Transportation Research Part C 190 (2026) 105796

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Priority-based multi-task scheduling for shared micro-mobility systems

Yimeng Zhang [a,b,c,d], Shuyang Zhu [e,a], Yuchun Chen [f], Peimin Li [a], Jie Feng [g], Ruijie Li [a,*], Xiaobo Liu [a,b,c,d], Mi Gan [a,b,c,d], Shuwen Yang [f], Nanxi Chen [f], Ruixue Ai [h]

[a] School of Transportation & Logistics, Southwest Jiaotong University, Chengdu, China
[b] National Engineering Laboratory of Integrated Transportation Big Data Application Technology, Southwest Jiaotong University, Chengdu, China
[c] Intelligent Comprehensive Transportation Key Laboratory of Sichuan Province, Chengdu, China
[d] National and Local Joint Engineering Research Center of Integrated Transportation Intelligence, Chengdu, China
[e] Department of Data & Systems Engineering, The University of Hong Kong, Hong Kong, China
[f] The Institute of Smart City and Intelligent Transportation, Southwest Jiaotong University, Chengdu, China
[g] School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, China
[h] University of Oslo, Oslo, Norway

ARTICLE INFO

Keywords:
Sustainable urban mobility
Shared micro-mobility
Multi-commodity vehicle rebalancing
Multi-task scheduling
Adaptive large neighborhood search

ABSTRACT

Shared micro-mobility systems advance sustainable transport by promoting environmentally friendly travel while enhancing convenience, accessibility, and operational efficiency. Most existing literature focuses primarily on vehicle rebalancing to improve shared micro-mobility system efficiency. However, system effectiveness cannot be achieved through rebalancing alone without ensuring reliable maintenance operations and battery functionality. To address these integrated challenges, this paper proposes a multi-task scheduling framework that unifies vehicle rebalancing, maintenance, battery replacement, and priority-based resource allocation mechanisms. We formulate a Mixed-Integer Programming (MIP) model and employ an Adaptive Large Neighborhood Search (ALNS) heuristic with customized operators to manage computational complexity effectively. Numerical experiments conducted on instances with varying request volumes demonstrate the superior performance of the proposed framework. The multi-task scheduling approach incorporating maintenance and battery replacement significantly outperforms single-task methodologies, substantially improving both availability and service rates of shared vehicles. We explore a range of high-priority request ratios, revealing the inherent trade-offs between service quality and operational cost. We further evaluate model performance under both concentrated and dispersed inventory-demand distribution scenarios. These findings provide practical insights for shared micro-mobility operators in designing strategic fleet deployment, optimizing rebalancing schedules, and allocating maintenance and battery resources effectively under diverse demand patterns and operational conditions.

1. Introduction

The transition to sustainable transport represents a critical global challenge, as the transport sector accounts for nearly one-quarter of energy-related CO_2 emissions (International Energy Agency, 2023). Among proposed solutions, shared micro-mobility systems have

* Corresponding author.
E-mail address: ruijie-li@swjtu.edu.cn (R. Li).

https://doi.org/10.1016/j.trc.2026.105796
Received 23 October 2025; Received in revised form 11 March 2026; Accepted 3 June 2026

emerged as a pivotal approach to urban green travel (Zhang and Mi, 2018). The global shared micro-mobility market demonstrates substantial growth potential, with penetration rates projected to increase from 11.4% in 2025 to 14.7% in 2030, ultimately reaching 1.2 billion users worldwide (Statista, 2025).

A fundamental challenge confronting shared micro-mobility systems is vehicle distribution imbalances, wherein certain areas experience oversupply while others face critical shortages. Most existing research focuses primarily on rebalancing strategies to address this spatial mismatch (Caggiani et al., 2018). However, without adequate maintenance protocols (Jin et al., 2022) and timely battery replacement operations (Zhang et al., 2020), rebalancing alone cannot prevent elevated idle rates, diminished vehicle turnover, and escalating operational costs (Guo et al., 2022), ultimately compromising both service quality and long-term system sustainability. To overcome these multifaceted limitations, this study proposes a multi-commodity priority-based multi-task scheduling framework that integrates vehicle rebalancing, maintenance operations, battery replacement, and priority-based resource allocation mechanisms. This unified approach aims to enhance operational efficiency while strengthening overall system resilience and service reliability.

As illustrated in Fig. 1(a), single-task scheduling focuses exclusively on vehicle rebalancing without considering maintenance requirements. Consequently, trucks departing from node d_3 transport normal shared mopeds together with malfunctioning and low-battery units to node d_1. As a result, non-functional vehicles are delivered to users, causing customer dissatisfaction. This problem persists when trucks leaving node d_1 carry both operational and malfunctioning shared bicycles to node d_2, further compromising service quality. In contrast, Fig. 1(b) demonstrates how multi-task scheduling effectively coordinates all operational tasks. At node d_3, only operational shared mopeds are transported to node d_1, while malfunctioning or low-battery units are redirected to the multi-task node t_2 for repair and battery replacement. Following restoration, these mopeds are then delivered to node d_1 as fully functional units. Similarly, trucks at node d_1 deliver operational shared bicycles directly to node d_2, while routing malfunctioning bicycles to maintenance node t_1 for repair before final delivery to node d_2. This approach ensures consistent service quality and customer satisfaction across all demand nodes.

In addition to multi-task scheduling, demand prioritization is essential for further enhancing system responsiveness and operational efficiency. As illustrated in Fig. 2(a), the traditional non-prioritized scheduling mechanism treats all requests equally, resulting in potential delays on urgent tasks. For instance, under this approach, the truck arrives at node d_2 at 11:00, node d_1 at 13:30, and node d_3 at 14:50. This sequence means that low- and medium-urgency requests are completed well before the high-urgency request at node d_3, leading to significant delays. In contrast, Fig. 2(b) demonstrates the prioritized scheduling mechanism, where request urgency levels directly determine the service sequence. Under this approach, the truck prioritizes node d_3, arriving as early as 10:30 to address the high-urgency request, then proceeds to node d_1 at 13:30 for the medium-urgency request, and finally visits node d_2 at 14:50 for the low-urgency request. This prioritization mechanism ensures that the most critical requests receive immediate attention, effectively minimizing service delays.

These comparisons highlight that both multi-task scheduling and prioritization are crucial for improving the responsiveness and efficiency of shared micro-mobility systems. Nevertheless, most existing studies either optimize these tasks separately (Chiariotti et al., 2018; De Chardon et al., 2016; Zhang et al., 2022) or fail to incorporate priority-based mechanisms (Jin et al., 2019), which limits their applicability in real-world operations. To bridge this gap, this study proposes an integrated optimization framework that simultaneously addresses multiple operational tasks while accounting for task urgency. Fig. 3 illustrates the overall operational process of this framework. The process begins with the reception of all requests at nodes (Step 1), followed by the identification and classification of different operational tasks, including vehicle rebalancing, maintenance, and battery replacement, as well as request types, including normal shared bicycles and mopeds, malfunctioning shared bicycles and mopeds, and low-battery mopeds (Step 2). These tasks and request types are then integrated into an optimization framework, where the ALNS algorithm coordinates vehicle scheduling under multiple practical constraints (Step 3). Following optimization, the system generates the optimal scheduling solution (Step 4). Finally, trucks execute the transport operations to different nodes according to the optimized schedules (Step 5).

The main contributions of this paper are summarized as follows: (a) developing a Mixed-Integer Programming (MIP) model for the Multi-commodity Priority-based Multi-task Scheduling Problem (MPMSP) in shared micro-mobility systems, which integrates multi-commodity operations, vehicle rebalancing, maintenance, battery replacement, and priority-based mechanisms into an operationally coupled optimization framework, thereby bridging traditionally separated tasks and better capturing the operational complexity of micro-mobility systems; (b) introducing an Adaptive Large Neighborhood Search (ALNS) heuristic with customized operators to efficiently solve diverse instances, significantly reducing computational time and enhancing practical applicability; (c) conducting extensive computational experiments across diverse instances to validate the effectiveness of the proposed model and solution approach; and (d) providing managerial insights that support improved resource allocation and operational strategies in shared micro-mobility management.

The remainder of this paper is organized as follows: Section 2 reviews existing research on vehicle rebalancing, maintenance, and battery replacement. Section 3 provides a detailed description of the MPMSP in shared micro-mobility systems. Section 4 presents the MIP model formulation to address the MPMSP. Section 5 outlines the solution methodology, introducing an ALNS algorithm specifically tailored for the MPMSP. Section 6 presents experimental designs and analyzes the results. Finally, Section 7 summarizes the paper and discusses potential future research directions.

2. Literature review

The Vehicle Routing Problem (VRP) has been extensively studied in logistics and transportation, resulting in substantial advances in exact algorithms (Wang et al., 2025a) and metaheuristic methods (Yang et al., 2024; Xu et al., 2025; Maximo et al., 2025). To better capture real-world operational complexity, numerous VRP variants have been proposed, such as multi-depot VRPs (Kraiem

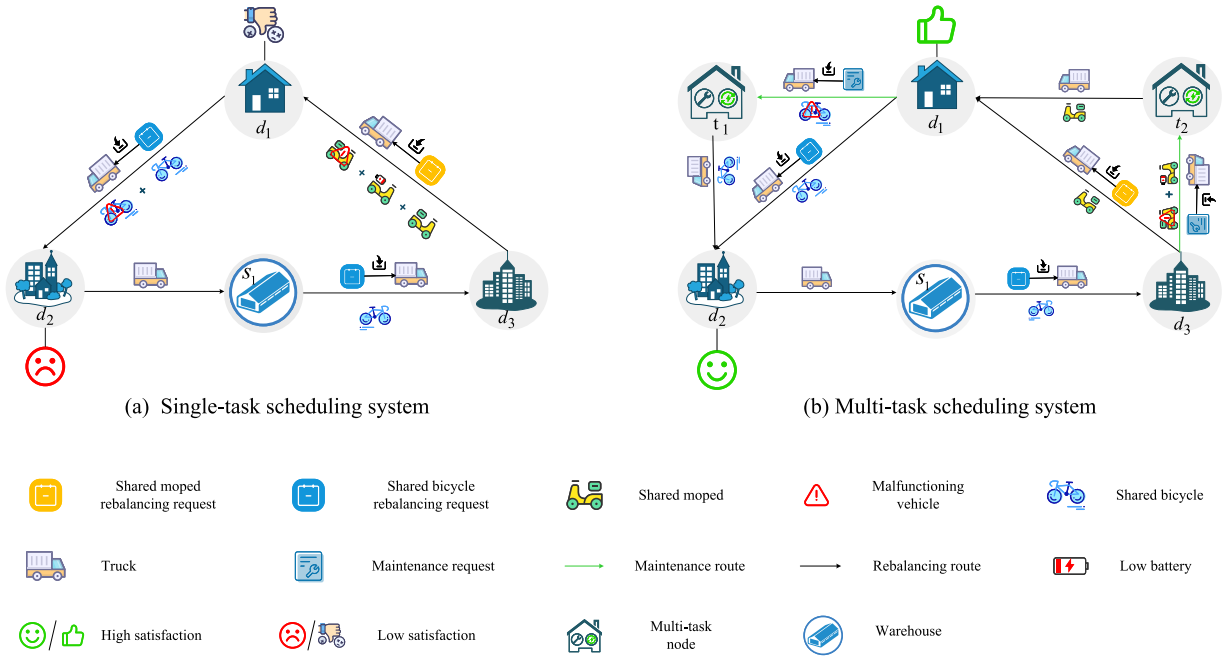


Fig. 1. Comparison of single-task and multi-task scheduling.

