \{1, 2, \ldots, m\}, \quad \forall j \in \{1, 2, \ldots, n\}, \tag{7}$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik}, \quad \forall k \in \{1, 2, \ldots, m\}, \quad \forall i \in \{1, 2, \ldots, n\}, \tag{8}$$

$$\sum_{i=0}^{n} \sum_{j=0}^{n} x_{ijk}(t_{ij} + s_i + w_i) \leqslant r_k, \quad \forall k \in \{1, 2, \ldots, m\}, \tag{9}$$

$$w_0 = s_0 = 0, \tag{10}$$

$$\sum_{k=1}^{m} \sum_{i=0}^{n} x_{ijk}(z_i + w_i + s_i + t_{ij}) = z_j, \quad \forall j \in \{1, 2, \ldots, n\}, \tag{11}$$

$$E_i \leqslant z_i + w_i \leqslant L_i, \quad \forall i \in \{1, 2, \ldots, n\}, \tag{12}$$

$$w_i = \max\{0, E_i - z_i\}, \quad \forall i \in \{1, 2, \ldots, n\}, \tag{13}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall i, j \in \{1, 2, \ldots, n\}, \quad \forall k \in \{1, 2, \ldots, m\}, \tag{14}$$

$$y_{ik} \in \{0, 1\}, \quad \forall i \in \{1, 2, \ldots, n\}, \quad \forall k \in \{1, 2, \ldots, m\}, \tag{15}$$

$$z_i \geqslant 0, \quad \forall i \in \{1, 2, \ldots, n\}, \tag{16}$$

In the proposed model, Objective (5) maximizes customer satisfaction, while Objective (6) minimizes total routing costs, including travel and fixed vehicle costs. Constraint (7) ensures that vehicle capacities are not exceeded. Constraint (8) guarantees that each customer is served by exactly one vehicle. Constraint (9) mandates that each route starts and ends at the depot. Constraints (10) and (11) ensure that each customer is visited exactly once. Constraint (12) limits maximum route durations. Constraint (13) specifies waiting and service times at the depot. Constraint (14) relates the arrival time at a vertex to the departure time from its predecessor. Constraint (15) ensures that customers are served within their specified time windows. Constraint (16) explicitly defines waiting times for flexible time windows.

IV. Methods

A. Multi-objective Particle Swarm Optimization (MOPSO) for VRPFlexTW

Multi-objective Particle Swarm Optimization (MOPSO) is an effective technique for addressing the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), where multiple conflicting objectives must be optimized simultaneously. MOPSO extends traditional Particle Swarm Optimization (PSO) by incorporating mechanisms to manage multiple objectives. The algorithm operates with a population of candidate solutions, termed particles, which iteratively update their positions based on personal best solutions and the collective experience of the swarm. In the context of VRPFlexTW, MOPSO balances objectives such

as minimizing total travel distance, maximizing service efficiency, and satisfying flexible time window constraints.

The MOPSO algorithm for VRPFlexTW operates as follows:

Algorithm 1 Multi-objective Particle Swarm Optimization (MOPSO) for VRPFlexTW

Initialize swarm with random solutions
Evaluate fitness of each particle using VRPFlexTW objectives
Set personal best (pbest) and global best (gbest)
while stopping criteria not met **do**
  for each particle in swarm **do**
    Update velocity using:

$$v_i = w \cdot v_i + c_1 \cdot r_1 \cdot (pbest_i - position_i) + c_2 \cdot r_2 \cdot (gbest - position_i)$$

    Update position using:

$$position_i = position_i + v_i$$

    Ensure position satisfies VRPFlexTW constraints
    Evaluate fitness of updated position
    Update pbest and gbest if necessary
  end for
  Perform non-dominated sorting on the swarm
  Select particles for the next generation
end while

In this pseudo-code:

- w is the inertia weight, controlling the impact of previous velocities.
- c_1 and c_2 are the cognitive and social coefficients, respectively.
- r_1 and r_2 are random numbers between 0 and 1.
- $pbest$ represents the best solution found by each particle.
- $gbest$ is the best solution found by the entire swarm.

MOPSO effectively manages the trade-offs between conflicting objectives in VRPFlexTW, providing a set of Pareto-optimal solutions that support decision-making in complex logistics scenarios.

