Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

PVNS-RSO: Perturbation-based variable neighborhood search with route sequencing optimization algorithm for pickup and delivery problem

Pham Bui Dinh
Hanoi University of Science and Technology
Tam Nguyen Thi
tamnt@vnu.edu.vn
VNU University of Science
Duc Bui Trong
Hanoi University of Science and Technology
Binh Huynh Thi Thanh
Hanoi University of Science and Technology

Research Article

Keywords: pickup and delivery problem, variable neighborhood search, route sequencing optimization

Posted Date: March 5th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8220070/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

PVNS-RSO: Perturbation-based variable neighborhood search with route sequencing optimization algorithm for the Pickup and delivery problem

Bui Dinh Pham[1], Nguyen Thi Tam[2*], Bui Trong Duc[1], Huynh Thi Thanh Binh[1]

[1]School of Information and Communications Technology, Hanoi University of Science and Technology, Hanoi, Vietnam.
[2]VNU University of Science, Hanoi, Vietnam.

*Corresponding author(s). E-mail(s): tamnt@vnu.edu.vn;
Contributing authors: pham.bd220039@sis.hust.edu.vn;
ducbt@soict.hust.edu.vn; binhht@soict.hust.edu.vn;

Abstract

This paper addresses the Energy-aware Pickup and Delivery Problem with Time Windows (E-PDPTW), which aims to minimize both total engine energy consumption and service lateness in logistics operations. Unlike traditional models, we integrate a physics-based energy consumption model that accounts for vehicle load, speed, and driving resistance, providing a more realistic assessment of fuel use. To solve this complex problem, we propose a Perturbation-based variable neighborhood search with the route sequencing optimization algorithm (PVNS-RSO). The algorithm features a tree-based solution representation that inherently preserves pickup-delivery precedence and vehicle capacity constraints. It combines multiple neighborhood structures for effective local search, enhanced by an RSO-based route refinement module for intensive local optimization. A strategic perturbation mechanism is incorporated to escape local optima and maintain search diversity. Comprehensive experiments on benchmark instances demonstrate that our proposed PVNS-RSO algorithm significantly outperforms state-of-the-art methods in both solution quality and computational efficiency, particularly excelling in large-scale and complex scenarios. The results validate the effectiveness of our approach in achieving sustainable and efficient routing solutions for modern logistics.

Keywords: pickup and delivery problem, variable neighborhood search, route sequencing optimization

1 Introduction

Vehicle Routing Problem (VRP) plays a crucial role in modern logistics and supply chain management, significantly influencing business efficiency, customer satisfaction, and urban sustainability [1]. VRP has evolved into a central topic in combinatorial optimization with broad applications in transportation, manufacturing, and e-commerce. However, VRP remains NP-hard, and its practical extensions continue to present complex challenges in real-world logistics operations [2, 3]. In recent years, the pickup and delivery problem with time windows (PDPTW) has emerged as a prominent VRP variant with a strong connection with services such as ride-sharing, courier delivery, and cold-chain transportation [4, 5]. The PDPTW requires designing vehicle routes to fulfill paired pickup and delivery requests within capacity and time-window constraints. Unlike conventional routing problems that focus solely on delivery, the PDPTW coordinates both pickup and delivery operations, thereby reducing empty mileage, optimizing resource utilization, and enhancing overall logistics efficiency.

Additionally, the rapid expansion of e-commerce and just-in-time distribution has drawn increasing attention to sustainability and energy-aware logistics planning [6]. Green routing aims not only to minimize travel distance or service time but also to reduce fuel consumption and carbon emissions in alignment with global decarbonization goals. Recent studies emphasize that realistic energy models, which account for vehicle mass, speed, and driving resistance, provide a more accurate basis for route optimization than distance-based surrogates [7]. However, incorporating energy consumption into PDPTW formulations introduces additional nonlinearities and trade-offs between efficiency, service punctuality, and environmental impact.

To address these issues, this paper proposes an energy-aware PDPTW formulation that minimizes total engine energy consumption and lateness penalties simultaneously. The model integrates an energy consumption model on various components, including aerodynamic drag, rolling resistance, and load weight. This approach captures the primary energy-related factors in logistics operations and provides a balanced framework between sustainability and service quality. Furthermore, we develop a perturbation-based variable neighborhood search with route sequencing optimization (PVNS-RSO) algorithm. The PVNS-RSO employs a tree-based route representation that inherently preserves pickup–delivery precedence and vehicle capacity constraints. Local operators such as subtree swap, node relocation, and subtree rotation promote diverse yet feasible neighborhood exploration. In addition, the route sequencing optimization (RSO) component refines individual routes, while adaptive perturbation mechanisms help prevent premature convergence and improve search robustness. Therefore, the main contributions of this study are summarized as follows:

- We develop an energy-aware PDPTW formulation that integrates a physics-based engine energy model with lateness penalties, enabling more realistic and sustainable route planning.
- We propose a perturbation-based variable neighborhood search (VNS) framework that employs a tree-based representation, which inherently preserves feasibility while supporting diverse neighborhood operations. In addition, a route-sequencing optimization module is incorporated to refine intra-route visit orders, improving local search efficiency and accelerating overall convergence.
- Numerical experiments on multiple benchmark sets demonstrate that the proposed approach delivers superior performance and strong scalability compared to state-of-the-art algorithms.

The remainder of this paper is organized as follows. Section 2 Section 3 provides an overview about related works. The problem formulation and energy model are presented in Section 3. Section 4 describes the proposed algorithm. Section 5 discusses results, and Section 6 concludes the paper.

2 Related works

The pickup and delivery problem (PDP) is a classical NP-hard routing problem in which a fleet of vehicles fulfills paired pickup-and-delivery requests [4, 5]. Each request includes a pickup and a corresponding delivery node, and routes must satisfy precedence and capacity constraints. When service time windows are imposed, the problem becomes the PDPTW, which aims to minimize routing cost while meeting service deadlines [3, 8, 9]. Zhang *et al.* [8] addresses a multi-type vehicle routing problem with integrated pickup and delivery under time windows. Yang *et al.* [9] introduces a time-dependent PDPTW for electric vehicles in cold supply chains, while Wang *et al.* [3] formulates a bi-objective model minimizing total operating cost and fleet size. Recent extensions of the PDP have incorporated environmental considerations, giving rise to the Green Vehicle Routing Problem (GVRP) and the Pollution Routing Problem (PRP), which explicitly account for fuel consumption and CO_2 emissions [10, 11, 12]. Sadati *et al.* [10] optimize multi-depot GVRPs using alternative-fuel vehicles, while Qi *et al.* [11] introduce a multi-objective model that minimizes travel duration, energy use, and customer dissatisfaction, considering slope, speed, and vehicle load. Amiri *et al.* [13] address total GHG emissions from mixed fleets of conventional and electric trucks. Wang *et al.* [14] further consider dynamic traffic and variable speed effects on fuel and emission costs. These studies highlight the growing attention to both operational efficiency and environmental sustainability in routing optimization.

2.1 Exact approaches

A wide range of algorithmic strategies has been developed to address the PDP and its variants. Exact methods, including branch-and-price and branch-and-cut, have proven effective for small- to medium-sized instances, particularly when enhanced with decomposition or metaheuristic components. Gabriel *et al.* [15] develops a branch-and-price algorithm for a capacitated fleet performing deliveries and profitable pickups under

time-window constraints, aiming to minimize total routing cost minus pickup revenues. Karaoglan *et al.* [16] proposes a branch-and-cut algorithm for depot location and vehicle routing, integrating valid inequalities and a simulated annealing–based local search to improve upper bounds. Additionally, Thomas *et al.* [17] formulate an integer linear programming (ILP) model for the Location and Routing Problem with Pickup and Delivery and introduce a column-generation framework using a label-setting algorithm to solve the subproblem efficiently. More recently, several studies have employed mixed-integer linear programming (MILP) models to capture various PDP extensions [18, 19, 20], achieving high solution accuracy and benchmark competitiveness. However, these exact approaches suffer from poor scalability, limiting their applicability to large-scale or real-time scenarios.