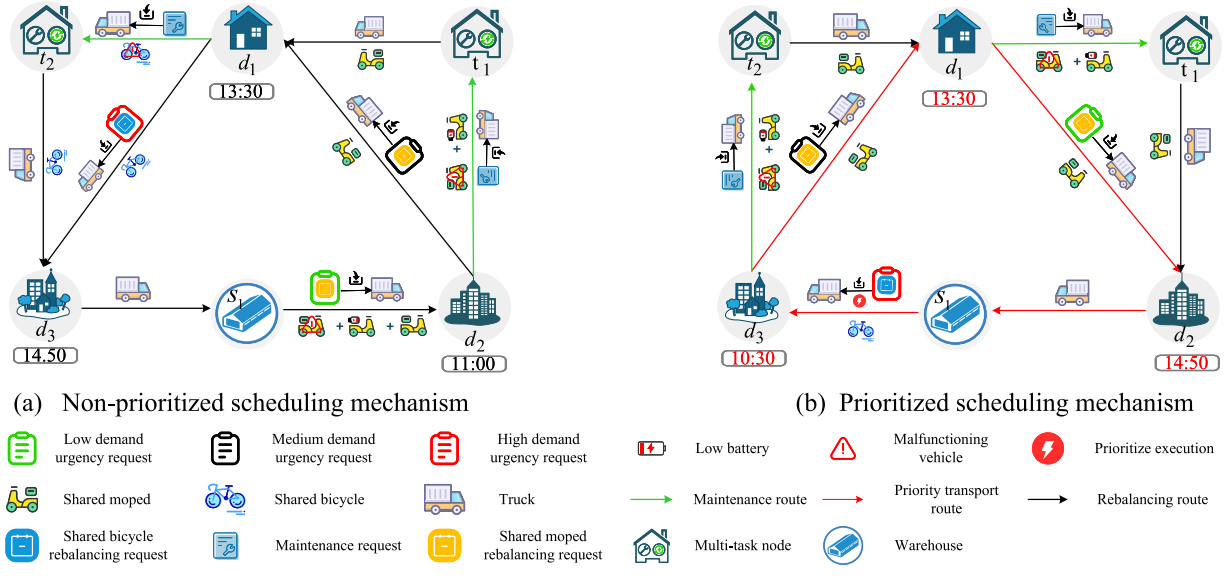


Fig. 2. Prioritization mechanisms in shared micro-mobility systems.

et al., 2025; Guan et al., 2025) and heterogeneous fleet VRPs (Wang et al., 2025a; Han and Yaman, 2024; Wang et al., 2025b; Pereira et al., 2024). In addition, VRPs with pickup and delivery have evolved to model complex real-world logistics, with recent advancements focusing on large-scale synchronous services (Cavaliere et al., 2024), multi-commodity optimization (Xu et al., 2024), drone routing (Jiang et al., 2024), the incorporation of locker networks (Dell'Amico et al., 2023), and multi-echelon structures with multi-trip planning (Lehmann and Winkenbach, 2024). Furthermore, studies have also addressed VRPs with time-related constraints, including time window assignment and delayed time-window assignment (Côté et al., 2024; Çelik et al., 2026; Wang et al., 2024).

Many operational problems in shared micro-mobility systems, including vehicle rebalancing, can be naturally formulated as VRP variants. To ground our problem within this broader methodological context and identify specific research gaps, this section reviews and synthesizes key literature in the field of shared micro-mobility systems. First, Section 2.1 analyzes the vehicle rebalancing problem in shared micro-mobility systems. Then, Section 2.2 examines maintenance issues and battery replacement in urban transport systems. Finally, Section 2.3 concludes with a summary that highlights the identified research gaps.

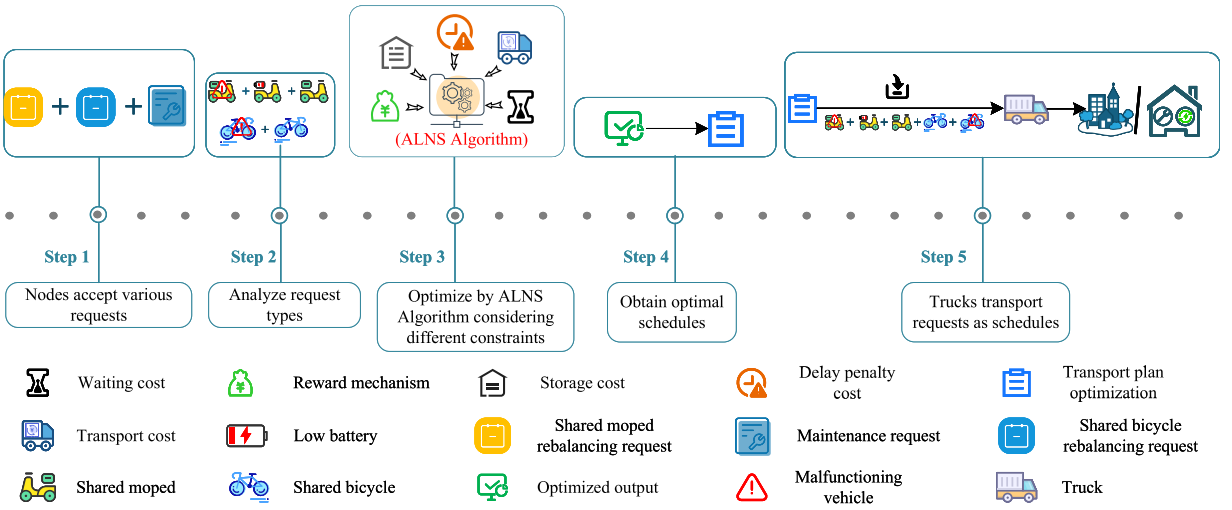


Fig. 3. Overall operation process of our proposed framework.

2.1. Vehicle rebalancing problem in shared micro-mobility systems

Early studies in shared mobility primarily address inventory management, exemplified by Nair and Miller-Hooks (2011), which develop a MIP model for optimizing node locations, capacities, and inventories. Subsequently, research attention expands to route optimization, particularly the vehicle rebalancing problem, which encompasses not only inventory management but also routing and scheduling decisions (Gammelli et al., 2022). Within this domain, both dynamic and static balancing models have been extensively studied. Static balancing problems (SBP) are typically performed overnight when traffic flow is minimal, yielding optimal initial configurations for the following day's mobility system (Raviv et al., 2013; Chemla et al., 2013; Ren et al., 2020; Zhang et al., 2018; Maggioni et al., 2019), whereas dynamic balancing operates continuously throughout the day (Gleditsch et al., 2024).

When addressing the bicycle rebalancing problem (BRP), several solution approaches have been explored. Some researchers employ exact methods (Dell'Amico et al., 2014; Erdoğan et al., 2015; Kadri et al., 2016; Kloimüllner and Raidl, 2017; Ren et al., 2020), including Row Generation (Fu et al., 2022) and Branch-and-Cut algorithms (Chastre and Andrade, 2023). Approximation techniques have also been extensively studied (Benchimol et al., 2011; Warrington and Ruchti, 2019; Brinkmann et al., 2020), such as the two-stage stochastic approximation algorithm for real-time vehicle repositioning proposed by Warrington and Ruchti (2019). Additionally, other studies adopt heuristic algorithms, including Tabu Search (Chemla et al., 2013), Neighborhood Search (Ren et al., 2020; Cruz et al., 2017; Ho and Szeto, 2017), Hybrid Parthenogenetic-genetic algorithms combined with Simulated Annealing (Liang et al., 2025), and Greedy-genetic Heuristics (Du et al., 2020).

At the problem level, BRP can be generally classified into user-based and operator-based strategies (Pan et al., 2019). User-based BRP has received relatively limited attention, with only a few studies (Fricker and Gast, 2016; Dötterl et al., 2017) proposing incentive mechanisms to encourage users to reposition bicycles across nodes. In contrast, operator-based rebalancing, which is directly managed by system operators, has been widely adopted (Singla et al., 2015; Shui and Szeto, 2018; Chastre and Andrade, 2023; Hu et al., 2021; Warrington and Ruchti, 2019). For example, Chastre and Andrade (2023) propose a static BRP explicitly designed for operator-based repositioning, which captures practical requirements of bicycle-sharing operators, including heterogeneous vehicles, vehicle autonomy, and route-duration limits.

BRP can be further classified into single-vehicle and multi-vehicle systems. A single-vehicle system refers to scenarios where each city district is served by one truck that independently handles bicycle redistribution (Chemla et al., 2013). Several studies have adopted this framework (Brinkmann et al., 2019; Legros, 2019; Pfrommer et al., 2014; Regue and Recker, 2014). However, control strategies that perform well in single-vehicle settings often become inefficient when extended to multi-vehicle systems, where multiple vehicles may converge on the same nodes and create operational conflicts (Chiariotti et al., 2018; Legros, 2019). Recent research addresses these limitations by developing coordinated multi-vehicle approaches that explicitly model vehicle interactions and jointly optimize routes. For example, Gleditsch et al. (2024) introduce a column-generation framework that integrates the routing of multiple vehicles, while Brinkmann et al. (2020) extend the dynamic single-vehicle model of Brinkmann et al. (2019) to a multi-vehicle setting.

Overall, BRP has been extensively studied, with solution methods ranging from exact algorithms, heuristics, and approximation approaches to machine learning techniques. At the problem level, existing studies encompass user-based and operator-based orientations, single- and multi-vehicle systems, and both static and dynamic planning approaches. Nevertheless, most existing studies on micro-mobility rebalancing focus solely on rebalancing itself, treating it in isolation from other operational tasks. They also restrict their scope primarily to bicycles, overlooking other vehicle types in shared micro-mobility systems.

2.2. Maintenance and battery replacement in urban transport

To sustain continuous and efficient operations, timely maintenance is essential. The existing literature is primarily devoted to facility location planning (Çelebi et al., 2018; Ma et al., 2023; Shoushtari et al., 2024); however, only a few studies have addressed maintenance issues by formulating them as routing problems (López-Santana et al., 2023).

In urban logistics, several studies propose different approaches to schedule maintenance operations, which can significantly affect overall delivery performance (Kim and Kim, 2021; Idris et al., 2022; Ndhaief et al., 2017). Among these, Ndhaief et al. (2017) investigate an integrated strategy that coordinates distribution and maintenance through two sequential plans. Building on these efforts, recent research has narrowed the focus from general urban logistics to specific shared mobility systems (Alvarez-Valdes et al., 2016; Chang et al., 2018; Zhang et al., 2018; Du et al., 2020; Chastre and Andrade, 2023). Specifically, Du et al. (2020) and Chastre and Andrade (2023) directly incorporate the collection of malfunctioning bicycles into their rebalancing optimization models.

With the growing emphasis on sustainability, many cities have adopted various electric vehicles. In this context, the management of vehicle batteries, particularly their replacement, has become a key research focus. Some studies concentrate on the strategic and tactical planning of charging and battery-replacement node locations (Wang and Lin, 2013; Sun et al., 2020; Janovec and Kohání, 2019). Others highlight the routing challenges of battery replacement, which are particularly acute in shared micro-mobility systems where mopeds are notable for their easily removable and portable batteries (Xu et al., 2022). To ensure shared electric mopeds always have sufficient battery power, Zheng et al. (2023) formulate a vehicle routing problem that integrates battery replacement and recycling in shared moped systems, while Jin et al. (2025) develop a two-stage stochastic programming model with an Alternating Direction Method of Multipliers method to jointly optimize vehicle and battery allocation under uncertainty. Similarly, Xu et al. (2022) design van-based mobile services that combine rebalancing and battery swapping, solved via a reinforcement learning framework.

Overall, although maintenance and battery replacement are crucial, particularly within shared mobility systems, they have received relatively limited attention. Motivated by this gap, this paper specifically addresses these issues in the context of shared micro-mobility systems.

2.3. Summary

Standard VRP variants typically assume independent tasks and homogeneous commodities. In contrast, our MIP formulation introduces a novel integration of multi-task operations, such as maintenance and battery replacement. These operations actively alter commodity states, transforming malfunctioning vehicles to a functional status at designated multi-task nodes. Consequently, our model features heterogeneous fleets managing diverse commodities across interdependent tasks. By explicitly capturing these operational dependencies, our formulation distinguishes itself from standard VRP variants in shared micro-mobility systems.

Table 1 summarizes representative studies on vehicle rebalancing, maintenance issues, and battery replacement, including their problem characteristics, objectives, models, and solution algorithms.

In general, the literature has primarily focused on vehicle rebalancing while largely overlooking multi-task scheduling that includes maintenance or battery replacement (Chemla et al., 2013; Warrington and Ruchti, 2019; Ren et al., 2020; Du et al., 2020; Zhang et al., 2023; Gleditsch et al., 2024; Brinkmann et al., 2020). As a result, research remains problem-specific, lacking a framework that addresses multiple operational tasks in shared mobility systems. While studies on maintenance and battery replacement are more common in the broader urban logistics field, they remain limited in the context of shared micro-mobility systems (Hsu et al., 2021; Yan et al., 2022; Guerrero et al., 2024).

Within the rebalancing problem, the scope remains limited in several key aspects. Many studies assume a single vehicle handles all rebalancing tasks, oversimplifying the operational reality of multi-vehicle coordination (Chemla et al., 2013; Warrington and Ruchti, 2019; Fu et al., 2022; Chastre and Andrade, 2023; Zhang et al., 2023; Chen et al., 2024; Jin et al., 2025; Hu et al., 2021; Lu et al., 2022). Additionally, few studies address multi-commodity flows Chastre and Andrade (2023), where different vehicle types (e.g., bicycles, mopeds) need to be jointly allocated within networks.

Moreover, priority mechanisms-essential for balancing urgent and flexible demands-have been largely overlooked, thereby limiting the applicability of existing models to scenarios with heterogeneous service requirements (Chemla et al., 2013; Warrington and Ruchti, 2019; Ren et al., 2020; Fu et al., 2022; Chastre and Andrade, 2023; Zhang et al., 2023; Chen et al., 2024; Jin et al., 2025; Zheng et al., 2023; Xu et al., 2022; Lu et al., 2022; Brinkmann et al., 2020).