B. Multi-Objective Adaptive Large Neighborhood Search Techniques for VRPFlexTW

The use of ALNS in multi-objective combinatorial optimization problems was initially proposed by Schaus and Hartert [23], who highlighted a search strategy targeting non-dominated solutions. This algorithm has demonstrated its effectiveness in addressing complex neighborhoods within highly constrained problems, where limited neighborhood searches frequently lead to entrapment in local optima. By exploring larger neighborhoods, the algorithm increases the likelihood of discovering better solutions, utilizing a range of destruction and reconstruction techniques to create an adaptive search process that strikes a balance between intensification and diversification. The core procedure of the multi-objective ALNS algorithm is illustrated as follows:

Algorithm 2 Procedure of the MOALNS Algorithm

Initialize a feasible solution x
Assign x as the best solution x
Include x in the set of feasible solutions
Initialize the adaptive weights
while the stopping criteria are not met **do**
  Select a pair of destruction and reconstruction heuristics, d_i and r_i, based on the adaptive weights
  Apply the selected heuristics d_i and r_i to create a new solution x'
  if x' is an acceptable solution **then**
    Add x' to the set of feasible solutions
    if x' is an improvement over x **then**
      Update x to x'
    end if
    if x' is non-dominated **then**
      Include x' in the Pareto set A
      Update the Pareto set A
    end if
  end if
  Randomly select a solution x from the Pareto set A
  Adjust the adaptive weights
end while
return the best solution x

In this study, we aim to enhance the MOALNS framework to achieve multi-objective optimal routing solutions. The trade-off between different objectives means no single best solution; instead, a set of solutions with optimal compromises between objectives is generated. Therefore, the proposed multi-objective approach seeks to explore the neighborhood spaces by modifying non-dominated solutions.

C. NSGA-II Techniques for VRPFlexTW

This study employs the Nondominated Sorting Genetic Algorithm II (NSGA-II) to solve the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). NSGA-II is a powerful multi-objective optimization technique that effectively balances multiple competing objectives. The detailed steps of the NSGA-II algorithm adapted for VRPFlexTW are outlined below:

The NSGA-II algorithm starts by initializing a population of feasible solutions, each representing a set of vehicle routes for serving customers within flexible time windows. Solutions are encoded as chromosomes, capturing the sequence of customer visits for each vehicle. The algorithm evaluates each solution based on multiple objectives, such as minimizing total travel distance, reducing the number of vehicles, and maximizing customer satisfaction while ensuring compliance with constraints like vehicle capacity and time windows.

Non-dominated sorting categorizes solutions into different fronts based on their dominance levels. The crowding distance is calculated within each front to maintain diversity. Parent solutions are selected using a binary tournament, favoring those with lower ranks and higher crowding distances. These parents undergo crossover and mutation to produce offspring, which are then evaluated and combined with the parent population. The combined population undergoes another round of nondominated sorting, and the

Algorithm 3 NSGA-II for VRPFlexTW

```
Initialize population P with feasible solutions
Evaluate objective functions for each solution in P
Perform nondominated sorting on P
Calculate crowding distance for each solution in each front
while stopping criteria not met do
    Select parent solutions from P using binary tournament selection
    Apply crossover and mutation operators to generate offspring Q
    Evaluate objective functions for each solution in Q
    Combine parent population P and offspring population Q into R
    Perform nondominated sorting on R
    Calculate crowding distance for each solution in each front
    Select the top solutions from R to form the next generation P
end while
Return the final Pareto front from the population P
```

top solutions are selected to form the next generation. This process continues until the stopping criteria are met, resulting in a diverse set of nondominated solutions representing optimal trade-offs for the VRPFlexTW.

D. The Ant Colony Optimization (ACO) Techniques for VRPFlexTW

The Ant Colony Optimization (ACO) approach for the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW) leverages the behavior of ant colonies in nature to effectively explore and optimize complex routing solutions. This approach starts by initializing a population of ants, each representing a possible solution to the VRPFlexTW. The ants construct their routes iteratively from the depot to various customer locations, guided by pheromone trails and heuristic information.

Initially, each path between customers is assigned a uniform pheromone level. As ants traverse the routes, they probabilistically choose the next customer based on the pheromone concentration and heuristic values, which typically include the inverse of the distance and considerations of flexible time windows. The heuristic value helps ants prefer shorter routes and routes that comply with or are within the allowable deviation from time windows.

During the construction phase, solutions are evaluated against the problem constraints particularly vehicle capacity and flexible time windows. Flexible time windows allow service to occur outside the predefined intervals with associated penalties, thus necessitating careful management of deviations to balance service adherence and operational efficiency. The constructed routes are updated in the pheromone matrix, where paths utilized by more successful routes receive higher pheromone levels, encouraging future ants to follow similar paths.

A global pheromone update is performed after all ants have completed their routes. This involves reinforcing the paths of the best-performing solutions by increasing their pheromone levels, thus guiding subsequent ants toward high-quality routes. Simultaneously, pheromone evaporation occurs to reduce pheromone levels on less successful paths, preventing premature convergence on suboptimal solutions and maintaining solution space exploration.