2.2 Approximate approaches

Regarding metaheuristics, algorithms such as evolutionary computation, swarm intelligence, and local search–based methods have become promising for solving large-scale PDPs. These approaches effectively balance exploitation and exploration, enabling them to handle precedence, capacity, and time-window constraints while maintaining computational efficiency. Cui *et al.* [21] propose an adaptive genetic algorithm that addresses the timeliness and high cost of logistics distribution. Wang *et al.* [22] introduce a hybrid framework that applies fuzzy C-means clustering to decompose the problem into smaller subproblems, which are then solved by an enhanced multipopulation genetic algorithm integrated with VNS. Ren *et al.* [23] design an improved ant colony optimization (ACO) algorithm that initializes pheromones using vehicle information and applies heuristic mechanisms to ensure high-quality initial solutions. Local search–based approaches also demonstrate strong performance, including adaptive large neighborhood search (ALNS) [24, 25], tabu search [26], and simulated annealing [27], which iteratively refine solutions through trajectory-based improvements. Several hybrid frameworks have been developed to combine the strengths of multiple metaheuristics. Oztac *et al.* [28] propose a hybrid algorithm integrating iterated local search, variable neighborhood descent, and threshold acceptance, where perturbation and intensification strategies enhance exploration efficiency. Gao *et al.* [29] employ logic-based Benders decomposition and dynamic programming to accelerate subproblem resolution.

Zhou et al. in [30] formulated a practical Dynamic Pickup and Delivery Problem (DPDP) model that incorporates real-world constraints such as limited docks, time windows, vehicle capacity, and last-in-first-out loading. To solve it, they proposed a Variable Neighborhood Search with Multiple Local Searches and Efficient disturbance algorithm, which dynamically reconstructs solutions as new orders arrive. The proposed method demonstrated strong performance, achieving the first place among 153 teams in the ICAPS 2021 DPDP competition. In parallel, other researchers have extended the pickup and delivery framework to green and sustainable logistics systems. For instance, [31] introduced the Cold Chain Green Multi-Depot Vehicle Routing Problem with Time Windows and Mixed Fleets , which integrates both electric and conventional vehicles under a realistic energy consumption model. The

proposed improved Variable Neighborhood Search algorithm minimizes total operational costs—including fixed, energy, damage, and environmental costs—and evaluates the impact of carbon pricing and time-window adjustments. On the other hand, the authors in [32] actually solved a relaxed version of the ICAPS-DPDP rather than the original problem. Specifically, they ignored the limited docking-port constraint at factories and consequently modified the benchmark dataset without explicitly mentioning it. Experimental evidence showed that their published results corresponded to a case where the number of docks was unlimited, thus making their comparison and claims of state-of-the-art performance misleading for the true ICAPS-DPDP.

Recently, learning-based methods have been incorporated into PDP optimization to improve adaptivity and generalization. Reinforcement learning (RL) frameworks [33, 34, 35] learn routing policies directly from experience, providing strong performance on unseen instances. Parameter-adaptive metaheuristics have also emerged, leveraging machine learning for automatic tuning. Pugliese *et al.* [36] combine machine learning techniques with VNS to explore promising regions of the solution space. Zhang *et al.* [37] propose a two-stage learning-based method that employs a graph convolutional network with attention for clustering, followed by a trained model to generate vehicle routes.

2.3 Research gaps and Motivations

Despite significant progress in solving PDP and its variants, several research gaps remain that hinder the development of more sustainable and adaptable routing systems. Exact algorithms, such as branch-and-price and branch-and-cut, provide optimality guarantees but are computationally prohibitive for large-scale or real-time applications [15, 16]. Their exponential complexity limits scalability when dealing with high-dimensional problems, particularly in applications involving a large number of vehicles, customers, or complex constraints. Metaheuristic and hybrid approaches have demonstrated strong global search capabilities and flexibility in handling complex constraints [24, 28, 22]. However, most existing studies rely on random initialization, which slows convergence and increases computational cost, and on restricted solution representations, which limit evolutionary diversity and hinder both the exploration and exploitation of the solution space. Although RL methods [33, 34] have recently shown promise in learning adaptive routing policies, they typically require large amounts of training data and extensive computational resources. Moreover, the generalization ability of RL-based models across heterogeneous PDP scenarios remains limited, especially when problem parameters or constraints change dynamically, as in time-dependent travel conditions or varying customer demands. Additionally, while many recent works have incorporated energy and environmental considerations into vehicle routing problems, they often assume fixed or simplified models of fuel consumption and rarely address the dynamic nature of vehicle load variation in pickup-and-delivery operations.

To bridge these gaps, this paper adopts a physics-based engine energy model in which the energy required to traverse each arc depends dynamically on the vehicle's instantaneous payload. This more realistic modeling approach enables accurate

assessment of fuel consumption as a function of vehicle weight and speed, providing deeper insight into the trade-offs between logistical efficiency and environmental impact. Extending the PDPTW problem to include such an energy objective results in the E-PDPTW, where energy consumption is directly affected by the pickup and delivery sequence. In this paper, we propose a PVNS-RSO algorithm that integrates VNS, route sequence optimization, and controlled perturbation mechanisms to effectively address the energy-efficient pickup and delivery routing problem. The proposed algorithm begins with a hybrid population initialization strategy to generate diverse and high-quality starting solutions. Then, a VNS framework is employed to explore multiple neighborhood structures for solution improvement, while an RSO optimization operator refines the visiting order of service nodes to minimize travel cost. Finally, a perturbation mechanism is applied to enhance population diversity and prevent premature convergence.

3 Problem Formulation

This section formally defines the Energy-aware Pickup and Delivery Problem with Time Windows (E-PDPTW) and presents the corresponding objective functions and constraints.

3.1 System Description

The problem is defined on a complete directed graph $G = (V, A)$, where V is the set of vertices and A is the set of directed arcs. Let R denote the set of n transportation requests. Each request $r \in R$ requires the transportation of a load q_r from the pickup location r^+ to its delivery location r^-.

Each pickup and delivery operation incurs a handling time t_{r^+} and t_{r^-}, respectively, and must satisfy corresponding time-window constraints:

$$[e_{r^+}, l_{r^+}] \quad \text{and} \quad [e_{r^-}, l_{r^-}],$$

which represent the earliest and latest allowable start times for service.

The vertex set is defined as:

$$V = \{0\} \cup \{r^+\} \cup \{r^-\},$$

where vertex 0 represents the depot, and all other vertices correspond to pickup and delivery points. The arc set A includes all possible directed connections between distinct vertices, with travel time t_{ij} associated with each arc $(i, j) \in A$. The fleet is represented by a finite set of vehicles $K = \{1, 2, \ldots, |K|\}$. Each vehicle $k \in K$ starts and ends at the central depot. We denote that Q is the capacity of each vehicle.

3.2 Energy Consumption Model

This work focuses on engine energy consumption, while refrigeration-related energy use is excluded for model tractability. Although refrigeration can contribute up to 40%

of total energy consumption in cold-chain logistics [7], including it would introduce numerous uncertain parameters, such as unloading duration, door-opening frequency, ambient temperature. Thus, the proposed model isolates the engine-related cost, and refrigeration effects are left for future work.

The total engine energy consumption of vehicle k along its route is defined as:

$$E_k = \sum_{(i,j)\in \text{path}_k} L_{ij}^k, \tag{1}$$

where $path_k$ is the route of vehicle k and L_{ij}^k is the energy consumed by vehicle k when traveling from node i to node j, which is calculated as follows:

$$L_{ij}^k = G_{ij}^k \cdot t_{ij}. \tag{2}$$

The instantaneous fuel (or energy) consumption rate G_{ij}^k is computed as:

$$G_{ij}^k = \frac{\xi}{\kappa \Psi} \left(\varpi R + \frac{T_{ij}^k}{\eta} \right), \tag{3}$$

where:

ξ fuel-to-air mass ratio,
κ heating value of the fuel (energy per unit mass),
Ψ engine friction coefficient,
ϖ constant representing idling or baseline consumption,
R baseline engine load,
η driveline efficiency ($0 < \eta \leq 1$).

The mechanical power output T_{ij}^k required to move vehicle k from i to j is given by:

$$T_{ij}^k = \frac{1}{2} c_d \rho A v^3 + (m_k + u_{ij}^k) g c_r v, \tag{4}$$

where:

c_d aerodynamic drag coefficient,
ρ air density,
A vehicle frontal area,
g gravitational constant,
c_r rolling resistance coefficient,
m_k curb weight of vehicle k,
u_{ij}^k total cargo weight carried on arc (i, j).

3.3 Objective Function

The total cost of the system is formulated as a weighted sum of energy consumption and lateness penalty:

$$\min f = p_1 \sum_{k \in K} E_k + p_2 \sum_{k \in K} P_k, \tag{5}$$

where p_1 and p_2 are weights for energy cost and lateness penalty, respectively. The objective is to minimize total operational energy while reducing service lateness across all vehicles.

For vehicle k, the total lateness penalty is:

$$P_k = \sum_{i \in \text{path}_k} \text{lateness}_i.$$

A lateness penalty is incurred if a vehicle arrives at a node after its latest service time. Therefore, the lateness at node i is defined as:

$$\text{lateness}_i = \max(0, \text{visit}_i - l_i),$$

where visit_i is the actual arrival time and l_i is the latest allowable time.