Recent studies have gradually explored multi-task scheduling problems. However, Du et al. (2020) address only the rebalancing of normal and malfunctioning bicycles, omitting key factors such as battery replacement, multi-commodity transport, and priority mechanisms. Likewise, Xu et al. (2022) and Jin et al. (2025) consider battery swapping and rebalancing but neglect critical aspects including maintenance of malfunctioning vehicles, multi-commodity transport, priority mechanisms, and the allowance of multiple visits to a single node.

To the best of our knowledge, this paper is the first to integrate multi-commodity operations, vehicle rebalancing, maintenance issues, battery replacement, and priority-based mechanisms through an operationally coupled routing framework. Unlike conventional studies, the proposed framework captures interactions among these operations. First, because vehicles may require repair or battery replacement during rebalancing, route feasibility and task sequencing strictly require that vehicles must be restored to an operational state during the routing process. Second, repair and battery-swapping operations introduce inter-route dependencies, since vehicles collected by one truck may be processed at multi-task nodes and subsequently redistributed by another truck. This creates coordination requirements that cannot be decomposed into independent rebalancing and maintenance subproblems. Third, the priority mechanism is integrated into operational decisions rather than treated as a simple weighting scheme, as high-priority requests may trigger

Table 1
Literature review on vehicle rebalancing, maintenance and battery replacement.

Literature	Problem	Objective	Shared mobility	Rebalancing	Multi-vehicle	Multi-commodity	Maintenance	Battery	Priority	Model	Solution algorithm
Chemla et al. (2013)	SBRP	Min C	✓	✓						SVOCPDP	TS
Warrington and Ruchti (2019)	DRRP	Min C	✓	✓						TSSP	SPAR
Ren et al. (2020)	SBRP	Min C	✓	✓	✓					MIP&MILP	IGVNS
Du et al. (2020)	SFFCRP	Min T	✓	✓	✓		✓		✓	ILP	GGH&CPLEX
Fu et al. (2022)	SLRSPP	Max R	✓	✓						TRO	RG
Chastre and Andrade (2023)	BRP	Min D	✓	✓		✓	✓			MILP	BC
Zhang et al. (2023)	DBRP	Min W, Max R	✓	✓						BLPM	Python
Chen et al. (2024)	DDBRP	Max P	✓	✓			✓			TSEDRO	Gurobi
Jin et al. (2025)	TVBSDU	Min C	✓					✓		TSSP	ADMM
Hu et al. (2021)	MODBRP	Min C, Max S	✓	✓					✓	MILP	MOEA/D
Gleditsch et al. (2024)	DDBRP	Min C	✓	✓	✓					MIP	CGH
Zheng et al. (2023)	MDVRPSPD	Min T	✓		✓			✓		ILP	VLSM-EA
Xu et al. (2022)	VJRBRP	Max P	✓	✓	✓			✓		MDP	DDDQN
Lu et al. (2022)	FFBMAP	Min T	✓				✓			MIL-SOCP	ML
Brinkmann et al. (2020)	SDIRP	Min U	✓	✓	✓					MDP & ADP	CLP
Zhang et al. (2018)	SBRP-DB	Min C	✓	✓	✓		✓			MILP	DPSO-VNS
Hsu et al. (2021)	FLAP	Min C			✓		✓	✓		AILP	TS-DHA
Guerrero et al. (2024)	SRMPMP	Max P					✓		✓	IP-TOP	BRSBH-RL
This study	MPMSP	Max S, Min C	✓	✓	✓	✓	✓	✓	✓	MTC-MIP	ALNS

Problem: SBRP: Static bicycle rebalancing problem; DRRP: Dynamic rebalancing and routing problem; SFFCRP: Static free-floating complete rebalancing problem; SLRSPP: Node location and rebalancing service area partitioning problem; BRP: Bicycle rebalancing problem; DBRP: Dynamic bicycle rebalancing problem; DDBRP: Dynamic deterministic bicycle rebalancing probelm; TVBSDU: Two-stage electric vehicle and battery scheduling under demand uncertainty; MODBRP: Multiobjective dynamic bicycle rebalancing problem; MDVRPSPD: Multiple-deport vehicle rebalancing problem with simultaneous pickup and delivery; VJRBRP: Van-based joint rebalancing and battery replacement problem; FFBMAP: Free-float faulty bicycle maintenance assignment problem; SDIRP: Stochastic-dynamic inventory routing problem; SBRP-DB: Static bicycle rebalancing problem-Defective bicycles; FLAP: Facility location and allocation problem; SRMPMP: Scheduling and routing of multi-period maintenance problem; MPMSP: Multi-commodity priority-based multi-task scheduling problem.
Objective: R: Revenue/Reward; C: Cost; D: Distance; T: Time; P: Profit; W: Workload; U: Unsatisfied demand; S: Satisfaction/Service rate;
Model: SVOCPDP: Single vehicle one-commodity capacitated pickup and delivery problem; TSSP: Two-stage stochastic programming; MIP: Mixed integer programming; MILP: Mixed integer linear programming; ILP: Integer linear programming; TRO: Two-stage robust optimization; BLPM: Bi-level programming model; TSEDRO: Two-stage enhanced robust optimization model; MDP: Markov Decision Process; MIP-SOCP: Mix integer liner second-request conic programming; ADP: Approximate dynamic programming; AILP: Augemented integar liner program; IP-TOP: Integer programming model for the team orienteering problem.
Method: TS: Tabu search; SPAR: Separable, projective, approximation,routine; IGVNS: Improved general variable neighborhood search; GGH: Greedy-genetic heuristic; RG: Row generation; BC: Branch-and-cut; ADMM: Alternating direction method of multipliers; MOEA/D: Multiobjective evolutionary algorithm based on decoposition; CGH: Column generation heuristic; VLSM-EA: Varible local-search-based mutation evolutionary Algorithm; DDDQN: Double Dueling Deep Q-Network; ML: Machine learning; CLP: Coordinated Lookahead Policy; DPSO-VNS : Discrete particle swarm optimization combined with variable neighborhood search; TS-DHA: Tabu search-Decomposition-based heuristic algorithm; BRSBH-RL: Biased-randomized savings-based heuristic- reinforcement learning; ALNS: Adaptive large neighborhood search;

Table 2
Notation.

Sets:	
K	Set of trucks indexed by k. $K = K^{sb} \cup K^{sm} \cup K^{b\&m}$, where K^{sb}, K^{sm} and $K^{b\&m}$ are the sets of trucks responsible for shared bicycles only, for shared mopeds only, and for multiple commodities (able to carry both shared bicycles and mopeds), respectively.
R	Set of requests, $R = R_b^n \cup R_m^n \cup R_b^f \cup R_m^f \cup R_m^l$, where R_b^n denotes for normal shared bicycle set, R_m^n for normal shared moped set, R_b^f for malfunctioning shared bicycle set, R_m^f for malfunctioning shared moped set, and R_m^l for low-battery shared moped set.
N	Set of nodes indexed by i and j. $N = S \cup D \cup T$, where S, D and T are the sets of supply nodes, normal demand nodes restricted to vehicle rebalancing operations, and multi-task nodes capable of performing maintenance and battery replacement, respectively.
A	Set of arcs; for $i, j \in N$, the arc from node i to node j is indexed by $(i, j) \in A$. $A_p \subseteq A$ represents the set of pickup arcs where $i \in S$ and $A_d \subseteq A$ represents the set of delivery arcs where $j \in D$.
Parameters:	
u_k	Capacity (ton) of truck k.
v_k	Speed of truck k.
q_r	Quantity (ton) of request r.
d_{ij}^k	Travel distance (km) for truck k between nodes i and j .
τ_{ij}^k	Time (hours) required for truck k traveling from node i to node j.
pl_r	Priority level of request r; higher values denote more urgency.
c_k^1	The time-based unit transport cost (CNY/hour/ton) using truck k.
$c_k^{1'}$	The distance-based unit transport cost (CNY/km/ton) using truck k.
c_k^2	The unit loading/unloading cost (CNY/ton) using truck k.
c_k^3	The unit storage cost (CNY/hour/ton) using truck k.
c_k^4	The unit cost (CNY/hour) of waiting time using truck k.
c_r^1	The unit delay penalty (CNY/hour/ton) for request r.
s_r	Pickup node for request r.
d_r	Delivery node for request r.
s_i^k	Service duration time of truck k at node i.
n_i	Net demand at node i, defined as the total demand minus its current inventory.
Δ_{ij}	The total quantity of requests transported from node i to node j.
$[a_{s_r}, b_{s_r}]$	Feasible pickup time interval for request r.
$[a_{d_r}, b_{d_r}]$	Feasible delivery time interval for request r.
$o(k)/o'(k)$	The origin depot and destination deport for truck k.
M	A large enough positive number.
Decision Variables:	
x_{ij}^k	A binary decision variable that equals 1 if truck k uses arc (i, j), and 0 otherwise.
y_{ij}^k	A binary decision variable that equals 1 if node i is positioned prior to node j (not necessarily adjacent) in the path of vehicle k, and 0 otherwise.
z_{ij}^{kr}	A binary decision variable that equals 1 if request r is carried by truck k on arc (i, j), and 0 otherwise.
m_{ir}^{kl}	A binary decision variable that equals 1 if request r is unloaded from truck k, undergoes maintenance or battery replacement at multi-task node t, and is subsequently loaded onto truck l; and 0 otherwise.
w_r	A binary decision variable that equals 1 if request r is served, and 0 otherwise.
t_{ki}^{wait}	Waiting time of truck k at node i.
t_r^{delay}	Delivery delay time of request r.
$t_i^{kr}/t_i'^{kr}/\bar{t}_i^{kr}$	The arrival time/service initiation time/service completion time of request r served by truck k at node i.
$t_i^k/t_i'^k/\bar{t}_i^k$	The arrival time/service initiation time/departure time of truck k at node i.

immediate dispatch and reshape routing, maintenance scheduling, and fleet coordination. Together, these features establish a highly integrated operational framework, fundamentally setting our approach apart from traditional rebalancing and VRP models.

3. Problem description

All symbols used in this section are defined in Table 2.

Unlike the classical Pickup and Delivery Problem (PDP), the MPMSP features strong interdependence among nodes, since the amount picked up at oversupplied nodes directly affects deliveries to undersupplied ones. The problem is also closely associated with real-world constraints, such as node inventory, vehicle capacity, route accessibility, and request time windows. Additionally, a task priority mechanism is introduced, allowing urgent requests to be scheduled and transported ahead of lower-priority ones. For example, for the same shared micromobility vehicles, maintenance and battery replacement tasks should take priority over rebalancing.

The set of nodes in the MPMSP transport system is defined as $N = S \cup D \cup T$. The request set is defined as R, where each request r belongs to $\{r_b^n, r_m^n, r_b^f, r_m^f, r_m^l\}$. Here r_b^n and r_m^n represent normal shared bicycles and normal shared mopeds, r_b^f and r_m^f denote malfunctioning shared bicycles and malfunctioning shared mopeds that require maintenance at multi-task demand nodes T, and r_m^l corresponds to requests for low-battery shared mopeds that need battery replacement at T. Trucks in the MPMSP is defined as $K = K^{sb} \cup K^{sm} \cup K^{b\&m}$. The transport distance between nodes i and j for truck $k \in K$ is denoted by d_{ij}^k. The arrival and departure

times of truck k at node i are denoted by t_i^k and $\bar{t}_i^k$, respectively. The travel time τ_{ij}^k on arc (i,j) is determined by the distance d_{ij}^k between nodes i and j divided by the speed v_k of truck k ($\tau_{ij}^k = \frac{d_{ij}^k}{v_k}$).

Notably, we capture the supply-demand balance of multiple request categories between different nodes. For each node, the net demand n_i determines whether the node acts as a demand node ($n_i > 0$) or a supply node ($n_i < 0$), thereby enabling the representation of both deliveries and surplus redistribution. After each allocation process, the net demand of the supply and demand nodes is updated according to the allocated request quantity Δ_{ij}, which represents the total quantity of requests transported from node i to node j. In other words, Δ_{ij} reflects how resources are routed to maintain regional balance. The updated net demands for nodes i and j are denoted as n'_i and n'_j, which are calculated as follows:

$$n'_i = n_i - \Delta_{ij}, \quad n'_j = n_j + \Delta_{ij} \tag{1}$$

where n_i and n_j represent the original net demands of supply node i and demand node j, respectively.

Each request r is also assigned a priority level pl_r ($pl_r \in [1, 100]$), which indicates its urgency. A higher pl_r value corresponds to greater transport urgency, meaning that request r needs to be scheduled as a priority to ensure responsiveness without compromising overall system efficiency. Each request r has pre-scheduled time windows, with pickup occurring during $[a_{s_r}, b_{s_r}]$ and delivery taking place during $[a_{d_r}, b_{d_r}]$. In this problem, pickup operations are treated as hard constraints, while delivery operations are considered soft constraints. If the delivery is not completed within its designated time window, the request will incur a delay penalty. This penalty is calculated by multiplying the delay time t_r^{delay} ($t_r^{\text{delay}} = \max(0, t_i^{kr} - b_{d_r})$) by a penalty coefficient c_r^1, thus incentivizing the timely completion of requests.