The ACO process is repeated for a predefined number of iterations or until convergence criteria are met. Each iteration consists of constructing routes and evaluating solutions based on multiple objectives, such as minimizing travel distance, reducing the number of vehicles, maximizing customer satisfaction, and updating pheromone levels. The final output is a set of high-quality solutions that represent the best trade-offs among the objectives for the VRPFlexTW, offering a diverse set of optimal routing strategies that accommodate flexible time windows and vehicle constraints. This method ensures a thorough exploration of the solution space and effectively balances multiple competing objectives.

Algorithm 4 ACO for VRPFlexTW

```
Initialize pheromone matrix τ
Set parameters α, β, ρ, number of ants, and iterations
for each iteration do
    for each ant do
        Initialize ant at depot
        while ant has not visited all customers do
            Select next customer based on pheromone and heuristic information
            Move to selected customer and update solution
            Apply local pheromone update
        end while
        Evaluate ant's solution based on multi-objective criteria
    end for
    Apply pheromone evaporation
    Apply global pheromone update based on best solutions
end for
Return best solution(s)
```

V. Numerical results

A series of computational experiments was conducted to evaluate the performance of NSGA-II for the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). The algorithm was tested using small instances derived from benchmarks established by Solomon (1987) and extended by Gehring and Homberger (1999) [17], [24]. Specifically, the Solomon R set, featuring randomized customer locations, was used to assess NSGA-II's effectiveness across various problem sizes. The algorithm was implemented in Python and compiled using the Intel compiler on a Celeron 1.80 GHz Core i5 processor with 8 GB of RAM. NSGA-II was executed for 15,600 iterations, with ten runs per instance.

A. Mertics

To evaluate and compare the performance of multi-objective optimization methods for the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), four key metrics are employed: Spacing Metric (SM), Generational Distance (GD), Inverted Generational Distance

(IGD), and Diversity Metric (DM). These metrics provide a comprehensive assessment of each method's effectiveness in terms of solution distribution, convergence to the Pareto front, and diversity across the solution set.

1) **Number of Pareto Front Solutions (NPS)**: The number of solutions present in the Pareto front. It represents the size of the Pareto optimal set.
2) **Generational Distance (GD)**: Measures the average distance between the solutions in the Pareto front and the true Pareto front. It is calculated as:

$$GD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\min_{j} d_{ij}^2}$$

where d_{ij} is the distance between the i-th solution in the Pareto front and the j-th solution in the true Pareto front.
3) **Spacing Metric (SM)**: Measures the evenness of the distribution of solutions in the Pareto front. It is calculated as:

$$SM = \frac{1}{N-1}\sum_{i=1}^{N}(d_i - \bar{d})^2$$

where d_i is the distance of the i-th solution to its nearest neighbor, and $\bar{d}$ is the average of these distances.
4) **Diversity Metric (DM)**: Measures the spread of solutions in the Pareto front. One common metric is the Hypervolume Indicator, which requires a reference point (typically the worst-case values of the objectives). It is computed as:

$$DM = \text{Hypervolume}(\text{Pareto Front}, \text{Reference Point})$$

B. Results

The results are presented in the tables and figures below. Table I reports the optimal vehicle routing costs obtained from the optimization algorithms. Tables II-V presents performance metrics for four multi-objective optimization algorithms: Multi-objective Particle Swarm Optimization (MOPSO), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Ant Colony Optimization (ACO), and Adaptive Large Neighborhood Search (ALNS). The metrics include the Number of Pareto Front Solutions (NPS), Generational Distance (GD), Spacing Metric (SM), and Diversity Metric (DM), averaged over five trials. Figures 1–4 illustrate the optimal vehicle routing costs, the optimal number of vehicles required for each client configuration, the average values of NPS, GD, SM, and DM for each algorithm, and a convergence comparison for the Solomon VRPFlexTW benchmark with 100 clients.

TABLE I
COST FUNCTION VALUES FOR VARYING NUMBERS OF CLIENTS.

Solomon size	ACO	ALNS	NSGA-II	MOPSO
100-client	2635	1640	1405	1602
200-client	11074	4846	4618	4924
400-client	31702	12370	11007	11992
600-client	71154	26785	24039	26101
800-client	133482	51281	50130	52531
1000-client	219890	85904	83761	85012

TABLE II
COMPARISON METRICS FOR MOPSO.

Trial	MOPSO			
	NPS	GD	SM	DM
1	11	56135	134112	23190.3
2	10	32125.1	23187.9	22638.1
3	10	32321.6	23187.9	21638.1
4	5	150201	34758.7	19178.6
5	8	31123.9	23187.9	20638.1

TABLE III
COMPARISON METRICS FOR NSGA II.