3.4 Constraints

To monitor the vehicle scheduling process, the following decision variables are defined:

x_{ij}^k Equals 1 if vehicle k travels along arc (i, j) to serve location j, and 0 otherwise;
$\tilde{q}_i^k$ Denotes the load carried by vehicle k upon visiting node i;
t_i Represents the service start time at location i;
M A sufficiently large constant.

Specifically, E-PDPTW is subject to the following constraints:

Service assignment constraint

Each request must be served exactly once by a single vehicle:

$$\sum_{k \in K} \sum_{j \in V} x_{r^+ j}^k = 1, \sum_{k \in K} \sum_{i \in V} x_{i r^-}^k = 1, \quad \forall r \in R, \tag{6}$$

$$\sum_{j \in V} x_{r^+ j}^k = \sum_{j \in V} x_{r^- j}^k, \quad \forall r \in R, \ \forall k \in K. \tag{7}$$

Flow continuity constraint

For each vehicle, the number of arcs entering a vertex must equal the number of arcs leaving it:

$$\sum_{i \in V} x_{ij}^k = \sum_{h \in V} x_{jh}^k, \quad \forall k \in K, \ \forall j \in V \setminus \{0\}. \tag{8}$$

This ensures that every vehicle maintains a continuous and feasible route.

Vehicle capacity constraint

At any time, the total load carried by a vehicle cannot exceed its capacity Q:

$$\tilde{q}_j^k \leq Q, \quad \forall k \in K,\ \forall j \in V, \tag{9}$$

$$\tilde{q}_i^k + d_j - M(1 - x_{ij}^k) \leq \tilde{q}_j^k, \quad \forall (i,j) \in A,\ \forall k \in K, \tag{10}$$

where d_j denotes the load quantity, being positive at pickup nodes and negative at delivery nodes.

Pickup-before-delivery constraint

Each delivery node must be visited after its corresponding pickup node:

$$t_{r^-} \geq t_{r^+} + s_{r^+} + t_{r^+r^-}, \quad \forall r \in R, \tag{11}$$

where s_{r^+} is the service time at the pickup location.

LIFO constraint

The loading and unloading sequence must follow the LIFO rule: the last item picked up must be delivered first. For any two requests r_1, r_2 served by the same vehicle:

$$t_{r_1^+} < t_{r_2^+}, \quad t_{r_1^-} > t_{r_2^-}, \quad \forall r_1, r_2 \in R_k. \tag{12}$$

Depot constraints

Each vehicle must start and end its route at the depot:

$$\sum_{j \in V \setminus \{0\}} x_{0j}^k = 1, \quad \forall k \in K, \tag{13}$$

$$\sum_{i \in V \setminus \{0\}} x_{i0}^k = 1, \quad \forall k \in K. \tag{14}$$

Time window constraints

Each node must be visited within its specified time window:

$$e_i \leq t_i \leq l_i, \quad \forall i \in V. \tag{15}$$

If a vehicle arrives early, waiting is allowed:

$$t_i \geq t_j + s_j + t_{ji} - M(1 - x_{ji}^k), \quad \forall (j,i) \in A,\ \forall k \in K, \tag{16}$$

Binary and non-negativity constraints

$$x_{ij}^k \in \{0,1\}, \quad t_i \geq 0, \quad \tilde{q}_i^k \geq 0. \tag{17}$$

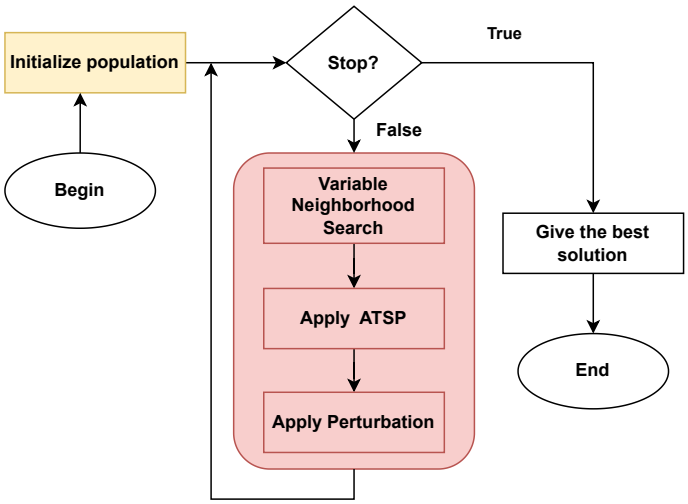


Fig. 1: Framework of the PVNS-RSO algorithm.

4 Proposed algorithm

4.1 Framework of Perturbation-based Variable Neighborhood Search with RSO optimization algorithm

The overall PVNS-RSO algorithm integrates VNS, route sequence optimization, and perturbation mechanisms to enhance both exploration and exploitation. The process begins with the population initialization phase (see Section 4.2), where an initial set of feasible solutions (individuals) is generated. Each individual is represented by a tree-based structure encoding the assignment of pickup and delivery requests to vehicles. Next, for each individual in the population, the algorithm applies the VNS. The VNS iteratively explores multiple neighborhood structures using the defined local moves to improve solution quality while maintaining feasibility. Afterward, the RSO operator is applied to each route within an individual to optimize the visiting order of service nodes. This step refines the route structure and reduces overall travel cost by solving a local sequencing subproblem for each vehicle. Finally, a perturbation operator is employed to diversify the population and prevent premature convergence. It introduces controlled randomness into selected individuals, allowing the algorithm to escape local optima and explore new regions of the search space. Figure 1 illustrates the framework of PVNS-RSO algorithm.

4.2 Individual representation

The pickup and delivery problem (DPDP) can be effectively represented by a tree-based structure, as shown in Figure 2. In this representation, the root node denotes the depot, serving as the origin and destination of all vehicles. The first-level child nodes correspond to the set of available vehicles (e.g., v_1, v_2, v_3). Each vehicle node

is associated with a subtree that specifies the sequence of pickup and delivery tasks assigned to that vehicle. Within each subtree, both internal nodes and leaf nodes may represent service locations, where a label '+' indicates a pickup node and a label '–' indicates a delivery node.

This hierarchical representation allows the entire solution to be decomposed into independent route substructures, one for each vehicle, while still preserving global precedence and capacity constraints. By applying an in-order traversal to each vehicle subtree, the exact service sequence can be derived. Importantly, this traversal mechanism naturally enforces the precedence constraint, i.e., every pickup location is always visited before its corresponding delivery location. For example, vehicle v_1 is assigned orders 1, 13, 8, 10, 12, 6, and traversing its subtree in in-order sequence produces the route: 1+, 13+, 13–, 8+, 8–, 1–, 10+, 10–, 12+, 6+, 6–, 12–. Similarly, vehicle v_2 serves orders 7, 11, 4, 9, 3, while vehicle v_3 serves orders 14, 5, 2.

This tree-based modeling approach offers several advantages. First, it ensures solution feasibility by construction, as the pickup–delivery precedence constraint is inherently satisfied. Second, it enables structured manipulation of solutions, since subtrees can be directly modified, recombined, or exchanged between vehicles in heuristic or evolutionary optimization algorithms. Finally, it provides a clear visualization of complex routes, making it easier to analyze, evaluate, and optimize routing strategies in large-scale DPDP instances.

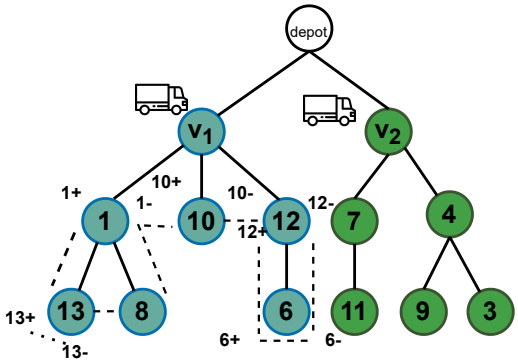


Fig. 2: Individual representation.

4.3 Heuristic Initialization

To provide a strong starting point for the PVNS-RSO algorithm, we adopt a hybrid greedy initialization strategy that combines deterministic and randomized constructions. The deterministic solution establishes a high-quality baseline following a time-based service policy, while the randomized individuals enhance diversity by exploring different regions of the solution space. All solutions are generated using the

same greedy best-insertion heuristic, ensuring computational efficiency. This hybrid design balances quality, diversity, and robustness, allowing the algorithm to start from a well-distributed set of promising solutions that effectively guide subsequent VNS, RSO optimization and perturbation phases. The detailed steps of the heuristic initialization are presented in Algorithm 1.