Fig. 4 illustrates the multi-tasking scheduling with priority strategies under complex operational scenarios. A detailed interpretation of this figure is provided below:

(a) Vehicle rebalancing routes (Black lines): Since node d_4 lacks normal shared bicycles r_b^n, truck k departs from node s_1 loaded with r_b^n. After meeting the demand at d_4, k loads the shared mopeds provided (including both normal shared mopeds r_m^n and inevitably some malfunctioning shared mopeds r_m^f) and delivers them to d_3. Once the shortage of r_m^n at d_3 is satisfied, k loads the normal shared bicycles r_b^n supplied by d_3 and proceeds to d_5, where k further loads normal shared mopeds r_m^n and normal shared bicycles r_b^n. These are then delivered respectively to d_2 and d_1, after which k loads the malfunctioning shared bicycles r_b^f at d_1 and returns to node s_1.

(b) Multi-task function routes (Green lines): Truck k from node d_3 transports the received malfunctioning shared mopeds r_m^f, along with the node's locally available low-battery shared mopeds r_m^l, to the multi-task node t_2. Concurrently, k transports the node's malfunctioning shared bicycles r_b^f to the multi-task node t_1. At t_2, the malfunctioning shared mopeds r_m^f are repaired, and the low-battery shared mopeds r_m^l are recharged (batteries replaced). These are then redistributed as normal shared mopeds r_m^n to other nodes. Similarly, at t_1, the malfunctioning shared bicycles r_b^f are repaired and subsequently redistributed to d_2 as normal shared bicycles r_b^n. Notably, the multi-task nodes t_1 and t_2 are not limited to providing maintenance and battery replacement services; they are also fully capable of performing normal rebalancing tasks. For example, t_1 can receive normal shared bicycles r_b^n from d_3 and transport them to other nodes. Similarly, t_2 can also receive normal shared mopeds r_m^n from s_1 and d_5 for redistribution.

(c) High-priority emergency routes (Red lines): When node d_2 generates high-priority demand for normal shared bicycles r_b^n and normal shared mopeds r_m^n, the conventional route from d_5 is unable to respond effectively due to excessive transport time. Consequently, multi-task nodes t_1 and t_2 urgently dispatch repaired r_b^n and r_m^n to d_2. Simultaneously, owing to its geographical proximity to d_2, node d_1 immediately deploys its available r_m^n to d_2, thereby ensuring minimal response time for high-priority request fulfillment.

The model is developed under the following assumptions: The parameters, including demands, network conditions, the supply ability, and the priority level of all requests, are deterministic and known in advance. Additionally, costs associated with maintenance or battery replacement performed at multi-task function node are not considered.

4. Mathematical model

In this section, a MIP model is developed to address the MPMSP in the micro-mobility system. The following formulation uses the notation defined in Table 2. In our model, two objectives are considered, with Objective (2) taking precedence over Objective (3). Objective (2) (F) aims to maximize the number of served requests, giving priority to those with higher priority levels (pl_r). Objective (3) (F') seeks to minimize the total operational cost. The cost objective F' comprises transport cost F_1, loading/unloading cost F_2, storage cost F_3, waiting cost F_4, and delay penalty cost F_5, all of which are formulated in Eqs. (3)–(8).

$$\max F = \sum_{r \in R} pl_r w_r \tag{2}$$

$$\min F' = F_1 + F_2 + F_3 + F_4 + F_5 \tag{3}$$

$$F_1 = \sum_{k \in K} \sum_{(i,j) \in A} \sum_{r \in R} \left(c_k^1 \tau_{ij}^k + c_k^{1'} d_{ij}^k \right) q_r z_{ij}^{kr} \tag{4}$$

$$F_2 = \sum_{k \in K} \sum_{(i,j) \in A_p \cup A_d} \sum_{r \in R} c_k^2 q_r z_{ij}^{kr} \tag{5}$$

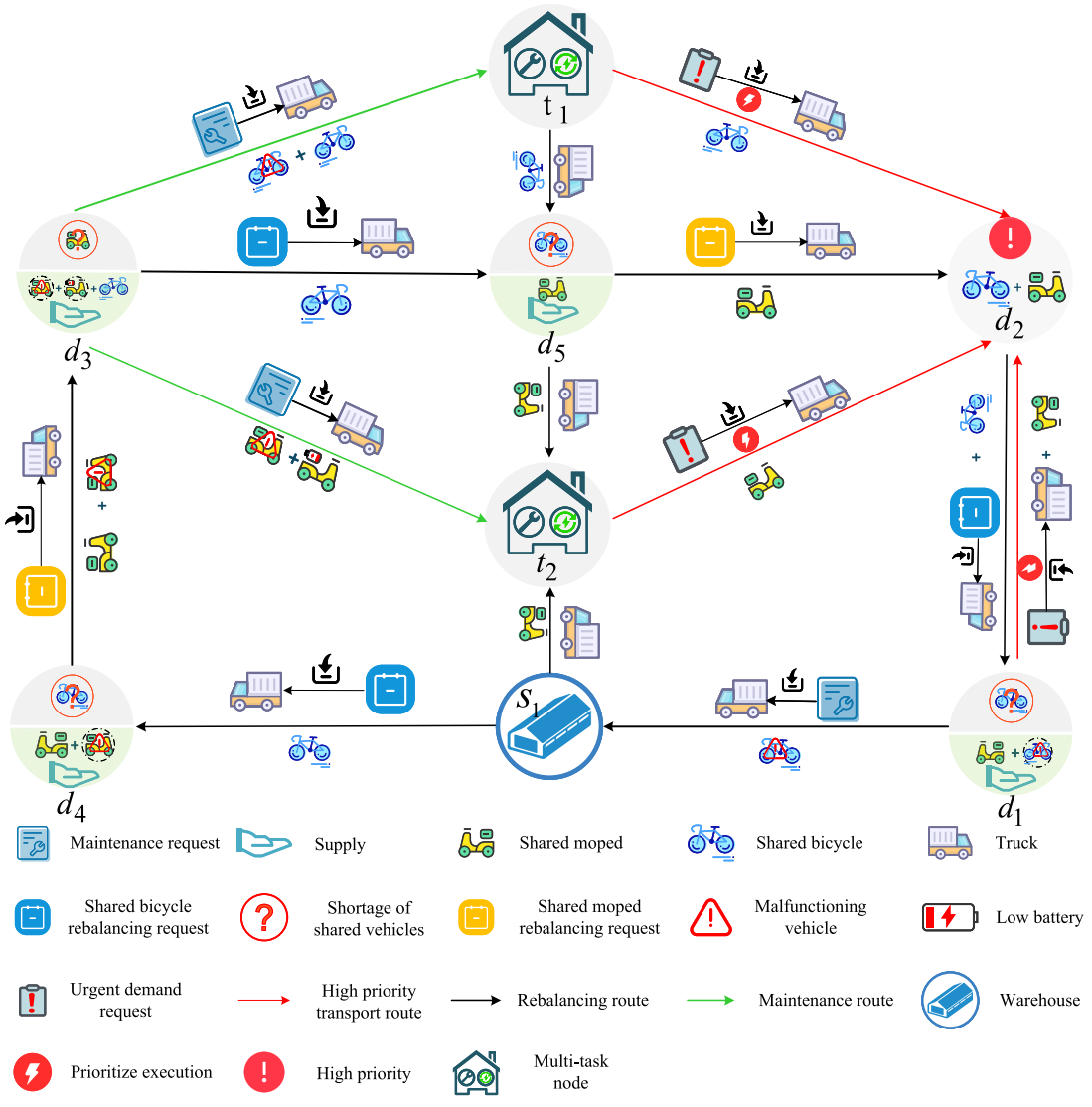


Fig. 4. MPMSP in shared micro-mobility systems.

$$F_3 = \sum_{k\in K}\sum_{(i,j)\in A_p}\sum_{r\in R} c_k^3 q_r z_{ij}^{kr}(t_i^{kr} - a_{s_r}) \tag{6}$$

$$F_4 = \sum_{k\in K}\sum_{i\in N} c_k^4 t_{ki}^{\text{wait}} \tag{7}$$

$$F_5 = \sum_{r\in R} c_r^1 q_r t_r^{\text{delay}} \tag{8}$$

It should be noted that x_{ij}^k and z_{ij}^{kr} are not redundant; they serve distinct and complementary purposes in our formulation. Specifically: (a) Trucks may need to travel empty (e.g., repositioning to a supply node). This requires setting $x_{ij}^k = 1$ while all $z_{ij}^{kr} = 0$, a scenario that cannot be represented if routing arcs are strictly tied to requests. (b) At multi-task nodes, a single truck k may transport multiple request types simultaneously (e.g., carrying malfunctioning mopeds for repair alongside operational ones for rebalancing). Distinguishing the specific request r onboard is critical for multi-task operations, necessitating the z_{ij}^{kr} variable with both k and r indices. (c) Priority-based costs, time windows, and delay penalties are intrinsically tied to individual requests. These parameters cannot be accurately computed without the z_{ij}^{kr} variable.Therefore, both variables are strictly essential for accurately modeling the complexities of the MPMSP.

Constraints (9)-(10) enforce route feasibility by restricting each truck k to depart from its origin depot $o(k)$ at most once and requiring that such a departure implies an arrival at its destination depot $o'(k)$.

$$\sum_{j\in S} x^k_{o(k)j} \le 1, \quad \forall k \in K \tag{9}$$

$$\sum_{j\in S} x^k_{o(k)j} = \sum_{j\in D} x^k_{jo'(k)}, \quad \forall k \in K \tag{10}$$

For each request $r \in R$, Constraints (11)-(12) enforce that at most one truck $k \in K$ is assigned to depart from its pickup node s_r and arrive at its delivery node d_r:

$$\sum_{k\in K}\sum_{j\in D} z^{kr}_{s_r j} \le 1, \quad \forall r \in R \tag{11}$$

$$\sum_{k\in K}\sum_{i\in S} z^{kr}_{i d_r} \le 1, \quad \forall r \in R \tag{12}$$

Constraints (13) ensure that a request can only be marked as served if it is transported on at least one arc by a truck, while Constraints (14) enforce that no transport arcs are assigned to an unserved request.

$$w_r \le \sum_{k\in K}\sum_{(i,j)\in A} z^{kr}_{ij}, \quad \forall r \in R \tag{13}$$

$$\sum_{k\in K}\sum_{(i,j)\in A} z^{kr}_{ij} \le M w_r, \quad \forall r \in R \tag{14}$$

Constraints (15) and (16) capture truck heterogeneity by ensuring vehicles are only assigned commodities they are equipped to handle. Constraints (15) prohibit trucks in K^{sb} from transporting any moped-related requests, while Constraints (16) prohibit trucks in K^{sm} from transporting any bicycle-related requests.

$$z^{kr}_{ij} = 0, \quad \forall (i,j) \in A,\ \forall k \in K^{\mathrm{sb}},\ \forall r \in R^n_m \cup R^f_m \cup R^l_m \tag{15}$$

$$z^{kr}_{ij} = 0, \quad \forall (i,j) \in A,\ \forall k \in K^{\mathrm{sm}},\ \forall r \in R^n_b \cup R^f_b \tag{16}$$

Constraints (17) through (19) govern maintenance and battery replacement operations. Specifically, Constraints (17) ensure that any request requiring these services must be routed through a multi-task node t. Constraints (18) guarantee that a request undergoes processing only once at a given multi-task node t, while Constraints (19) prohibit a request from being unloaded and subsequently reloaded onto the exact same vehicle k.

$$\sum_{i\in T} m^{kl}_{ir} \ge w_r \quad \forall r \in R^f_b \cup R^f_m \cup R^l_m,\ \forall k,l \in K \tag{17}$$

$$\sum_{j\in N} z^{kr}_{ji} + \sum_{j\in N} z^{lr}_{ij} \leqslant m^{kl}_{ir} + 1 \quad \forall r \in R^f_b \cup R^f_m \cup R^l_m,\ \forall i \in T,\ \forall k,l \in K \tag{18}$$

$$m^{kk}_{ir} = 0 \quad \forall r \in R^f_b \cup R^f_m \cup R^l_m,\ \forall i \in T,\ \forall k \in K \tag{19}$$

Constraint (20) enforces flow conservation by ensuring that, for each truck k, the number of arcs entering any intermediate node $i \in N \setminus \{o(k), o'(k)\}$ equals the number of arcs leaving it.

$$\sum_{i\in N} x^k_{ij} - \sum_{j\in N} x^k_{ji} = 0, \quad \forall k \in K \quad \forall i \in N \backslash o(k), o'(k) \tag{20}$$

As specified in Constraints (21), request flow balance requires that for each intermediate node $i \in N \setminus \{s_r, d_r\}$, the flow of requests entering and leaving the node is balanced:

$$\sum_{j\in N} z^{kr}_{ij} - \sum_{j\in N} z^{kr}_{ji} = 0, \quad \forall i \in N \setminus \{s_r, d_r\},\ \forall k \in K,\ \forall r \in R \tag{21}$$