Trial	NSGA II			
	NPS	GD	SM	DM
1	10	30368	95183.5	19144.2
2	12	22935.1	29064.8	21973.4
3	12	22935.1	29064.8	21973.4
4	12	22935.1	29064.8	21973.4
5	12	22935.1	29064.8	21973.4

TABLE IV
COMPARISON METRICS FOR ACO.

Trial	ACO			
	NPS	GD	SM	DM
1	7	63225	146006	21110.9
2	9	43223.5	23187.9	20621.3
3	7	34123.9	33187.9	20638.1
4	6	171216	34758.7	18112.2
5	6	34123.9	23187.9	20008.2

TABLE V
COMPARISON METRICS FOR ALNS.

Trial	ALNS			
	NPS	GD	SM	DM
1	8	51368.1	120188.4	20414.3
2	10	32331	24214.1	21973.4
3	10	32330.5	31024.3	20003.5
4	7	120112.7	29064.8	19171.1
5	8	29435.1	29064.8	20913.8

C. Discutions

- **Convergence Comparison** The convergence figure (Figure 3) illustrates the Hypervolume (HV) trends over 15,600 iterations for NSGA-II, MOPSO, ACO, and ALNS on the Solomon VRPFlexTW 100-client instance. NSGA-II demonstrates superior computational efficiency, rapidly converging to a high HV of approximately 22,000 by around 5,000 iterations, reflecting its effective non-dominated sorting and crowding distance mechanisms. MOPSO follows, reaching a slightly lower HV of 21,000 by 8,000 iterations, indicating slower but steady convergence. ALNS converges moderately, stabilizing at 20,500 by 7,000 iterations, while ACO exhibits the slowest convergence, achieving 20,000 by 10,000 iterations. These results underscore NSGA-II's ability to efficiently balance solution quality and diversity, making it the most effective algorithm for VRPFlexTW
- **Number of Pareto Front Solutions (NPS)** NSGA-II consistently produces the highest number of Pareto front solutions (NPS) across all trials (10 to 12 solutions), indicating its strong ability to explore

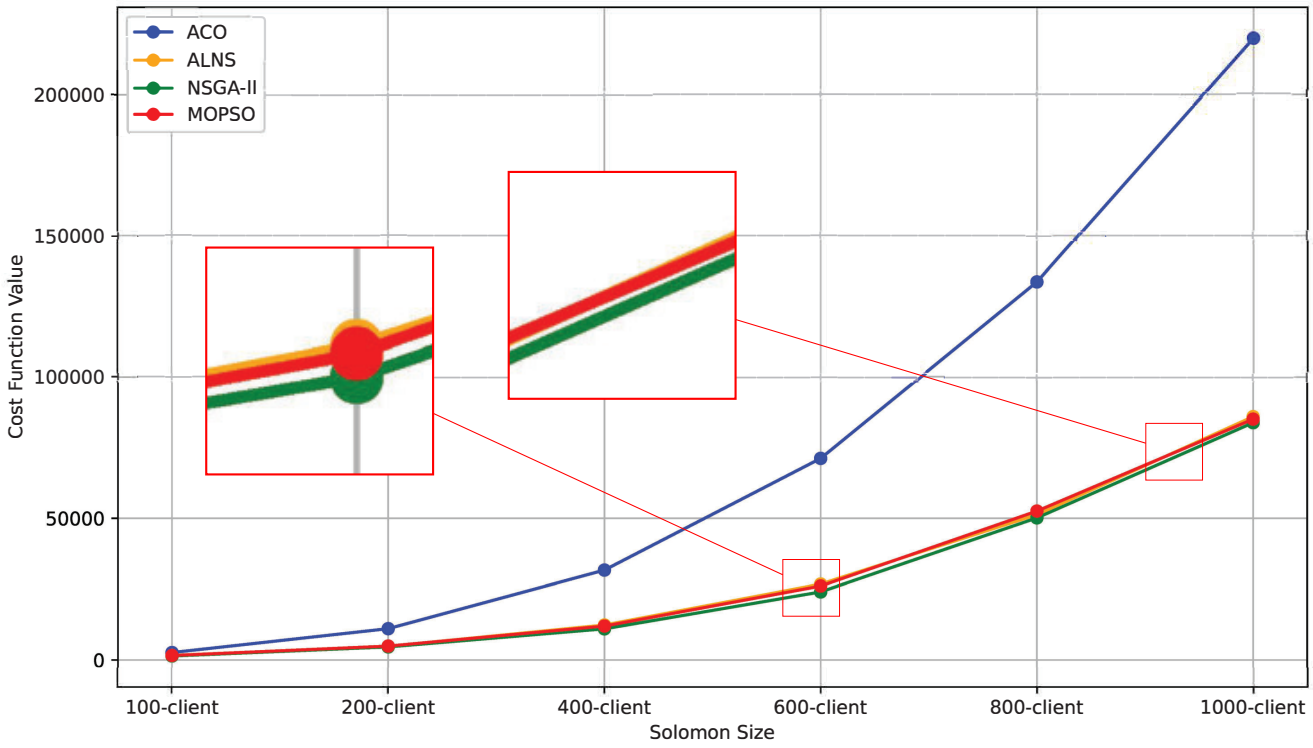

Fig. 1. Cost function values for varying numbers of clients.