Algorithm 1: Population Initialization

Input: Set of requests R, set of vehicles K, graph G, population size N_{pop}
Output: Initial population $\mathcal{P} = \{S_1, S_2, \ldots, S_{N_{\text{pop}}}\}$

Initialize empty population $\mathcal{P} \leftarrow \emptyset$;
`// Generate deterministic individual`
$R_{\text{ordered}} \leftarrow \text{Sort_By_Earliest_Pickup_Time}(R)$;
$S_1 \leftarrow \text{Heuristic_Initialization}(R_{\text{ordered}}, K, G)$;
$\mathcal{P} \leftarrow \mathcal{P} \cup \{S_1\}$;
`// Generate randomized individuals`
for $i = 2$ **to** N_{pop} **do**
    $R_{\text{shuffled}} \leftarrow \text{Random_Shuffle}(R)$;
    $S_i \leftarrow \text{Heuristic_Initialization}(R_{\text{shuffled}}, K, G)$;
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{S_i\}$;
return $\mathcal{P}$;

Step 1: Deterministic individual generation. The first individual is constructed deterministically to provide a strong quality baseline. All transportation requests are first sorted in ascending order of their earliest pickup times:

$$R_{\text{ordered}} = \text{Sort}(R, \text{by } e_{r^+}).$$

This ordering reflects a time-prioritized service policy that minimizes waiting and lateness at the early stage of route construction. Then, the best insertion heuristic is applied to sequentially insert each request into the available vehicle routes, resulting in the deterministic solution S_1. This solution ensures temporal consistency and typically yields a feasible, near-greedy baseline for the population.

To ensure a high-quality baseline solution in the initial population, we incorporate a deterministic initialization strategy that processes requests in a specific order based on their temporal characteristics. This approach provides a stable, reproducible starting point that serves as a reference for evaluating the diversity introduced by randomized initializations.

The deterministic strategy sorts all requests in ascending order of their earliest pickup time (or release time) before applying the best insertion heuristic. Formally, requests are ordered such that:

$$e_{r_1^+} \leq e_{r_2^+} \leq \cdots \leq e_{r_n^+}, \tag{18}$$

where $e_{r_i^+}$ denotes the earliest allowable pickup time for request r_i.

By using this deterministic ordering for one individual in the population, we establish a consistent benchmark that represents a reasonable, time-conscious routing

strategy. This deterministic individual often produces competitive results and provides a stable foundation for the population-based search.

Step 2: Randomized individual generation. To introduce solution diversity, the remaining $N_{\text{pop}} - 1$ individuals are generated using randomized permutations of the request sequence:

$$R^{(i)}_{\text{shuffled}} = \text{Random_Shuffle}(R), \quad i = 2, 3, \ldots, N_{\text{pop}}.$$

For each randomized sequence, the same best insertion heuristic is applied to construct a feasible solution S_i. By randomizing the request order, these individuals explore different regions of the search space, promoting population diversity and preventing premature convergence.

Step 3: Best inserttion heuristic. Apply the best insertion heuristic to each $R^{(i)}_{\text{shuffled}}$ to obtain S_i.

The heuristic initialization employs a best insertion strategy that iteratively processes requests one by one and inserts each pickup-delivery pair into the position that yields the minimum increase in the total objective value. The algorithm begins with empty vehicle routes and sequentially assigns requests by evaluating all possible insertion positions across all vehicles. For each request, the algorithm considers all feasible combinations of pickup and delivery positions, selects the one with minimum cost increase, and updates the solution accordingly. The heuristic initialization is formally detailed in Algorithm 2.

For a problem instance with n requests, m vehicles, and an average route length of ℓ, the computational complexity can be analyzed as follows. For each request, the number of candidate insertion positions per vehicle is on the order of $O(\ell^2)$. Considering all m vehicles, each request therefore requires $O(m\ell^2)$ evaluations. When processing n requests sequentially, the overall complexity becomes $O(nm\ell^2)$. Since the average route length ℓ typically increases with the number of inserted requests, in the worst case where $\ell \approx n$, the computational complexity reaches $O(mn^3)$.

Although the algorithm exhibits a cubic worst-case complexity, it remains computationally efficient in practice. This efficiency arises from three main factors. First, infeasible insertions are often discarded early during feasibility checks, which significantly reduces the number of evaluated candidates. Second, the feasible search space is relatively sparse, as capacity and time-window constraints restrict the number of valid insertion positions. Finally, route lengths tend to grow gradually and approximately linearly with the number of inserted requests, rather than quadratically as in the theoretical worst case.

4.4 Variable Neighborhood Search

The proposed framework incorporates several local moves. These operators differ in the scale of modification they introduce: subtree-based moves support large-scale adjustments, often redistributing requests among vehicles, while node-based moves enable fine-grained changes within or across routes. By combining both types, the search process achieves a balance between diversification and intensification, which is crucial for escaping local optima and discovering high-quality solutions.

Algorithm 2: Best Insertion Heuristic

Input: Set of requests R, vehicles K, and graph G
Output: Initial tree-based solution S

```
// Initialization
```
Initialize each vehicle $k \in K$ with route $\text{route}_k \leftarrow [0, 0]$; `// Depot at both ends`
$R' \leftarrow \text{Random_Shuffle}(R)$;
Request insertion:
foreach $r \in R'$ **do**
    best $\leftarrow (\infty, \text{null}, -1, -1)$; `// (cost, vehicle, pickup, delivery)`
    foreach $k \in K$ **do**
        for $i = 1$ **to** $|route_k| - 1$ **do**
            for $j = i + 1$ **to** $|route_k|$ **do**
                new_route $\leftarrow$ Insert(r^+, i) and Insert(r^-, j) into route_k;
                if *Is_Feasible*(*new_route*, G) **then**
                    $\Delta C \leftarrow \text{Cost}(\text{new_route}, G) - \text{Cost}(\text{route}_k, G)$;
                    if $\Delta C <$ *best.cost* **then**
                        best $\leftarrow (\Delta C, k, i, j)$;
    if *best.vehicle* $\neq$ *null* **then**
        Insert r^+ and r^- into $\text{route}_{\text{best.vehicle}}$ at positions best.pickup and best.delivery;
Finalize:
$S \leftarrow \text{Routes_To_Tree}(\{\text{route}_k \mid k \in K\})$;
return S;

4.4.1 Subtree swap

In the local search framework based on the tree representation, a typical neighborhood operator is the subteee swap. In this operation, two subtrees are selected from different branches of the solution tree, where each branch corresponds to the route of a particular vehicle. Each subtree represents a contiguous set of service tasks (including both pickup and delivery nodes) assigned to the corresponding vehicle. The subtree swap operator exchanges the positions of these two subtrees between the routes of the selected vehicles.

For example in Figure 3, consider a subtree in the route of vehicle v_1 representing a pickup–delivery pair and another subtree in the route of vehicle v_2 corresponding to a different order. The subtree swap exchanges these subtrees, such that the order previously served by v_1 is reassigned to v_2, and vice versa. Importantly, the internal structure of each subtree remains unchanged during the swap, thereby preserving the precedence relation between pickup and delivery nodes. The subtree swap operator offers two main benefits. First, it enables the exploration of new solution spaces

by reallocating orders between vehicles while maintaining feasibility with respect to precedence constraints. Second, it contributes to search diversification, reducing the likelihood of being trapped in local optima when combined with other neighborhood moves.

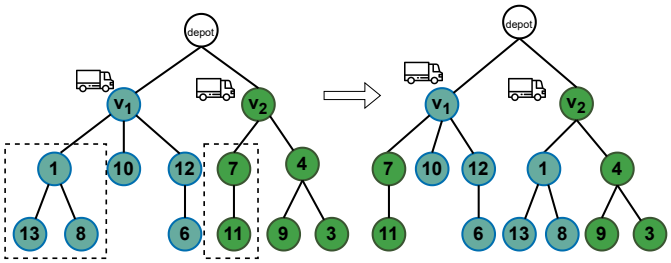


Fig. 3: Subtree swap.

4.4.2 Node swap

Another local move is the *node swap*, where two arbitrary nodes in the tree are selected and their positions are exchanged. Each node represents either a pickup and delivery location within a vehicle route. The swap operation alters the relative positions of the two selected service nodes, potentially reordering the sequence of visits in one or two vehicle routes depending on the location of the swapped nodes. Figure 4 shows an example of this operator, a node 8 in vehicle v_1 and a node 9 in vehicle v_2 are chosen, the node swap operator exchanges their positions, reassigning 8 to the route of v_2 and 9 to the route of v_1. Similarly, if both nodes are selected within the same route, the swap only modifies the internal visiting sequence of that vehicle.