Constraints (22) and (23) govern flow conservation for request r that requires neither maintenance nor battery replacement at multi-task node $i \in T$ when truck k visits i to serve other requests.

$$\sum_{j\in N} z^{kr}_{ij} - \sum_{j\in N} z^{kr}_{ji} \leqslant \sum_{l\in K} m^{lk}_{ir} \quad \forall k \in K,\ \forall r \in R^n_b \cup R^n_m,\ \forall i \in T \setminus s_r, d_r \tag{22}$$

$$\sum_{j\in N} z^{kr}_{ji} - \sum_{j\in N} z^{kr}_{ij} \leqslant \sum_{l\in K} m^{kl}_{ir} \quad \forall k \in K,\ \forall r \in R^n_b \cup R^n_m,\ \forall i \in T \setminus s_r, d_r \tag{23}$$

Constraints (24) enforce the consistency between z^{kr}_{ij} and x^k_{ij}, such that a request assigned to a vehicle must correspond to an arc that the same vehicle traverses.

$$z^{kr}_{ij} \le x^k_{ij}, \quad \forall (i,j) \in A,\ \forall k \in K,\ \forall r \in R \tag{24}$$

The sub-tour elimination constraints formulated in Constraints (25)–(27) represent a polynomial-size formulation that is known to yield tight bounds:

$$x_{ij}^{k} \le y_{ij}^{k}, \quad \forall (i,j) \in A,\ \forall k \in K \tag{25}$$

$$y_{ij}^{k} + y_{ji}^{k} = 1, \quad \forall (i,j) \in A,\ \forall k \in K \tag{26}$$

$$y_{ij}^{k} + y_{jp}^{k} + y_{pi}^{k} \le 2, \quad \forall i,j,p \in N,\ \forall k \in K \tag{27}$$

The capacity constraints (28) ensure that the total quantity of requests q_r served by a truck k on arc $(i,j) \in A$ does not exceed the capacity of the truck u_k:

$$\sum_{r \in R} q_r z_{ij}^{kr} \le u_k x_{ij}^{k}, \quad \forall (i,j) \in A,\ \forall k \in K \tag{28}$$

Constraints (29) ensures that the service initiation time t'^{kr}_{i} at node i for request r must not be earlier than the request's arrival time t_i^{kr}:

$$t_i^{kr} \le t'^{kr}_{i}, \quad \forall i \in N,\ \forall k \in K,\ \forall r \in R \tag{29}$$

Constraints (30) require that the service initiation time $t'^{kr}_{s_r}$ at the pickup node s_r for request r must not be earlier than the earliest allowable time a_{s_r} if truck k uses arc (i,j) and Constraints (31) require that the service completion time $\bar{t}_{s_r}^{kr}$ at the pickup node s_r for request r must not be later than the latest allowable time b_{s_r} if truck k uses arc (i,j). Constraints (32) enforce time synchronization for maintenance and battery replacement operations at multi-task node i. Specifically, if vehicle l arrives before vehicle k departs, vehicle l must wait until vehicle k completes its unloading process before it can proceed:

$$t'^{kr}_{s_r} \ge a_{s_r} z_{ij}^{kr}, \quad \forall (i,j) \in A,\ \forall k \in K,\ \forall r \in R \tag{30}$$

$$\bar{t}_{s_r}^{kr} \le b_{s_r}\left(z_{ij}^{kr} + M(1 - z_{ij}^{kr})\right), \quad \forall (i,j) \in A,\ \forall k \in K,\ \forall r \in R \tag{31}$$

$$\bar{t}_{i}^{kr} - t'^{lr}_{i} \leqslant M(1 - m_{ir}^{kl}) \quad \forall r \in R_b^f \cup R_m^f \cup R_m^l,\ \forall i \in T,\ \forall k,l \in K,\ k \ne l \tag{32}$$

Constraints (33) and (34) define the delay t_r^{delay} for request r as the positive difference between the service completion time of request r at delivery node $\bar{t}_{d_r}^{kr}$ and the latest delivery time b_{d_r}:

$$t_r^{\text{delay}} \ge \left(\bar{t}_{d_r}^{kr} - b_{d_r}\right) \sum_{i \in N} z_{id_r}^{kr}, \quad \forall k \in K,\ \forall r \in R \tag{33}$$

$$t_r^{\text{delay}} \ge 0, \quad \forall r \in R \tag{34}$$

Constraints (35) mean that, for each truck $k \in K$, the waiting time t_{ki}^{wait} must be at least equal to the difference between the service initiation time t'^{k}_{i} and the arrival time t_i^k. In other words, the waiting time represents the time interval between the truck's arrival and the service initiation time.

$$t_{ki}^{\text{wait}} \ge t'^{k}_{i} - t_i^{k}, \quad \forall i \in N,\ \forall k \in K \tag{35}$$

Constraints (36) ensure that if request r is served by truck k at node i ($\sum_{j \in N} z_{ij}^{kr} = 1$), the service completion time $\bar{t}_i^{kr}$ is not earlier than the service initiation time t'^{kr}_{i} plus the service duration time s_i^{kr}.

$$t'^{kr}_{i} + s_i^{kr} \sum_{j \in N} z_{ij}^{kr} \le \bar{t}_i^{kr}, \quad \forall i \in N, \forall k \in K, \forall r \in R \tag{36}$$

Constraints (37) ensure that the departure time of truck k from node i ($\bar{t}_i^k$) is no earlier than the completion time of any request r performed by the truck at that node ($\bar{t}_i^{kr}$), thereby guaranteeing that all requests are finished before departure.

$$\bar{t}_i^{k} \ge \bar{t}_i^{kr}, \quad \forall i \in N, \forall k \in K, \forall r \in R \tag{37}$$

Constraints (38) ensure that the arrival time t_i^{kr} of the request r at node i is no earlier than the arrival time t_i^k of truck k at that node, so that services cannot start before the vehicle arrives.

$$t_i^{k} \le t_i^{kr}, \quad \forall i \in N, \forall k \in K, \forall r \in R \tag{38}$$

Constraints (39) ensure that the service initiation time t'^{k}_{i} of truck k at node i is no earlier than the service initiation time of request t'^{kr}_{i} at that node.

$$t'^{k}_{i} \ge t'^{kr}_{i}, \quad \forall i \in N, \forall k \in K, \forall r \in R \tag{39}$$

Constraints (40) and (41) ensure that if truck k travels from node i to node j ($x_{ij}^k = 1$), the arrival time is consistent with the departure time and travel time.

$$\bar{t}_i^k + \tau_{ij}^k - t_j^k \leq M(1 - x_{ij}^k), \quad \forall (i,j) \in A, \forall k \in K \tag{40}$$

$$\bar{t}_i^k + \tau_{ij}^k - t_j^k \geq -M(1 - x_{ij}^k), \quad \forall (i,j) \in A, \forall k \in K \tag{41}$$

5. ALNS heuristic algorithm

The proposed model is solved using a solution framework based on the Adaptive Large Neighborhood Search (ALNS) algorithm, incorporating specific strategies to effectively enhance solution feasibility and efficiency. Unlike traditional Large Neighborhood Search, ALNS possesses self-regulating capabilities in operator selection, exhibiting stronger stability and convergence (Mara et al., 2022).

Algorithm 1 illustrates the ALNS framework tailored specifically for the MPMSP. It embeds the problem's characteristics, including multi-task operations, multi-commodity dynamics, priority-driven service, and node-specific feasibility, directly into the search logic, constraint checks, and acceptance criteria.

Algorithm 1: ALNS algorithm designed for MPMSP.

Input: $K, R, N, A, MaxIterations, T_{\text{temp}}, c, X_{\text{temp}}, pl_r$;
Output: X_{best};
$[K, R, N, A] \leftarrow$ Preprocessing(K, R, N, A);
Initialize set of removal operators $\mathcal{O}_{\text{Removal}} = \{\mathcal{O}_{\text{Worst}}, \mathcal{O}_{\text{Random}}, \mathcal{O}_{\text{Related}}, \mathcal{O}_{\text{History}}, \mathcal{O}_{\text{Route}}\}$ and insertion operators $\mathcal{O}_{\text{Insertion}} = \{\mathcal{O}_{\text{Greedy}}, \mathcal{O}_{\text{Random}}, \mathcal{O}_{\text{Regret}}\}$;
Initialize request pool R_{pool} containing unserved requests;
Construct initial solution X_{initial}, prioritizing requests with higher pl_r first; Ensure maintenance and battery replacement at multi-task nodes T for malfunctioning and low-battery requests $\{r_b^f, r_m^f, r_m^l\}$;
Set initial temperature $T_{\text{initial}} \leftarrow T_{\text{temp}}$ $(T_{\text{temp}} > 0)$ based on X_{initial};
Set $X_{\text{last}} \leftarrow X_{\text{initial}}$, $X_{\text{best}} \leftarrow X_{\text{last}}$;
for *iteration = 1* ***to*** *MaxIterations* **do**
    $[X_{\text{temp}}, R_{\text{pool}}] \leftarrow \mathcal{O}_{\text{Removal}}(X_{\text{temp}}, R_{\text{pool}})$;
    while $R_{pool} \neq \emptyset$ **do**
        $[X_{\text{temp}}, R_{\text{pool}}] \leftarrow \mathcal{O}_{\text{Insertion}}(X_{\text{temp}}, R_{\text{pool}})$;
    end
    if X_{temp} *violates multi-task or multi-commodity related constraints* (15)–(19), (22), (23), *or* (32) **then**
        continue;
    end
    Prioritize computing upper-level objective $F(X_{\text{temp}})$; then compute lower-level operational cost objective $F'(X_{\text{temp}})$
    if $F(X_{temp}) > F(X_{previous})$ **then**
        $X_{\text{previous}} \leftarrow X_{\text{temp}}$;
    else
        if $F(X_{temp}) = F(X_{previous})$ ***and*** $F'(X_{temp}) < F'(X_{previous})$ **then**
            $X_{\text{previous}} \leftarrow X_{\text{temp}}$;
        else
            $\Delta \leftarrow F'(X_{\text{temp}}) - F'(X_{\text{previous}})$;
            $X_{\text{previous}} \leftarrow X_{\text{temp}}$ with probability $p = \exp\left(-\frac{\Delta}{T_{\text{temp}}}\right)$;
        end
    end
    if $F(X_{previous}) > F(X_{best})$ **then**
        $X_{\text{best}} \leftarrow X_{\text{previous}}$;
    else
        if $F(X_{previous}) = F(X_{best})$ ***and*** $F'(X_{previous}) < F'(X_{best})$ **then**
            $X_{\text{best}} \leftarrow X_{\text{previous}}$;
        end
    end
    $T_{\text{temp}} \leftarrow c \times T_{\text{temp}}$
end

The procedure begins with a preprocessing step to align the input data (line 3) and initializes a suite of problem-specific removal and insertion operators (line 4). An initial solution is then constructed using a priority-first principle, ensuring that high-urgency requests are served early and that malfunctioning or low-battery vehicles are correctly routed through multi-task nodes for maintenance or battery replacement (line 6).

Within the main iterative loop (lines 9–36), the algorithm first applies a selected removal operator to extract requests from the current solution into a request pool (line 10). Subsequently, insertion operators iteratively reassign these requests into partial solutions to restore feasibility and enhance solution quality (lines 11–13). Following reconstruction, the solution is rigorously evaluated against MPMSP-specific constraints, including those governing multi-task and multi-commodity operations (line 14). Infeasible solutions are discarded immediately to preserve computational efficiency (line 15).

For feasible solutions, the upper-level priority objective is evaluated first to maximize urgency-aware service (line 17). The lower-level operational cost is considered only if the priority objective does not improve, strictly adhering to the hierarchical optimization structure of the MPMSP. The acceptance mechanism employs a lexicographic rule: improvements in the priority objective are accepted. If the priority objective remains unchanged, cost reductions are evaluated. If neither improves, a simulated annealing criterion probabilistically accepts inferior solutions based on the cost differential, promoting search diversification and helping the algorithm escape local optima (lines 18–27).

Finally, the incumbent solution is updated using this same lexicographic logic (lines 28–34) to ensure that schedules with higher priority service levels are consistently favored. The temperature is then reduced according to a cooling rate (line 35), and upon termination, the algorithm returns the best-found transport plan.

To address the MPMSP in micro-mobility systems, this study designs a customized set of removal and insertion operators. Previous studies have shown that such operator customization alone has been recognized as a significant contribution (Schiffer and Walther, 2018; Gschwind and Drexl, 2019; François et al., 2019; Santos and de Carvalho, 2021; Chen et al., 2021; Voigt, 2025). As shown in Fig. 5, the removal phase strategically extracts specific requests from the current schedule to release resources and disrupt inefficient multi-task structures, such as the suboptimal bundling of maintenance/battery requests or delays to high-priority tasks. This process enhances solution diversity and promotes broader search space exploration.