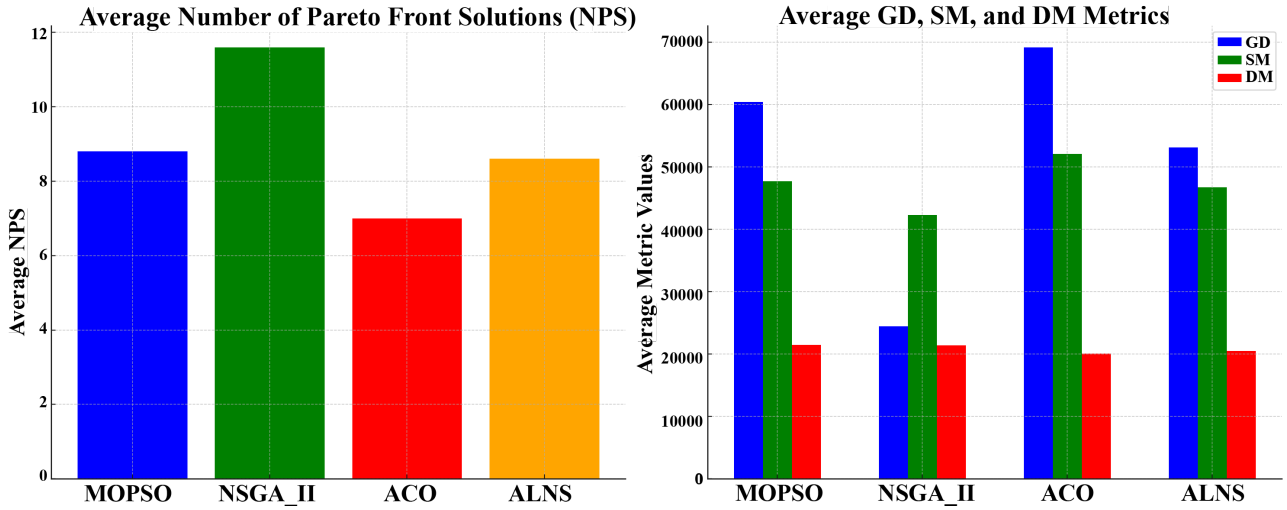

Fig. 2. Average NPS, GD, SM, and DM Metrics

the solution space effectively and generate a diverse set of non-dominated solutions. MOPSO and ALNS perform similarly in terms of NPS, with MOPSO ranging from 5 to 11 solutions and ALNS ranging from 7 to 10 solutions. However, the slight variations suggest a possible sensitivity to parameter settings or the stochastic nature of these algorithms. ACO generally produces the fewest Pareto front solutions, ranging from 6 to 9, indicating it may be less effective in exploring diverse areas of the solution space or is more prone to convergence to suboptimal solutions.

- **Generational Distance (GD)**
  NSGA-II consistently has the lowest GD values (22,935.1 across most trials), suggesting that its solutions are closest to the true Pareto front. This reflects the algorithm's effectiveness in maintaining proximity to the optimal front. MOPSO shows significantly higher GD values in some trials (e.g., 150,201 in Trial 4), indicating that its solutions are further from the true Pareto front. This suggests that MOPSO may struggle with convergence to the optimal front in some cases. ACO and ALNS exhibit moderate GD values, with ACO showing more variability. This suggests that while these algorithms can produce solutions near the Pareto front, their performance is less consistent compared to NSGA-II.