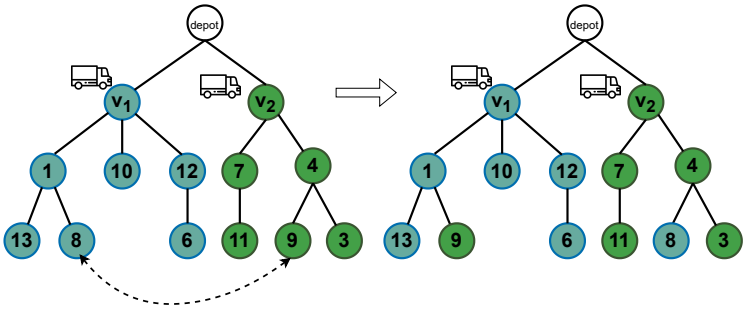


Fig. 4: Node swap.

4.4.3 Subtree relocate

This operator selects a subtree corresponding to one or more pickup-delivery requests from the route of a given vehicle and reassigns it to the route of another vehicle. The subtree is detached from its original vehicle route and then inserted into the route of a different vehicle, typically at a feasible position that maintains the pickup–delivery precedence and vehicle capacity constraints.

An example is shown in Figure 5, consider a subtree with root 4 in the route of vehicle v_2. Applying the subtree relocate move transfers this subtree to vehicle v_1, effectively reassigning the corresponding customer request from v_2 to v_1. After the relocation, the internal structure of the subtree remains unchanged, while the overall workload distribution among vehicles is altered. This operator is particularly useful for balancing load across vehicles, as well as for reducing route costs in cases where transferring a group of requests to another vehicle results in shorter travel distances or better time window alignment.

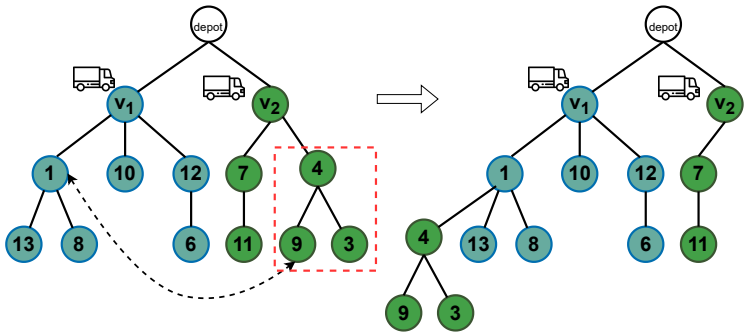


Fig. 5: Subtree relocate.

4.4.4 Node relocate

This operator modifies the solution tree by detaching a single node from its current position and reinserting it into a different part of the tree. The procedure is as follows: first, a node x is randomly selected. The children of x are then directly reattached to the parent of y, thereby preserving the connectivity of the tree after x is removed. Next, another node y is randomly chosen, and the detached node x is reinserted as a child of y. An example of this operator is in Figure 7, with x being node 4, and y being node 10.

4.4.5 Subtree rotate

This operator enables internal restructuring of a subtree by promoting a selected node to become the new root while preserving the relative relationships among other nodes through systematic tree rotations. The operation proceeds as follows: first, a subtree

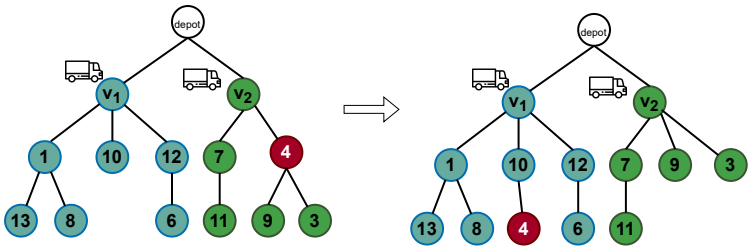


Fig. 6: Node relocate.

S with root r is selected from the solution tree. Then, a node x within S is chosen to become the new root. The transformation involves repeatedly applying tree rotations along the path from x to r until x reaches the root position.

During this process, for every node u that is either x itself or an ancestor of x in the original subtree, the branch from u to its parent becomes the rightmost branch in the restructured subtree. This systematic rightward shifting ensures that all precedence constraints within the subtree are maintained while achieving the desired reorganization.

For example, as illustrated in Figure 7, a subtree rooted at node 1 is selected. Within this subtree, node 8 is designated to become the new root. The rotation operation is then successively applied along the path from node 8 to the original root 1, resulting in node 8 being promoted to the root position. Throughout this process, the relative parent–child relationships among other nodes, such as nodes 13 and 1, are preserved.

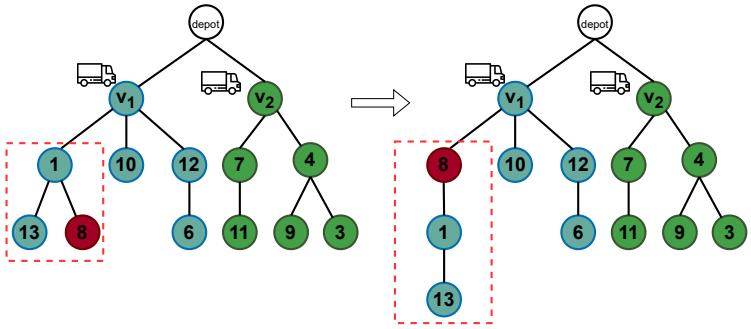


Fig. 7: Subtree rotate.

4.5 RSO Optimization

The objective of this stage is to optimize the visiting sequence of child nodes within each local route tree in order to minimize the total travel cost. Two distinct strategies are employed depending on the size of the branch. For small branches, a brute-force

search is applied. All $n!$ possible permutations of the child nodes are generated, and the total cost of each permutation is evaluated. The permutation yielding the minimum cost is selected. This approach ensures an optimal visiting sequence for small-scale subtrees. For branches containing more than seven child nodes, the computational cost of brute-force search is prohibitively high. Therefore, the Regret-2 heuristic is employed. This heuristic iteratively inserts the next node in a way that minimizes the additional cost. It provides an efficient approximation with near-optimal performance.

4.6 Perturbation Operator

The perturbation process proceeds as follows. First, the subtree with the highest accumulated route cost is identified from the solution tree. This subtree typically corresponds to a vehicle route that contributes the most to the total objective value. Next, all nodes within this subtree are removed from the solution. Finally, the removed nodes are reinserted sequentially into the tree, each at its best feasible position determined according to the cost-minimization criterion while maintaining precedence and capacity feasibility. This perturbation mechanism introduces significant structural modifications while preserving the feasibility of the overall solution. It allows the algorithm to escape local minima by reconstructing suboptimal routes in a more efficient form, thereby improving global search capability and solution diversity.

5 Experiments

5.1 Experimental Setting

This paper evaluates the proposed algorithm using the PDPTW benchmark instances [38], which are widely recognized as the standard for assessing PDPTW algorithms due to their diversity and complexity. Each instance name encodes key characteristics in the format `[Type][Size]-[Category]-[Index]`, where `Type` indicates the geographic distribution of customers, such as `C`: clustered, `R`: random, `RC`: mixed; `Size` denotes the number of requests, including 100, 200 and 400 requests, `Category` specifies the time-window width, such as `1`: narrow, `2`: wide, and `Index` identifies the instance. For example, `C200-4-8` corresponds to a clustered distribution with 200 requests, wide time windows, and is the eighth instance. The number of trucks and the capacity of each truck are predefined for each dataset. Therefore, the performance of the proposed algorithm is evaluated across a total of 180 instances.

All algorithms are implemented in Python 3.9 and run on a machine running Ubuntu 22.04 LTS, equipped with an Intel Core i7-12700K CPU @ 3.60GHz and 32GB of DDR4 RAM. Each algorithm is evaluated over 10 independent runs per instance with different random seeds to ensure statistical validity. Each run is limited to 50,000 solution evaluations, and average results across runs are reported for analysis.

5.2 Experimental Results

5.2.1 Impact of initialization on algorithm performance

To evaluate the effectiveness of our proposed heuristic initialization strategy, we conduct a comprehensive ablation study comparing the strategy against random initialization. Table 1 presents the comparative results across instances of varying sizes and distributions. The objective value combines total energy consumption and lateness penalty as defined in Equation 5.