The Worst Removal operator is designed to improve solution quality in the MPMSP by eliminating requests that contribute the highest marginal cost within a route. As shown in Fig. 5(a), truck k departs from supply node s_1, initially carries $r_1 \in R_b^n$ to node d_1. After replenishing the shortage of r_1 at d_1, the truck then picks up $r_2 \in R_b^f$ and $r_3 \in R_m^f$ from d_1 and transports them to node t_1 for repair. Once the repairs are completed, k delivers r_2 to node d_3, followed by transporting r_3 to node d_4, and finally delivers $r_4 \in R_b^n$ to node d_2, before returning to s_1. In this route, request r_2 at d_3 incurs the greatest cost due to the long detour distance; therefore, r_2 is removed to disrupt the costly structure and create opportunities for generating more efficient routing solutions.

The Random Removal operator is designed to diversify the search process in the MPMSP by randomly eliminating requests from existing routes. As shown in Fig. 5(b), truck k departs from node s_1, initially carrying high-priority request $r_1 \in R_b^n$ to node d_1. After that, the truck then picks up $r_2 \in R_m^f$, $r_3 \in R_m^l$ and $r_4 \in R_b^f$ from d_1 and transports them to node t_1 for repair and battery-replacement. Once completed at t_1, the truck then delivers r_4 to node d_2 and then r_2 and r_3 to node d_3, finally k delivers $r_5 \in R_b^n$ to node d_4, before returning to s_1. In this case, the operator may randomly select a request, such as r_5 at d_4, and remove it from the route, thereby introducing stochasticity that helps the algorithm escape local optima and explore alternative routing structures.

The Related Removal operator is designed to enhance the exploration ability of the MPMSP by simultaneously removing a set of highly similar nodes. As shown in Fig. 5(c), truck k departs from node s_1, carrying $r_1 \in R_m^n$ to node d_1. After that, k picks up $r_2, r_3 \in R_b^n$ from d_1 and transports them to nodes d_2 and d_3, respectively. The truck then collects $r_4 \in R_m^f$, $r_5 \in R_m^l$, and $r_6 \in R_b^f$ from d_3 and takes them to node t_1 for repair and battery replacement. Finally, k delivers r_4, r_5, and r_6 to node d_4, and after replenishing the shortage of r_4 and r_5 at d_4, the truck carries r_6 back to s_1. The operator first randomly selects a request r_2 for removal. Since request r_3 is geographically close to r_2 and shares similar attributes such as quantity, priority level, commodity type, task category, and time window, these two requests exhibit a high degree of similarity. Consequently, both r_2 and r_3 are removed together, thereby disrupting locally correlated structures and enabling the algorithm to explore alternative scheduling and routing configurations.

The History Removal operator is designed to improve solution quality in the MPMSP by leveraging historical information to remove requests placed in suboptimal positions. Each request r records its lowest insertion cost c_r^{lowest} during past iterations, and the gap $\Delta c_r = c_r^{current} - c_r^{lowest}$ is used to identify poorly positioned requests. As shown in Fig. 5(d), a truck k departs from node s_1, carrying $r_1 \in R_b^n$ to node d_1. After that, k picks up $r_2 \in R_m^n$ at d_1 and transports it to node d_2. The truck then collects $r_3 \in R_m^f$ and $r_4 \in R_b^n$ from d_2 and delivers it to node d_5. After replenishing the shortage of r_4 at d_5, k picks up $r_5 \in R_b^f$ and takes r_3 and r_5 to node t_1 for repair. Once completed, k delivers r_3 and r_5 to node d_4 before returning to s_1. Historical data indicate that the request r_4 at d_5 is suboptimal, as its current cost at d_5 is higher than the historical lowest cost achieved at node d_3; therefore, r_4 at d_5 is removed to allow reinsertion at a lower-cost position and enable the construction of improved routing solutions.

The Route Removal operator is applied in the MPMSP when local perturbations are insufficient to improve the solution. Routes with lower capacity utilization are more likely to be cleared, and in extreme cases, all routes may be eliminated and all associated requests returned to the request pool. As shown in Fig. 5(e), truck k departs from node s_1, serves demand nodes d_1, t_2, d_2, and then returns to s_1. If this route is identified as underutilized, the operator removes the entire path, discarding all current served requests and returning them to the request pool R_{pool}. This allows the algorithm to construct alternative routes, such as reallocating other unserved requests to d_3, d_4, and t_2, thereby improving vehicle utilization and enabling more efficient solutions.

As shown in Fig. 6, the repair phase focuses on reinserting the removed requests into scheduling routes in a reasonable manner to construct high-quality feasible solutions.

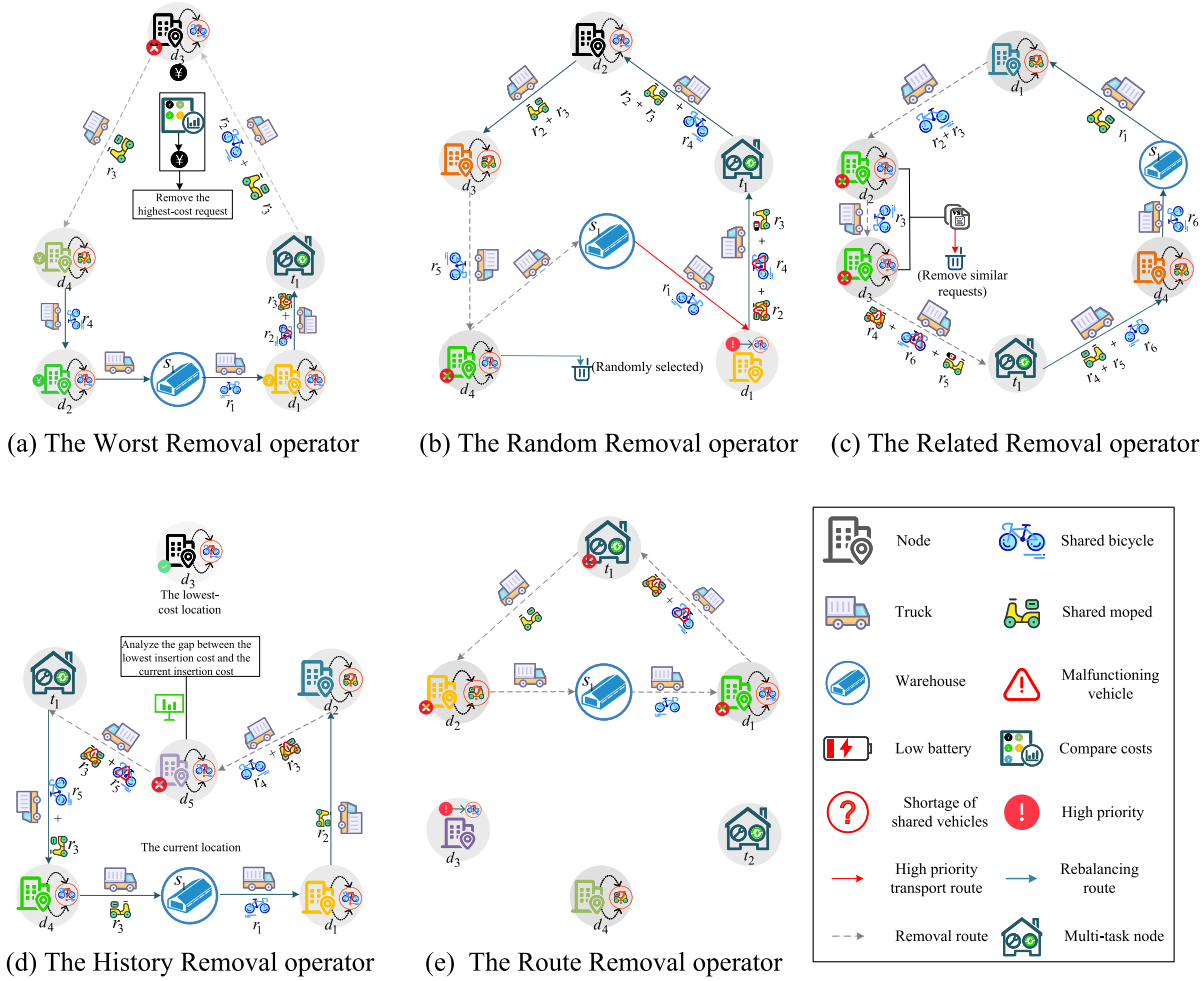


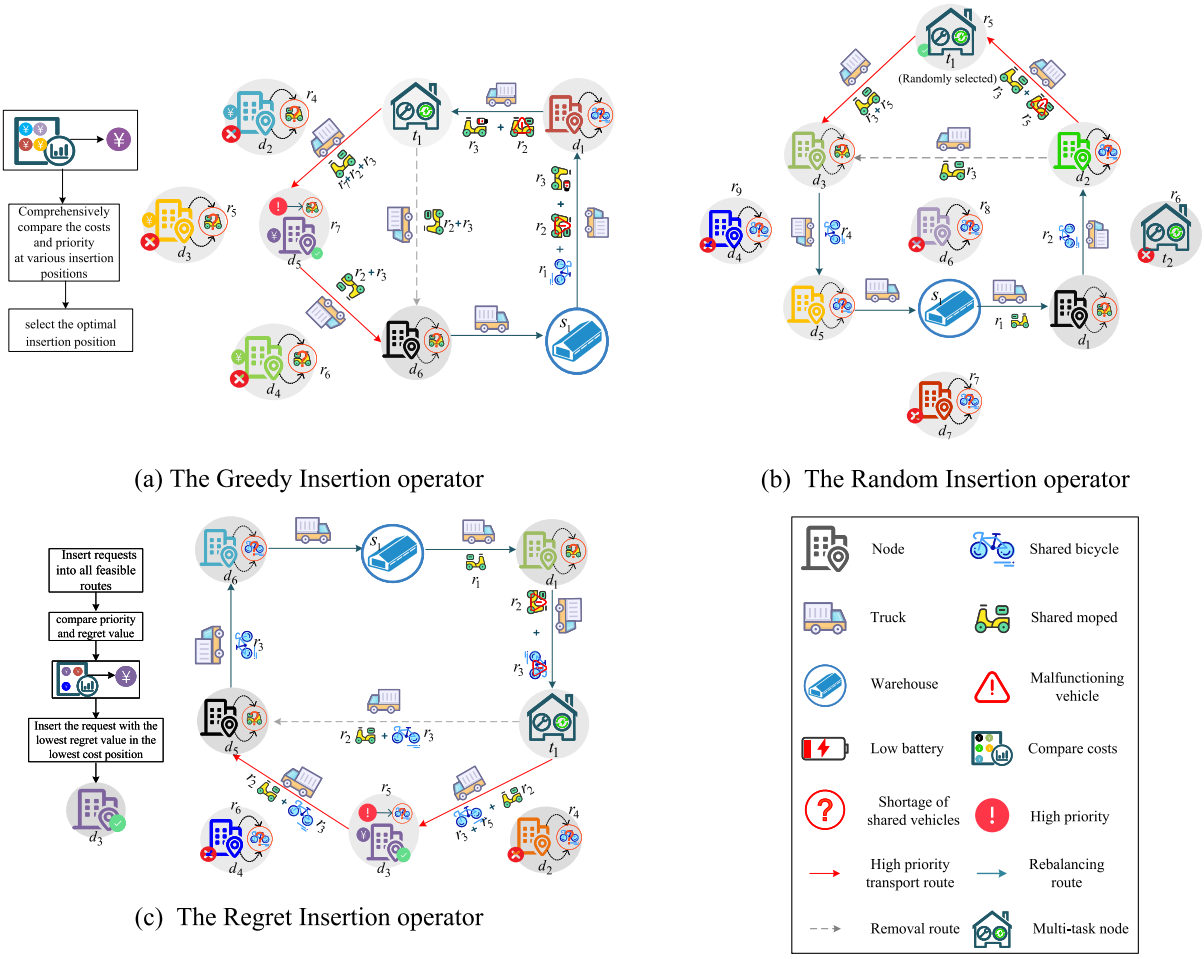
Fig. 5. Removal operators customized to MPMSP.

The Greedy Insertion operator restores tasks by selecting the insertion position that offers the highest priority level and the lowest cost. In Fig. 6(a), truck k departs from supply node s_1, initially carrying $r_1 \in R_b^n$, $r_2 \in R_m^f$, and $r_3 \in R_m^l$ to node d_1. After replenishing the shortage of r_1 at d_1, the truck then transports r_2 and r_3 to node t_1 for repair and battery replacement. Once the repairs are completed, the truck then delivers r_2 and r_3 to node d_6 before returning to s_1. Now, suppose new requests $r_4, r_5, r_6, r_7 \in R_m^n$ arise at nodes d_2, d_3, d_4, and d_5, respectively, and need to be inserted into the route. Among these candidates, r_7 at d_5 has the highest priority value, and inserting it results in the smallest increase in total operational cost. Therefore, r_7 is inserted first, while r_4, r_5, and r_6 are discarded due to their lower priority levels and higher insertion costs. This ensures that only the urgent and cost-efficient tasks are more likely to be restored, thereby improving the overall solution quality in the MPMSP.