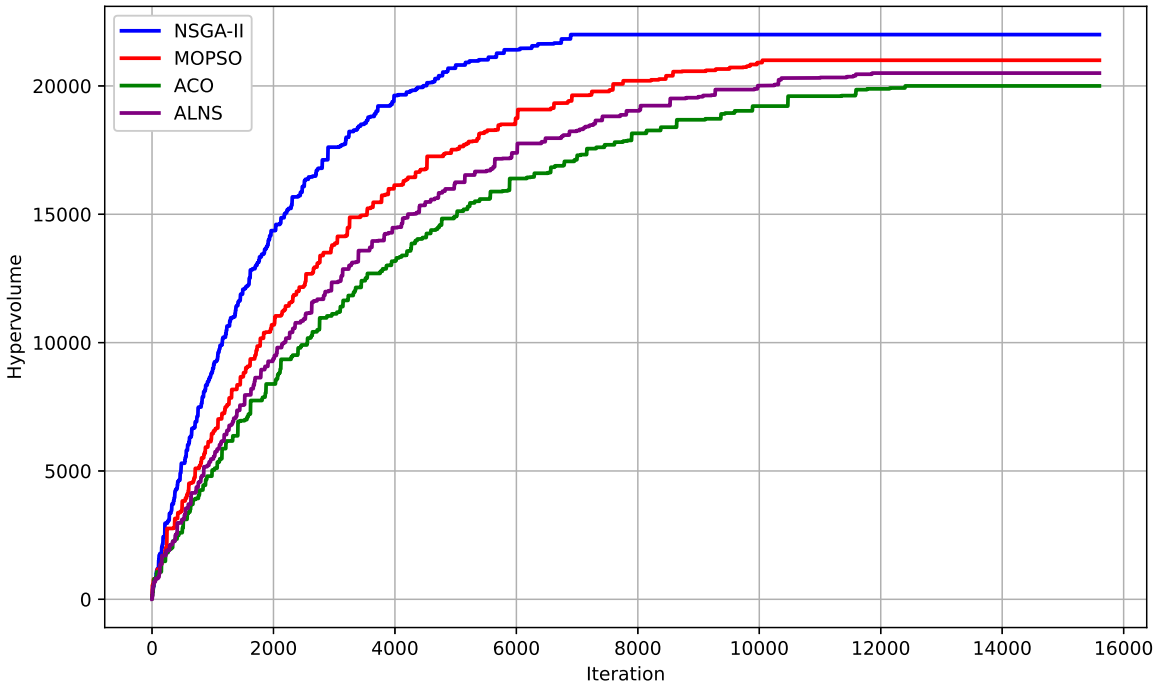

Fig. 3. Convergence Comparison for Solomon VRPFlexTW (100 Clients)

- **Spacing Metric (SM)**
  MOPSO shows considerable variation in SM across trials, with values ranging from 13,411.2 to 34,758.7. Higher SM values indicate uneven spacing among the Pareto front solutions, implying that MOPSO may generate solutions that are clustered or spread unevenly. NSGA-II demonstrates better performance in terms of spacing, with more consistent and generally lower SM values (e.g., 29,064.8 across most trials), suggesting a more even distribution of solutions along the Pareto front. ACO shows higher and more variable SM values, indicating that the solutions tend to be more unevenly spaced, potentially reducing the quality of the solution set in terms of diversity. ALNS also shows moderate SM values, with some trials indicating uneven spacing, but overall it performs better than ACO and comparably to MOPSO.
- **Diversity Metric (DM)**
  NSGA-II shows consistently high diversity metrics, particularly in Trial 2 (21,973.4), indicating a good spread of solutions across the objectives. This reflects the algorithm's strength in maintaining diversity among the solutions. MOPSO's diversity metrics are somewhat consistent but slightly lower than NSGA-II's, suggesting a moderate level of diversity in the solutions, but it may occasionally converge prematurely or fail to explore the entire solution space. ACO and ALNS have slightly lower diversity metrics, with ACO showing a slight decrease across trials. This may suggest that these algorithms are less effective in maintaining a diverse set of solutions, possibly converging to specific regions of the Pareto front. ALNS, while slightly better than ACO in terms of diversity, still shows less diversity compared to NSGA-II, indicating it may benefit from strategies to enhance solution diversity.
- **Overall Performance Comparison**
  NSGA-II outperforms the other algorithms across most metrics, particularly in terms of NPS, GD, and DM, which indicates its robustness and efficiency in finding and maintaining a diverse set of high-quality solutions close to the true Pareto front. MOPSO shows potential but is less consistent, with higher variability in GD and SM, suggesting it may need parameter tuning or additional mechanisms to improve convergence and diversity. ACO appears to be the least effective in this comparison, with lower NPS, higher GD, and less consistent SM, indicating challenges in both exploration and exploitation of the solution space. ALNS performs reasonably well but generally lags behind NSGA-II, with moderate performance across all metrics, indicating that while it is a viable option, there might be room for improvement.

The figure 4 demonstrates NSGA-II's superior performance in maintaining solution diversity and Pareto front spread across VRPFlexTW problem sizes (100–1000 clients). The top subplot shows NSGA-II's consistently low Spacing Metric (30,000–35,000), indicating uniform solution distribution, while MOPSO, ACO, and ALNS exhibit higher SM values (up to 190,000), reflecting uneven clustering. The bottom subplot highlights NSGA-II's high Diversity Metric (21,973.4 to 20,800), maintaining a broad Pareto front, compared to the sharper declines in MOPSO, ACO, and ALNS (down to 18,000, 16,000, and 18,500, respectively). These results confirm NSGA-II's robustness for large-scale VRPFlexTW instances.