Table 1: Ablation study comparing heuristic and random initialization in terms of average objective value and average computation time

Instance	Random Initialization		Heuristic Initialization	
	Average objective	Average time	Average objective	Average time
100-C	2.52E+05 ± 2.86E+05	23.77 ± 0.62	**2.04E+04 ± 2.81E+03**	26.55 ± 3.66
100-R	1.67E+05 ± 2.87E+05	23.70 ± 0.80	**3.22E+04 ± 3.14E+04**	25.71 ± 5.73
100-RC	2.41E+05 ± 3.06E+05	23.69 ± 0.78	**3.06E+04 ± 5.06E+03**	25.67 ± 4.83
200-C	3.38E+06 ± 3.08E+06	42.93 ± 1.00	**7.01E+04 ± 1.31E+04**	60.83 ± 10.96
200-R	1.85E+06 ± 1.80E+06	42.62 ± 1.14	**9.06E+04 ± 1.79E+04**	69.94 ± 27.02
200-RC	1.53E+06 ± 1.21E+06	42.55 ± 1.86	**7.37E+04 ± 1.04E+04**	67.17 ± 17.77
400-C	4.30E+07 ± 2.24E+07	84.12 ± 10.45	**1.95E+05 ± 3.85E+04**	162.12 ± 38.61
400-R	2.73E+07 ± 1.64E+07	93.84 ± 15.78	**2.38E+05 ± 7.41E+04**	215.39 ± 105.62
400-RC	2.87E+07 ± 1.37E+07	89.37 ± 13.54	**1.96E+05 ± 2.51E+04**	186.49 ± 70.19

The ablation study results demonstrate the critical importance of our proposed heuristic initialization strategy. The most striking observation is the dramatic improvement in solution quality achieved by heuristic initialization. Across all instance, the heuristic initialization produces solutions that are one to two orders of magnitude better than those obtained through random initialization. The computational trade-off between the two methods reflects their inherent complexity differences. Random initialization operates with a time complexity of $O(n)$, yielding faster execution times ranging from 23.77 to 93.84 seconds across instances. In contrast, heuristic initialization exhibits $O(n^3)$ complexity, requiring substantially higher computational resources. This complexity difference manifests in the observed runtime patterns: for 100-request instances, heuristic initialization requires approximately $8 - 12\%$ additional time; for 200-request instances, the time increase expands to $40 - 64\%$, and for 400-request instances, heuristic initialization takes approximately $1.9 - 2.3$ times longer than random initialization.

Despite the increased computational demand, the exceptional improvement in solution quality fully justifies this investment. The heuristic method's intensive optimization of penalty time objectives effectively reduces overall service tardiness while ensuring timely order fulfillment and maintaining solution feasibility across all operational constraints. Furthermore, heuristic initialization demonstrates superior

consistency and reliability, as evidenced by the significantly smaller standard deviations in objective values. In 100-C instances, the standard deviation decreases from 2.86×10^5 with random initialization to 2.81×10^3 with heuristic initialization. This substantially lower variance indicates stable and predictable performance, making the heuristic approach more suitable for practical applications where solution reliability is paramount. In contrast, random initialization exhibits high volatility, underscoring its limitations for real-world deployment.

5.2.2 Impact analysis of algorithm components

To further validate the effectiveness of key components in our proposed algorithm, we conducted an additional ablation study examining the impact of removing the Asymmetric Traveling Salesman Problem (RSO) optimization module and the perturbation mechanism. Table 2 presents the comparative results across different instance categories.

Table 2: Ablation study on algorithm components

Instance	w/o RSO				w/o Perturbation			
	Worse	Better	Equal	%	Worse	Better	Equal	%
100-C	14/17	3/17	0/17	4.97%	15/17	2/17	0/17	2.42%
100-R	22/23	1/23	0/23	4.06%	21/23	2/23	0/23	2.69%
100-RC	15/16	1/16	0/16	4.65%	14/16	2/16	0/16	3.72%
200-C	14/20	6/20	0/20	3.24%	16/20	4/20	0/20	3.91%
200-R	12/20	8/20	0/20	3.06%	18/20	2/20	0/20	3.47%
200-RC	16/20	4/20	0/20	3.33%	17/20	3/20	0/20	3.04%
400-C	14/20	2/20	3/20	3.25%	16/20	1/20	3/20	3.33%
400-R	15/20	1/20	4/20	3.72%	14/20	1/20	5/20	4.69%
400-RC	13/20	1/20	6/20	3.18%	15/20	1/20	4/20	3.83%

The ablation results in Table 2 highlight the essential roles of both algorithmic components, demonstrating the individual and synergistic contributions of the RSO optimization and perturbation mechanisms.

Evaluation of RSO Optimization for Energy-Efficient Routing

The removal of the RSO component consistently leads to significant performance degradation across all instance types and sizes. The empirical data show that solutions without RSO optimization perform worse in the vast majority of cases, 12/20 to 22/23 instances across different categories, with average performance deterioration ranging from 3.06% to 4.97%. This substantial decline underscores the indispensable role of RSO optimization in generating energy-efficient vehicle routes. The RSO component serves as the computational engine that minimizes energy consumption costs by solving the underlying routing subproblems with optimal sequencing. Despite handling routes of varying lengths and complexity, the RSO optimizer maintains computational efficiency and effectively balances solution quality with reasonable computational overhead. This component ensures that vehicles follow the most energy-efficient paths while

satisfying all operational constraints, making it fundamental to the overall system performance.

Evaluation of Perturbation Mechanism in Escaping Local Optima

The results demonstrate that removing perturbation leads to worse solutions in 14/16 to 18/20 cases, with performance degradation averaging between 2.42% and 4.69%. This consistent pattern confirms the perturbation mechanism's essential function in preventing premature convergence to local optima. The strategic perturbations enable the algorithm to escape local optima and explore previously inaccessible yet potentially promising regions of the solution space. This capability proves particularly valuable in complex instances with 400 requests, where the search landscape contains numerous local optima that can trap simpler algorithms.

The consistent performance degradation observed when removing either component highlights their powerful synergistic relationship within our proposed framework. The RSO optimization provides intensive local search capability, delivering high-quality refinements to vehicle routes and energy consumption patterns. Meanwhile, the perturbation mechanism ensures sustained global exploration capacity, continuously redirecting the search toward unexplored, potentially superior regions of the solution space. This complementary interaction establishes a well-balanced algorithmic framework that harmonizes intensification and diversification. The RSO component focuses on the exploitation of high-quality solutions, while the perturbation mechanism facilitates broad exploration across the solution landscape. Therefore, the strategies constitute a robust optimization engine capable of effectively managing the intricate trade-offs among route efficiency, energy consumption, and computational tractability in vehicle routing problems with time windows.

5.2.3 Comparison with State-of-the-Art Algorithms

To comprehensively assess the effectiveness of the proposed PVNS-RSO, extensive comparative experiments are conducted against two state-of-the-art algorithms, including CZW [31], an improved VNS incorporating balanced perturbation and memory-based local search, and CLZ algorithm [30], a VNS framework integrating multiple local search strategies and efficient disturbance mechanisms. Notably, CLZ achieved first place in the ICAPS 2021 competition and represents the current state-of-the-art for solving complex pickup and delivery problems with capacity and LIFO constraints. Figure 8 demonstrates superior performance and robustness of the proposed algorithm across diverse instances.

Comparison in solution quality

For smaller instances with 100 requests, PVNS-RSO achieves solution quality comparable to CLZ while significantly outperforming CZW. This advantage becomes more evident as problem complexity increases. For medium-sized instances with 200 requests, PVNS-RSO demonstrates a clear performance, yielding substantially lower objective values than both baselines across all distribution patterns. The improvement is particularly notable in the 200-RC category, where the performance gain is most

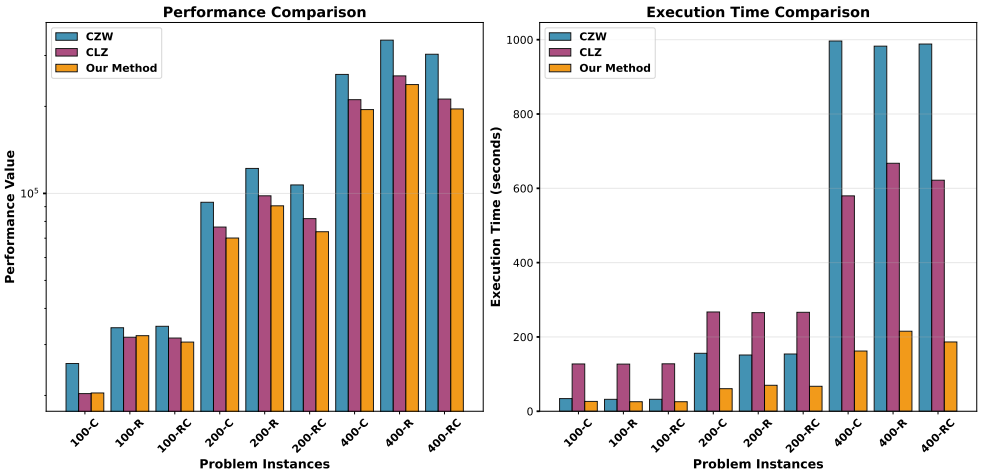


Fig. 8: Performance comparison with state-of-the-art algorithms. Our method consistently demonstrates superior performance across most problem instances, particularly excelling in larger and more complex scenarios.

pronounced. In large-scale instances with 400 requests, PVNS-RSO exhibits outstanding scalability, maintaining low objective values even as CZW and CLZ experience considerable degradation. The most substantial improvements occur in the 400-RC category, where the proposed method reduces objective values by approximately 5% relative to CZW and 8% relative to CLZ.