The Random Insertion operator randomly selects requests and insertion positions while ensuring feasibility. In Fig. 6(b), truck k departs from node s_1, initially carrying $r_1 \in R_m^n$ to node d_1. After that, the truck delivers $r_2 \in R_b^n$ to node d_2. Following the replenishment of the shortage of r_2 at d_2, the truck picks up $r_3 \in R_m^n$ and transports it to node d_3 for rebalancing. It then proceeds to transport $r_4 \in R_b^n$ to node d_1 before returning to the starting node s_1. Suppose new requests r_5, r_6, r_7, r_8, and r_9 at nodes t_1, t_2, d_4, d_6, and d_7 need to be inserted. In this case, the operator randomly selects $r_5 \in R_m^f$ at t_1 for insertion into the route, thereby introducing stochasticity into the repair process and enhancing search diversity in the MPMSP.

The Regret Insertion operator inserts requests into routes based on regret values. In Fig. 6(c), truck k departs from supply node s_1, carrying $r_1 \in R_m^n$ to node d_1. After replenishing the shortage of r_1 at d_1, the truck then picks up $r_2 \in R_m^f$ and $r_3 \in R_b^f$ and transports them to node t_1 for repair. Once the repairs are completed, k then delivers r_2 to node d_5 and r_3 to node d_6, before returning to s_1. Three new requests, r_4, r_5 and r_6 (at d_2, d_3 and d_4, respectively), need to be inserted into the route. The operator first enumerates all feasible insertion positions for r_4 along the route and records the lowest insertion cost ΔF^4_{best}, the second-lowest insertion cost $\Delta F^4_{\text{second best}}$, and the priority value of the request pl_4. Similarly, it computes ΔF^5_{best}, $\Delta F^5_{\text{second best}}$ and pl_5 for r_5 and ΔF^6_{best}, $\Delta F^6_{\text{second best}}$ and pl_6 for r_6. The regret value for each request is then calculated using the formula:

$$c_r = \alpha \cdot pl_r + \beta \cdot [\Delta F'^r_{\text{second best}} - \Delta F'^r_{\text{best}}] \tag{42}$$



(a) The Greedy Insertion operator

(b) The Random Insertion operator

(c) The Regret Insertion operator

Fig. 6. Insertion operators customized to MPMSP.

This value reflects the additional cost that may be incurred if request r is not inserted into its optimal position. By comparing the regret values, if $c_4 < c_6 < c_5$, it indicates that r_5 has a higher insertion priority (as delaying its insertion may lead to a greater delay and cost increase). Therefore, r_5 at d_3 is prioritized and inserted at its optimal position corresponding to ΔF^5_{best}, resulting in the updated route sequence: $s_1 \rightarrow d_1 \rightarrow t_1 \rightarrow d_3 \rightarrow d_5 \rightarrow d_6 \rightarrow s_1$. This operator prioritizes requests with the highest opportunity cost by quantifying the potential penalty of delayed insertion, thereby achieving a balance between local optimality and global foresight in dynamic routing decisions.

Through the coordinated use of these operators, the algorithm efficiently destroys and repairs routes, reallocates tasks, and adapts vehicle schedules, thereby ensuring solution quality and search diversification in the MPMSP.

6. Numerical experiments

A university district in Chengdu, China, is selected for this case study due to its high reliance on shared micro-mobility and the frequent occurrence of low-battery vehicles and mechanical failures, which collectively diminish service reliability and user satisfaction. As shown in Fig. 7, the transport network includes 12 nodes, with the geographic locations and transport routes explicitly depicted.

In this network, each node has the potential to function as a supply node (s) or a demand node (d). The role of a node is unique within a single transport instance but can vary across different instances. In addition, the number and identity of multi-task demand nodes T are not fixed but randomly generated in different experimental instances. This design avoids overfitting to a specific network configuration and enhances the generalizability and applicability of our model. All requests are delivered through this network by homogeneous fleets of trucks. Each truck can visit multiple nodes, and each node may also be served sequentially by multiple trucks. The main operational parameters are as follows: the load capacity of each truck u_k is 35 tons, the average travel speed v_k is 45 km/h, the unit transport cost $c^1_k/c^{1'}_k$ is 5.4 CNY per hour per ton/0.7 CNY per kilometer per ton, the unit loading/unloading cost c^2_k is 2 CNY per ton, the unit storage cost c^3_k is 1 CNY per hour per ton, the unit waiting cost c^4_k is 1 CNY per hour (Chen et al., 2023; Guo et al.,

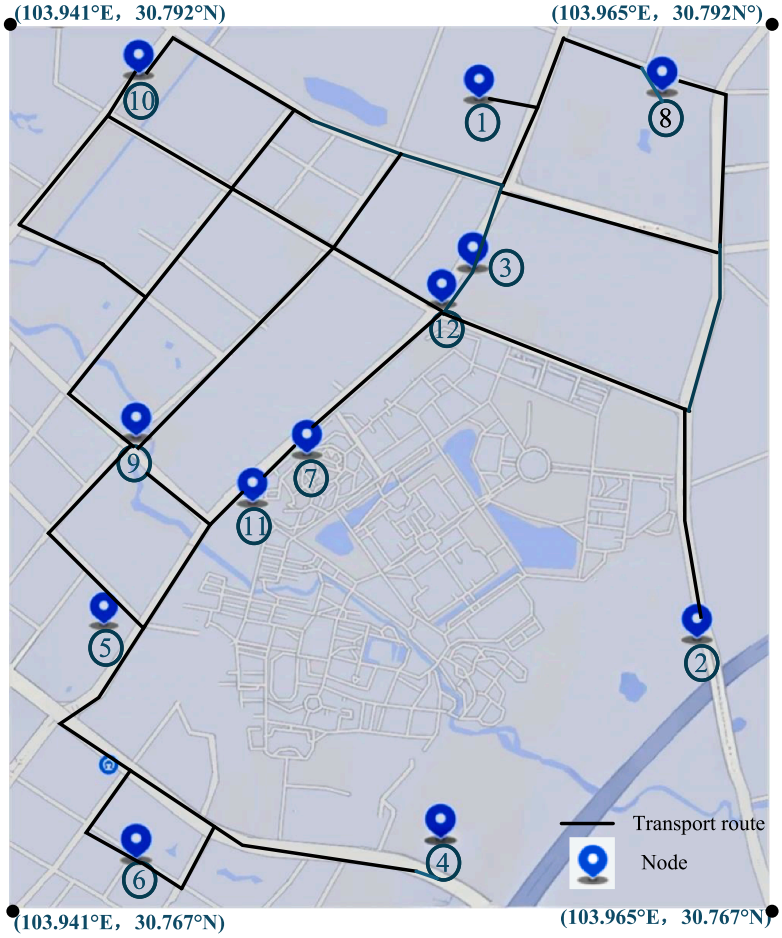


Fig. 7. The geographic location and transport routes of the nodes.

2024). The experiments are implemented in Python 3.9 and executed on a Windows 11 system with 16 GB RAM and a 12th Gen Intel(R) Core(TM) i7-12700H processor running at 2.30 GHz.

Section 6.1 examines the design of cost objective function and the performance of the tailored ALNS algorithm. Section 6.2 compares the single-task and multi-task scheduling mechanisms. Section 6.3 investigates the impact of different high-priority request ratios to identify the optimal allocation strategy. Section 6.4 tests the performance of the proposed model under concentrated or dispersed inventory and demand instances.

6.1. The design and performance of ALNS algorithm

To comprehensively evaluate the proposed ALNS algorithm, a series of analyses are conducted. First, Section 6.1.1 investigates the impact of different sub-objective function designs. Following this, Section 6.1.2 validates the algorithm's performance by benchmarking its solutions against results from an exact method (Gurobi). Finally, Section 6.1.3 evaluates individual removal and insertion operators to identify their unique contributions and synergistic effects within the algorithm.

6.1.1. The design of cost objective function

To analyze the rationality of each sub-cost in the objective function and their impact on the total cost, we categorize the five cost components into three conceptually distinct groups according to their spatial, operational, and temporal characteristics: F_1: direct transport-related cost, which is incurred continuously along the arcs of the road network during vehicle movement (arc-based cost); F_2, F_3, and F_4: service node-based handling costs, which arise from physical operations and time spent at service nodes, encompassing loading/unloading (F_2), storage (F_3), and waiting costs (F_4); F_5: time-dependent delay penalty, which reflect the cost of violating specified deadlines at demand nodes. Separating the delay penalties (F_5) from the node-based handling costs (F_2, F_3, F_4) allows the model to explicitly control the trade-off between improving internal operational efficiency and satisfying customer time requirements.

Table 3
Impact of different sub-costs.

Instances	Weights			F_1	$F_2 + F_3 + F_4$	F_5	F'	Gap
	W_1	W_2	W_3	(CNY)	(CNY)	(CNY)	(CNY)	(%)
Instance 1	1	0	0	**8664***	2850	3236	14,750	14.18
Instance 2	0	1	0	13,138	**2608***	2971	18,718	44.90
Instance 3	0	0	1	11,326	2850	**312***	14,488	12.15
Instance 4	0.2	0.7	0.1	10,793	2693	2726	16,212	25.50
Instance 5	0.2	0.1	0.7	10,855	2850	665	14,370	11.24
Instance 6	0.7	0.1	0.2	8896	2850	1218	12,964	0.36
Instance 7	0	0.5	0.5	12,325	2697	774	15,796	22.28
Instance 8	0.4	0.4	0.2	9097	2700	1160	12,957	0.30
Instance 9	0.333	0.333	0.334	9170	2704	1043	**12918***	–

F_1: transport cost; F_2: loading/unloading cost; F_3: storage cost; F_4: waiting cost; F_5: delay penalty; F': total cost.

Each category is assigned a weight W_1, W_2, and W_3, respectively. The weighted cost equation can be formulated as follows:

$$F_C = W_1 \times F_1 + W_2 \times (F_2 + F_3 + F_4) + W_3 \times F_5 \tag{43}$$

To evaluate the influence of each sub-cost on the total cost objective, various comparative instances with different weight configurations were designed: Instance 1: $W_1 = 1, W_2 = 0, W_3 = 0$ (transport cost only); Instance 2: $W_1 = 0, W_2 = 1, W_3 = 0$ (service node-based handling costs only); Instance 3: $W_1 = 0, W_2 = 0, W_3 = 1$ (delay penalty only); Instance 4: $W_1 = 0.2, W_2 = 0.7, W_3 = 0.1$; Instance 5: $W_1 = 0.2, W_2 = 0.1, W_3 = 0.7$; Instance 6: $W_1 = 0.7, W_2 = 0.1, W_3 = 0.2$; Instance 7: $W_1 = 0, W_2 = 0.5, W_3 = 0.5$; Instance 8: $W_1 = 0.4, W_2 = 0.4, W_3 = 0.2$; Instance 9: $W_1 = 0.333, W_2 = 0.333, W_3 = 0.334$ (equal weights). To highlight the contrast under different weight configurations, experiments are conducted under an instance with 30 requests. The corresponding test results are presented in Table 3. Each sub-cost F_i is always calculated based on the actual routing and scheduling decisions. A weight of zero merely removes that sub-cost term from the optimization objective, and the ALNS algorithm does not attempt to minimize it. The reported total cost F' in Table 3 is the sum of all sub-costs, not the weighted cost equation value F_C.

Experimental results demonstrate that when the optimization process strictly prioritizes a single cost category, the resulting solutions exhibit significant operational imbalances and fail to reach a global optimum. For example, Instance 1 (focusing solely on transport cost) generated the lowest transport cost-ranging from 76% to 80% of other instances, but incurred a delay penalty 10.4 times higher than that of Instance 3. Similarly, Instance 2 (minimizing service node handling costs) reduced handling expenses to between 91% and 98% of other instances, yet inflated transport costs to 152% of Instance 1 and delay penalties to 9.5 times that of Instance 3. Instance 3 strictly prioritized delay penalties, successfully suppressing them to only CNY 312 (24% to 47% of other instances), but consequently drove transport costs up to 131% of Instance 1.

The composite weight configurations further expose the inherent trade-offs among these cost components. Instance 4, which heavily weights handling costs, maintains low node-based expenses but yields a transport cost 25% higher than Instance 1, resulting in the highest total cost among the composite instances. Conversely, prioritizing delays in Instance 5 successfully mitigates penalties but produces suboptimal routing and handling metrics. Instances 6 and 8, which heavily favor transport efficiency while moderately penalizing delays, yield total costs very close to the global optimum, highlighting the impact of transport cost on the model's overall objective.

Ultimately, the balanced configuration evaluated in Instance 9, assigning equal weights to all three cost categories, achieves the minimum global total cost. Although none of the individual components reach their isolated optimums, this solution effectively harmonizes the competing factors, avoiding the severe financial imbalances of the extreme weighting schemes. The total cost of Instance 9 represents only 88%, 69%, and 89% of the costs in Instances 1, 2, and 3, respectively, and remains strictly lower than Instances 4, 5, and 7. These findings show that a balanced objective function is necessary to derive transport plans that are both operationally superior and economically efficient.