VI. Conclusion

The primary objective of this study was to conduct a comparative analysis of Multi-objective Particle Swarm Optimization (MOPSO), Non-dominated Sorting Genetic

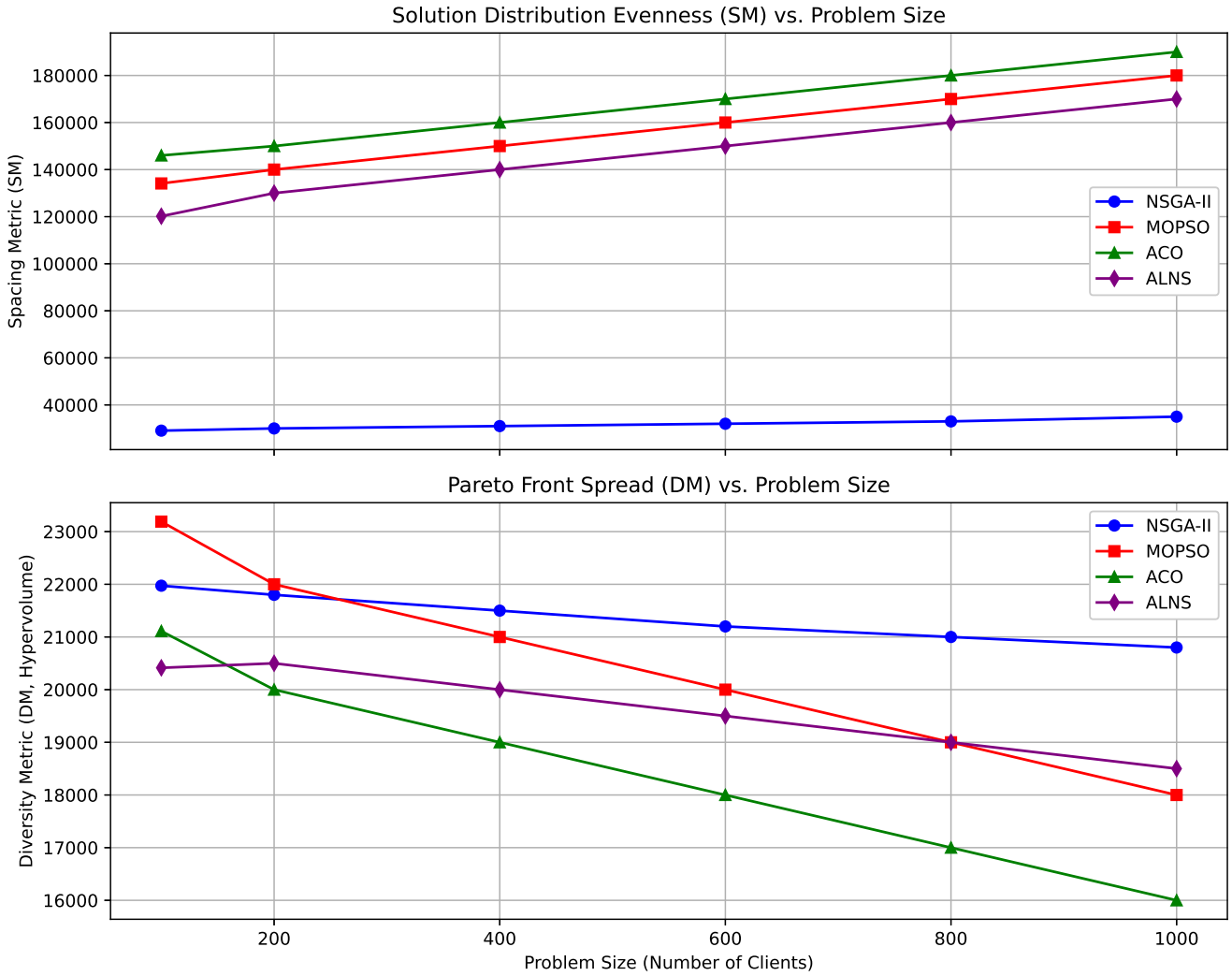

Fig. 4. SM and DM for NSGA-II, MOPSO, ACO, and ALNS across VRPFlexTW problem sizes (100–1000 clients). Top: SM (lower is better); Bottom: DM (higher is better). NSGA-II excels in solution distribution and Pareto front spread.

Algorithm II (NSGA-II), Adaptive Large Neighborhood Search (ALNS), and Ant Colony Optimization (ACO) for solving the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). This paper reviews the current state of research on VRPFlexTW and describes various modified versions of ALNS. The results indicate that NSGA-II is the most reliable and effective algorithm among those tested, particularly in balancing convergence (low Generational Distance, GD) and diversity (high Diversity Metric, DM). MOPSO and ALNS are competitive but exhibit greater variability, suggesting a need for further parameter tuning to achieve consistent performance. ACO, while effective, may require enhancements to better address the problem's complexities and improve overall performance.