Comparison in computational efficiency

Beyond solution quality, the proposed algorithm exhibits remarkable computational efficiency that further enhances the practical applicability. PVNS-RSO consistently requires significantly less computation time than both comparative methods across all instances. For 100-request instances, the proposed method achieves an average runtime of 25.97 seconds, representing a 22.8% reduction compared to CZW and 79.5% reduction compared to CLZ. On the 200-request instances, PVNS-RSO is 2.3 times faster than CZW and 4.0 times faster than CLZ. The performance gap is most pronounced on the largest 400-request instances, where the proposed method with 187.97 seconds outperforms CZW and CLZ by a factor of 5.3 and 3.3, respectively. Notably, these significant computational efficiency gains are achieved while simultaneously delivering improved solution quality across all instance types

This combination of speed and accuracy arises from the algorithm's well-balanced design. The heuristic initialization provides high-quality starting points, reducing the need for extensive search. The strategic integration of perturbation and RSO optimization enables rapid convergence to near-optimal solutions without excessive computational overhead. The effectiveness of the approach can be attributed to the seamless interaction among the core components: the tree-based representation

ensuring feasibility, the VNS providing thorough exploration, the RSO optimization enabling intensive exploitation, and the perturbation mechanism maintaining diversity. Therefore, this design effectively balances the competing demands of solution quality and computational efficiency, achieving notable improvements over existing state-of-the-art methods. Figure 8 illustrates that the PVNS-RSO algorithm delivers substantial advances in solving complex vehicle routing problems, particularly for large-scale real-world applications where both solution quality and computational efficiency are critical.

Impact of problem parameters on algorithm performance

To further evaluate the practical applicability and efficiency of the proposed PVNS-RSO algorithm, this paper conducts comprehensive experiments analyzing the performance under reduced vehicle fleet scenarios. This analysis is particularly relevant for real-world logistics operations where optimizing fleet utilization can lead to significant cost savings and improved sustainability, including two vehicle reduction scenarios: 50% and 80% reduction from the original fleet size. These scenarios represent challenging operational conditions where efficient route planning becomes critical to maintain service quality with limited resources.

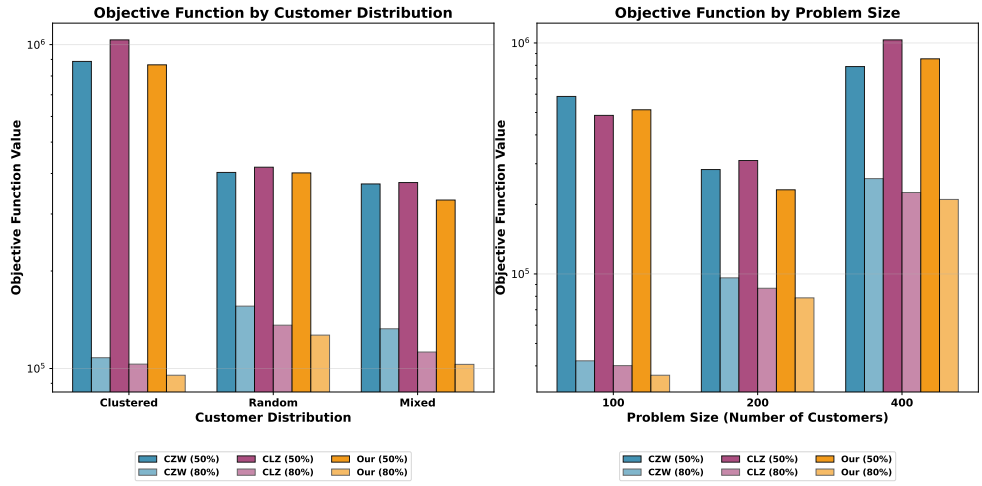


Fig. 9: Performance comparison under vehicle fleet reduction scenarios: (left) by customer distribution type, (right) by problem size. Our PVNS-RSO algorithm demonstrates superior performance in both 50% and 80% vehicle reduction scenarios across all distribution patterns and problem scales.

Impact of the number of vehicles on datasets with different customer distributions: The experimental results in Figure 9 demonstrate a significant correlation between vehicle availability and solution quality across all algorithms. It is emphasized that reducing the number of available vehicles leads to substantially

increased objective function values, reflecting the heightened challenge of serving the same customer demand with limited resources. A particularly notable observation is the dramatic impact of vehicle reduction on smaller problem instances. For 100-customer instances, reducing the fleet to 50% of the original size causes an exponential increase in objective function values across all algorithms. This sharp deterioration indicates that smaller instances lack the flexibility to absorb significant resource constraints while maintaining service quality. However, PVNS-RSO exhibits remarkable resilience under the more severe 80% vehicle reduction scenario. Despite the extreme constraints, the proposed method maintains superior performance compared to both CZW and CLZ across all distribution types. This robustness can be attributed to the effective integration of tree-based representation and RSO optimization, which enables efficient resource allocation even under tight constraints.

Impact of the number of vehicles on datasets with different problem sizes: For 100-customer instances, the 50% vehicle reduction produces the most dramatic performance degradation across all methods. However, the proposed algorithm demonstrates the smallest relative deterioration, maintaining its competitive edge over benchmark methods. In 200-customer instances, PVNS-RSO demonstrates exceptional performance in the 50% reduction scenario, achieving the best results among all compared methods. This suggests that PVNS-RSO is particularly effective at handling medium-scale optimization problems under substantial resource constraints. For the largest 400-customer instances, the proposed method maintains consistent superiority in both reduction scenarios. The performance advantage is most pronounced in the 80% reduction case, where PVNS-RSO significantly outperforms both competitors. This pattern underscores the scalability of our approach and its ability to manage complex routing challenges with severely limited resources.

Therefore, fleet reduction intensifies the difficulty of maintaining service quality, particularly with respect to time window satisfaction. However, PVNS-RSO effectively addresses this challenge through a combination of adaptive perturbation and route sequencing optimization. The perturbation mechanism resolves penalty hotspots by reconstructing overloaded routes, while the RSO component improves temporal coordination to minimize lateness. Furthermore, the integrated objective design enables the algorithm to balance energy efficiency and service quality, even when resources are severely constrained. These mechanisms collectively explain the strong resilience and superior performance of PVNS-RSO in penalty-sensitive, reduced-fleet scenarios.

6 Conclusions

This paper investigated the E-PDPTW, aiming to minimize both total energy consumption and service lateness in logistics operations. To effectively solve this complex optimization problem, we proposed a novel algorithm, called PVNS-RSO, which integrates the perturbation-based VNS with a route sequencing optimization algorithm. Additionally, the method implements a tree-based representation to ensure feasibility, multiple neighborhood structures for diversified exploration, and an RSO module for local refinement. Extensive experiments on standard benchmark instances

demonstrate that PVNS-RSO achieves superior performance compared with state-of-the-art approaches in terms of both solution quality and computational efficiency, leading to the potential of the proposed approach to enhance sustainability and operational efficiency in modern logistics networks. For future research, several directions can be explored. First, the proposed PVNS-RSO framework can be extended to multi-objective formulations that jointly consider fuel consumption, carbon emissions, and operational costs. Second, incorporating stochastic travel times or uncertain energy parameters would further enhance the model's applicability to real-world conditions. Finally, integrating the proposed algorithm with real-time data and online decision-making strategies could enable adaptive routing in dynamic logistics environments.

References

[1] Sar, K., Ghadimi, P.: A systematic literature review of the vehicle routing problem in reverse logistics operations. Computers & Industrial Engineering **177**, 109011 (2023)

[2] Hao, Y., Chen, Z., Sun, X., Tong, L.: Planning of truck platooning for road-network capacitated vehicle routing problem. Transportation Research Part E: Logistics and Transportation Review **194**, 103898 (2025)

[3] Wang, Y., Gou, M., Luo, S., Fan, J., Wang, H.: The multi-depot pickup and delivery vehicle routing problem with time windows and dynamic demands. Engineering Applications of Artificial Intelligence **139**, 109700 (2025)

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

[5] Rios, B.H.O., Xavier, E.C.: Metaheuristic approaches for the stochastic capacitated multi-depot vehicle routing problem with pickup and delivery. Expert Systems with Applications, 128258 (2025)

[6] Alzate, P., Londoño, G.A.M., Carrasco, J.M.S., Isaza, G.A., Toro, E.M., Jaramillo-Garzón, J.A.: Scientific mapping and research perspectives of the vehicle routing problem: An approach from sustainability strategies. Sustainable Futures, 100390 (2024)

[7] Osorio-Castañeda, C.D., Orejuela-Cabrera, J.P., Bravo-Bastidas, J.J.: Refrigerated electric vehicle routing considering time-dependent temperature energy consumption models. Annals of Operations Research **349**(3), 1817–1854 (2025)

[8] Zhang, W., Li, S., Deng, M., Mu, Y., Li, P.: Multi-stage hybrid multiobjective evolutionary algorithm for multi-type vehicle routing problem with simultaneous pickup and delivery and time windows. Computers & Industrial Engineering, 111242 (2025)

[9] Yang, S., Xia, J., Guo, S., Fang, X., Ni, C.: A novel time-dependent electric vehicle pickup and delivery problem with time windows in cold supply chains. Expert Systems with Applications, 128813 (2025)

[10] Sadati, M.E.H., Çatay, B.: A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem. Transportation Research Part E:

Logistics and Transportation Review **149**, 102293 (2021)
[11] Qi, R., Li, J.-q., Wang, J., Jin, H., Han, Y.-y.: Qmoea: A q-learning-based multi-objective evolutionary algorithm for solving time-dependent green vehicle routing problems with time windows. Information Sciences **608**, 178–201 (2022)
[12] Wang, W., Li, Y., Yan, H., Zhao, W., Zhao, Q., Luo, K.: A two-phase algorithm for the dynamic time-dependent green vehicle routing problem in decoration waste collection. Expert Systems with Applications **262**, 125570 (2025)
[13] Amiri, A., Amin, S.H., Zolfagharinia, H.: A bi-objective green vehicle routing problem with a mixed fleet of conventional and electric trucks: Considering charging power and density of stations. Expert Systems with Applications **213**, 119228 (2023)
[14] Su, Y., Zhang, S., Wang, Y., Cui, X., Wu, Y.: An adaptive large neighborhood search with multi-deletion operators for multi-depot green vehicle routing problem with time windows. Swarm and Evolutionary Computation **95**, 101942 (2025)
[15] Gutiérrez-Jarpa, G., Desaulniers, G., Laporte, G., Marianov, V.: A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows. European Journal of Operational Research **206**(2), 341–349 (2010)
[16] Karaoglan, I., Altiparmak, F., Kara, I., Dengiz, B.: A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery. European Journal of Operational Research **211**(2), 318–332 (2011)
[17] Capelle, T., Cortés, C.E., Gendreau, M., Rey, P.A., Rousseau, L.-M.: A column generation approach for location-routing problems with pickup and delivery. European Journal of Operational Research **272**(1), 121–131 (2019)
[18] Fu, Z., Chow, J.Y.: The pickup and delivery problem with synchronized en-route transfers for microtransit planning. Transportation Research Part E: Logistics and Transportation Review **157**, 102562 (2022)
[19] Agrali, C., Lee, S.: The multi-depot pickup and delivery problem with capacitated electric vehicles, transfers, and time windows. Computers & Industrial Engineering **179**, 109207 (2023)
[20] Zhou, S., Zhang, D., Yuan, W., Wang, Z., Zhou, L., Bell, M.G.: Pickup and delivery problem with electric vehicles and time windows considering queues. Transportation Research Part C: Emerging Technologies **167**, 104829 (2024)
[21] Cui, H., Qiu, J., Cao, J., Guo, M., Chen, X., Gorbachev, S.: Route optimization in township logistics distribution considering customer satisfaction based on adaptive genetic algorithm. Mathematics and Computers in Simulation **204**, 28–42 (2023)
[22] Wang, H., Chen, S., Yin, X., Meng, L., Wang, Z., Wang, Z.: A hybrid fuzzy c-means heuristic approach for two-echelon vehicle routing with simultaneous pickup and delivery of multicommodity. IEEE Transactions on Fuzzy Systems **33**(1), 218–230 (2024)
[23] Ren, T., Luo, T., Jia, B., Yang, B., Wang, L., Xing, L.: Improved ant colony optimization for the vehicle routing problem with split pickup and split delivery. Swarm and Evolutionary Computation **77**, 101228 (2023)
[24] Xu, X., Wei, Z.: Dynamic pickup and delivery problem with transshipments and

lifo constraints. Computers & Industrial Engineering **175**, 108835 (2023)
[25] Bustos-Coral, D., Costa, A.M.: Adaptive large neighborhood search for drayage routing problems involving longer combination vehicles. Computers & Operations Research **173**, 106826 (2025)
[26] Cai, J., Zhu, Q., Lin, Q., Ming, Z., Tan, K.C.: Decomposition-based multiobjective evolutionary optimization with tabu search for dynamic pickup and delivery problems. IEEE Transactions on Intelligent Transportation Systems **25**(10), 14830–14843 (2024)
[27] Leelertkij, T., Buddhakulsomsiri, J., Huynh, V.-N.: A multi-thread simulated annealing for multi-objective vehicle routing problem with time windows and demand priority. Computers & Industrial Engineering, 111253 (2025)
[28] Öztaş, T., Tuş, A.: A hybrid metaheuristic algorithm based on iterated local search for vehicle routing problem with simultaneous pickup and delivery. Expert Systems with Applications **202**, 117401 (2022)
[29] Gao, J., Zhen, L., Wang, S.: Multi-trucks-and-drones cooperative pickup and delivery problem. Transportation Research Part C: Emerging Technologies **157**, 104407 (2023)
[30] Cai, J., Zhu, Q., Lin, Q.: Variable neighborhood search for a new practical dynamic pickup and delivery problem. Swarm and Evolutionary Computation **75**, 101182 (2022)
[31] Chen, W., Zhang, D., Van Woensel, T., Xu, G., Guo, J.: Green vehicle routing using mixed fleets for cold chain distribution. Expert Systems with Applications **233**, 120979 (2023)
[32] Horváth, M., Kis, T.: On recent computational results for a dynamic pickup and delivery problem. Memetic Computing **17**(2), 27 (2025)
[33] Soroka, A., Meshcheryakov, A., Gerasimov, S.: Deep reinforcement learning for the capacitated pickup and delivery problem with time windows. Pattern Recognition and Image Analysis **33**(2), 169–178 (2023)
[34] Tian, R., Sun, Z., Chang, L., Wu, J., Lu, X.: Rapid solution for flexible pickup and delivery services problem based on improved actor-critic deep reinforcement learning. IEEE Transactions on Intelligent Transportation Systems (2025)
[35] Bogyrbayeva, A., Meraliyev, M., Mustakhov, T., Dauletbayev, B.: Machine learning to solve vehicle routing problems: A survey. IEEE Transactions on Intelligent Transportation Systems **25**(6), 4754–4772 (2024)
[36] Pugliese, L.D.P., Ferone, D., Festa, P., Guerriero, F., Macrina, G.: Combining variable neighborhood search and machine learning to solve the vehicle routing problem with crowd-shipping. Optimization Letters **17**(9), 1981–2003 (2023)
[37] Zhang, K., Li, M., Wang, J., Li, Y., Lin, X.: A two-stage learning-based method for large-scale on-demand pickup and delivery services with soft time windows. Transportation Research Part C: Emerging Technologies **151**, 104122 (2023)
[38] Phan Duc, H., Bui Trong, D., Nguyen Thi, T., Huynh Thi Thanh, B.: Pareto front grid guided multiobjective optimization in dynamic pickup and delivery problem considering two-sided fairness. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 277–285 (2025)

Declarations

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Authors' contribution statements

- **Bui Dinh Pham**: Algorithm, Software implementation, Experiments, Writing – original draft.
- **Nguyen Thi Tam**: Investigation, Methodology, Algorithm, Software implementation, Writing – original draft, Supervision.
- **Bui Trong Duc**: Investigation, Algorithm, Software implementation, Experiments, Writing – original draft.
- **Huynh Thi Thanh Binh**: Methodology, Algorithm, Writing – original draft, Supervision.

Data availability

The source codes and datasets for this paper can be retrieved from the following link:

https://github.com/cssartori/pdptw-instances
https://github.com/Pham2004/PVNS-RSO