Future research directions include investigating hybrid algorithms that combine NSGA-II with other metaheuristic techniques or machine learning methods to enhance solution quality and computational efficiency. Additionally, evaluating NSGA-II's performance in dynamic or stochastic environments, where customer demands and time windows vary unpredictably, could provide valuable insights into its adaptability and robustness.

References

[1] A. S. Umair, W. Zhang, Z. Han, and S. H. U. Haq, "Impact of logistics management on customer satisfaction: A case of retail stores of islamabad and rawalpindi," *American Journal of Industrial and Business Management*, vol. 9, no. 8, pp. 1723–1752, 2019.

[2] R. Gevaers, E. Van de Voorde, and T. Vanelslander, "Cost modelling and simulation of last-mile characteristics in an innovative b2c supply chain environment with implications on urban areas and cities," *Procedia-Social and Behavioral Sciences*, vol. 125, pp. 398–411, 2014.

[3] C. Prodhon and C. Prins, "A survey of recent research on location-routing problems," *European journal of operational research*, vol. 238, no. 1, pp. 1–17, 2014.

[4] C. Dhaenens, M.-L. Espinouse, and B. Penz, "Problemes combinatoires classiques et techniques de résolution," *Recherche Opérationnelle et Réseaux*, 2002.

[5] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[6] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications*. SIAM, 2014.

[7] B. H. O. Rios, E. C. Xavier, F. K. Miyazawa, P. Amorim, E. Curcio, and M. J. Santos, "Recent dynamic vehicle routing problems: A survey," *Computers & Industrial Engineering*, vol. 160, p. 107604, 2021.

[8] M. Asghari, S. M. J. M. Al-e, *et al.*, "Green vehicle routing problem: A state-of-the-art review," *International Journal of Production Economics*, vol. 231, p. 107899, 2021.

[9] N. A. El-Sherbeny, "Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods," *Journal of King Saud University-Science*, vol. 22, no. 3, pp. 123–131, 2010.

[10] F. Labdiad, M. Nasri, I. Hafidi, and H. Khalfi, "A modified adaptive large neighbourhood search for a vehicle routing problem with flexible time windows," *Mathematical modeling and computing*, no. 8, Num. 4, pp. 716–725, 2021.
[11] M. Sajid, J. Singh, R. A. Haidri, M. Prasad, V. Varadarajan, K. Kotecha, and D. Garg, "A novel algorithm for capacitated vehicle routing problem for smart cities," *Symmetry*, vol. 13, no. 10, p. 1923, 2021.
[12] H. Zhang, H. Ge, J. Yang, and Y. Tong, "Review of vehicle routing problems: Models, classification and solving algorithms," *Archives of Computational Methods in Engineering*, pp. 1–27, 2022.
[13] I. Kucukoglu, R. Dewil, and D. Cattrysse, "The electric vehicle routing problem and its variations: A literature review," *Computers & Industrial Engineering*, vol. 161, p. 107650, 2021.
[14] O. Bräysy and M. Gendreau, "Vehicle routing problem with time windows, part i: Route construction and local search algorithms," *Transportation science*, vol. 39, no. 1, pp. 104–118, 2005.
[15] 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, pp. 2033–2062, 2022.
[16] N. Sluijk, A. M. Florio, J. Kinable, N. Dellaert, and T. Van Woensel, "Two-echelon vehicle routing problems: A literature review," *European Journal of Operational Research*, vol. 304, no. 3, pp. 865–886, 2023.
[17] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.
[18] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations research*, vol. 12, no. 4, pp. 568–581, 1964.
[19] S. K. Sharma, S. Routroy, and U. Yadav, "Vehicle routing problem: recent literature review of its variants," *International Journal of Operational Research*, vol. 33, no. 1, pp. 1–31, 2018.
[20] P. Toth and D. Vigo, *The vehicle routing problem*. SIAM, 2002.
[21] R. Wagemaker, "A gpu based parallel algorithm for the vehicle routing problem with time windows," 2016.
[22] L. G. V. Suarez, *A dynamic programming operator for metaheuristics to solve vehicle routing problems with optional visits*. PhD thesis, INSA de Toulouse, 2016.
[23] P. Schaus and R. Hartert, "Multi-objective large neighborhood search," in *Principles and Practice of Constraint Programming: 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings 19*, pp. 611–627, Springer, 2013.
[24] J. Homberger and H. Gehring, "Two evolutionary metaheuristics for the vehicle routing problem with time windows," *INFOR: Information Systems and Operational Research*, vol. 37, no. 3, pp. 297–318, 1999.