These results indicate that the transport optimization process constitutes a complex system where minimizing individual costs does not necessarily lead to the global optimum, thereby validating the necessity and rationality of incorporating multiple sub-costs in the objective function. Furthermore, the request service rate remains consistently stable across all four instances, demonstrating the stability of the proposed model.

6.1.2. Comparison with the exact approach

Considering that ALNS can only obtain near-optimal solutions, to demonstrate the quality of the solutions achieved by the proposed ALNS algorithm in this paper, this section compares the performance of the commercial solver (Gurobi) and the ALNS algorithm. The computation time for both methods is limited in 1 h. For 10 instance sizes ranging from 1 to 10 requests (using a fleet of 34 trucks), Table 4 summarizes the numbers of decision variables and constraints, as well as the optimal solutions and computation times for both approaches. To ensure result reliability, each instance is tested three times, and average total costs and computation time are reported.

Across all test instances, ALNS consistently produces the same optimal solutions as Gurobi, achieving a zero optimality gap while requiring only a fraction of the computation time. Specifically, ALNS completes all cases using merely 0.02% to 0.51% of the

Table 4
Comparison with gurobi.

Problem Complexity			AVG. Total Cost (CNY)		Gap	AVG. Best Time (s)		Ratio
$\|O\|$	Variable	Constraint	Exact approach	ALNS	(%)	Exact approach	ALNS	(%)
1	19024	70814	153.67	153.67	0	44.94	0.01	0.02
2	25910	91989	518.89	518.89	0	88.92	0.02	0.02
3	38074	135636	865.79	865.79	0	167.76	0.67	0.40
4	45318	158344	1246.17	1246.17	0	270.91	0.72	0.27
5	53344	184284	1415.29	1415.29	0	406.74	1.29	0.32
6	59540	202140	1568.95	1568.95	0	537.57	2.39	0.44
7	72124	247814	2805.62	2805.62	0	731.25	2.47	0.34
8	84018	290610	3428.27	3428.27	0	978.39	5.01	0.51
9	96872	337133	4067.97	4067.97	0	1196.80	5.45	0.45
10	103068	354989	4298.47	4298.47	0	1397.51	7.13	0.51
20	198438	680519	–	8538.51	–	6000*	14.26	–

$|O|$: number of requests
$*$ time limit reached (6000 s). - no feasible solution has been found given the time constraint.

Table 5
Impact of different operator designs on ALNS performance.

Designs	$\|R\|$	Total	Gap (%)	Transport	Load/unload	Storage	Waiting	Delay penalty	Best time (s)
Greedy	5	1956	20.88	1311	510	1	0	133	0.15
Random	5	1932	19.41	1371	510	1	0	50	0.87
Proposed ALNS	5	1618	–	1044	510	2	0	61	0.59
Greedy	10	4972	5.39	4070	870	24	8	0	3.14
Random	10	4737	0.40	3826	870	28	13	0	20.58
Proposed ALNS	10	4718	–	3835	870	4	9	0	24.78
Greedy	20	9802	7.04	7797	1740	7	34	224	26.20
Random	20	11,569	26.29	9766	1770	6	26	0	44.16
Proposed ALNS	20	9158	–	7383	1740	10	25	0	31.22
Greedy	30	18,547	27.67	13,821	2640	8	42	2036	150.61
Random	30	15,136	4.21	9757	2490	12	44	2834	155.77
Proposed ALNS	30	14,524	–	11,786	2640	11	46	41	174.83

$|R|$: number of requests; Gap (%): increase of Greedy/Random over Proposed ALNS for the same request size, computed as Gap (%) $= \frac{\text{Greedy or Random}-\text{Proposed ALNS}}{\text{Proposed ALNS}} \times 100\%$; All costs are measured in CNY.

exact approach's runtime. The computational advantage becomes increasingly evident as problem size grows; for instance, in the 20-request case, Gurobi fails to obtain any feasible solution within the time limit (6000 s), whereas ALNS remains computationally tractable. These results clearly demonstrate that ALNS achieves exact-quality solutions in small-scale instances with dramatically lower computational effort, highlighting its superior efficiency, scalability, and practical applicability in real-world.

6.1.3. The impact of different operator designs

In this study, three insertion operators and five removal operators are tailored for MPMSP. To investigate the impact of different operator designs, we compare our current design with two typical benchmark designs:

(a) Greedy: only greedy operators (the Greedy Insertion operator and Worst Removal operator) are used;

(b) Random: only random operators (the Random Insertion operator and Random Removal operator) are used;

(c) Proposed ALNS: the full set of eight operators is used, including greedy operators, random operators, and MPMSP-specific operators (i.e., the Regret Insertion operator, Related Removal operator, History removal operator, Route Removal operator).

These three designs are tested across four instance sizes, with the number of requests set to 5, 10, 20, and 30, respectively. The results in Table 5 show that across all tested instance sizes, our proposed design consistently achieves the lowest total cost. Greedy operators produce solutions that are on average 15.2% higher in total cost than ours (ranging from 5.39% to 27.67%), while Random exhibits even more unstable performance, with cost gaps varying between near-optimal (0.40%) and substantially worse (26.29%). In terms of computational effort, the data indicate that our design does not incur a significant runtime overhead compared with the two benchmarks. These results demonstrate that the customized operator design not only reduces the objective function of the solution but also improves the stability and scalability of ALNS for MPMSP without sacrificing computational efficiency.

Table 6
Comparison between our proposed ALNS and three benchmark metaheuristics.

	Total operational cost (CNY)							The number of served requests				Computation time (s)			
$\|R\|$	Proposed ALNS	GA	Gap (%)	SA	Gap (%)	TS	Gap (%)	Proposed ALNS	GA	SA	TS	Proposed ALNS	GA	SA	TS
3	847	847	0.00	847	0.00	847	0.00	3	3	3	3	0.18	0.32	0.39	0.74
5	1431	1431	0.00	1431	0.00	1431	0.00	5	5	5	5	0.83	1.26	0.71	1.66
10	4758	4758	0.00	4758	0.00	4758	0.00	10	10	10	10	2.26	2.97	1.57	5.72
15	6147	6161	0.23	6219	1.17	6147	0.00	14	14	14	14	13.95	16.16	8.75	20.04
20	8780	9260	5.47	9395	7.00	8780	0.00	18	18	18	18	37.43	37.61	24.09	42.75
25	11983	13528	12.89	12938	7.97	12162	1.49	23	23	23	23	72.36	61.19	55.88	95.44
30	13483	14948	10.87	14643	8.60	13869	2.86	27	27	27	27	143.16	191.32	72.82	149.95
35	17289	18778	8.61	18231	5.45	17645	2.06	30	30	30	30	222.17	210.93	109.38	257.07
40	20491	22048	7.60	22048	7.60	20894	1.97	34	34	34	34	319.00	252.63	123.35	360.94

$|R|$: The number of requests; Proposed ALNS: The customized adaptive large neighborhood search proposed in this paper; GA: Genetic algorithm; SA: Simulated annealing algorithm; TS: Tabu search algorithm.

6.1.4. Comparison with benchmark metaheuristic algorithms

To more comprehensively evaluate the performance of the customized ALNS algorithm proposed in this paper, we compare it with three widely used metaheuristic algorithms: the Genetic Algorithm (GA), the Simulated Annealing Algorithm (SA), and the Tabu Search Algorithm (TS). These algorithms are commonly employed as benchmark methods in scheduling and routing research that also adopts ALNS as the solution algorithm (Li et al., 2021; Tang et al., 2025; Wang et al., 2026; Mohri et al., 2020; Masmoudi et al., 2016; Woo et al., 2024; Zandieh et al., 2023; He et al., 2023; Jiang et al., 2026; Kuhn et al., 2021; Martins et al., 2019). All the three benchmark algorithms are subjected to the same termination criterion as the proposed ALNS. The experiments are performed on seven instances ranging from 3 to 40 requests, with an analysis on three performance indicators: total operational cost, the number of served requests, and computation time. The detailed results are presented in Table 6.

As shown in Table 6, the proposed customized ALNS consistently achieves the lowest (or equal) total operational cost among all compared metaheuristics across all instances. For small instances with 3 to 15 requests, all four algorithms produce identical or nearly identical results, with cost gaps of at most 1.17% (SA at $|R| = 15$). However, as the problem scale increases to 20 requests and beyond, the superiority of ALNS becomes clearly evident, with average gaps of approximately 9.29% over GA and 7.32% over SA. The most significant performance advantages are observed at $|R| = 25$ against GA (a 12.89% gap) and at $|R| = 30$ against SA (an 8.60% gap), whereas the optimality gaps compared to TS remain comparatively narrower across all instance sizes. At larger scales (e.g., $|R| = 35$ and $|R| = 40$), ALNS persistently maintains strong improvements. Regarding computational efficiency, although ALNS requires more runtime than the SA heuristic in most cases, the additional computation time remains acceptable in practical settings. The trade-off between marginally longer runtime and significantly improved solution quality is favorable for real-world applications. Overall, the comparison confirms that the customized ALNS constitutes a competitive and efficient solution method for the MPMSP.

6.2. Comparison of single-task scheduling and multi-task scheduling

This section evaluates the proposed multi-task scheduling mechanism by comparing it with the single-task mechanism, serving as a key analysis to demonstrate one of the core innovations of this study. As shown in Fig. 8, the multi-task transport system achieves significant enhancements in the availability and service rate of shared bicycles and mopeds compared to the single-task transport system.

Specifically, traditional single-task scheduling yields availability rates of only 41.2% for shared bicycles and 39.1% for shared mopeds, which are far below acceptable levels for shared mobility operators. In contrast, our proposed multi-task scheduling achieves full availability (100%) for both shared bicycles and mopeds, thereby eliminating the risk of non-operational shared vehicles. The reason is evident: under single-task scheduling, only vehicle rebalancing is considered, neglecting tasks such as bicycle maintenance and moped battery replacement. This often results in malfunctioning vehicles or low-battery mopeds being relocated to demand nodes, greatly reducing availability. By contrast, multi-task scheduling routes defective or low-battery shared vehicles to service nodes for repair or battery replacement, ensuring that only operational shared vehicles are sent to demand nodes, while serviced ones rejoin the system in subsequent cycles.

Meanwhile, compared with single-task scheduling, multi-task scheduling yields substantial improvements in service rate: the rate for shared bicycles increases to 86.8%, and that for mopeds rises to 90.1%, corresponding to relative improvements of 110.7% and 130.4%, respectively. The higher service rate of multi-task scheduling arises from both demand and supply sides. On the demand side, in single-task scheduling, although the same number of shared vehicles are dispatched to demand nodes, many of them are malfunctioning or low-battery and therefore fail to meet actual demand. This mismatch forces additional reallocations, yet the number of truck fleets is limited, leaving part of the demand unsatisfied. On the supply side, multi-task scheduling integrates maintenance and battery replacement into planning, allowing malfunctioning bicycles and low-battery mopeds to be restored and quickly redeployed, resulting in a higher overall service rate.

Figs. 9 and 10 present the node-level comparison of shared bicycles and mopeds service rates under single-task and multi-task scheduling. Since multi-task scheduling consistently maintains 100% shared vehicle availability, further analysis of availability is unnecessary; therefore, the discussion focuses exclusively on service rate.

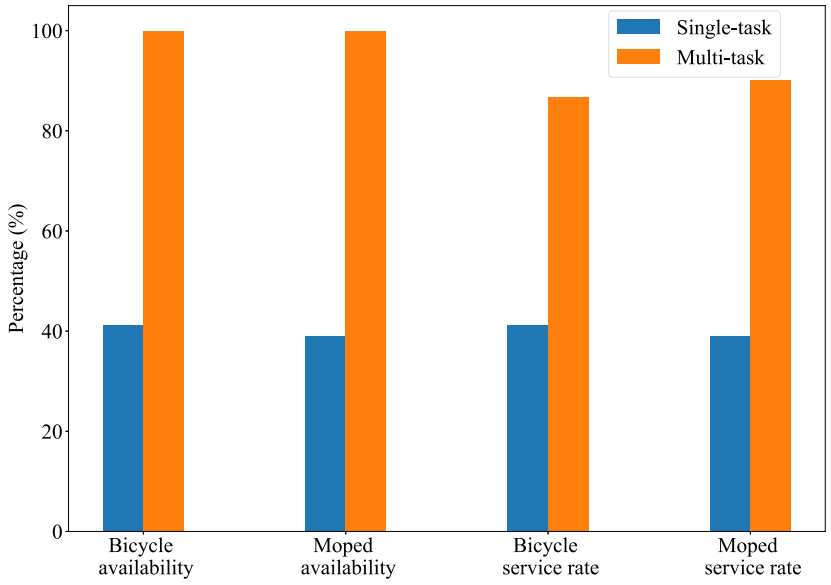


Fig. 8. Overall comparison between single-task and multi-task scheduling.

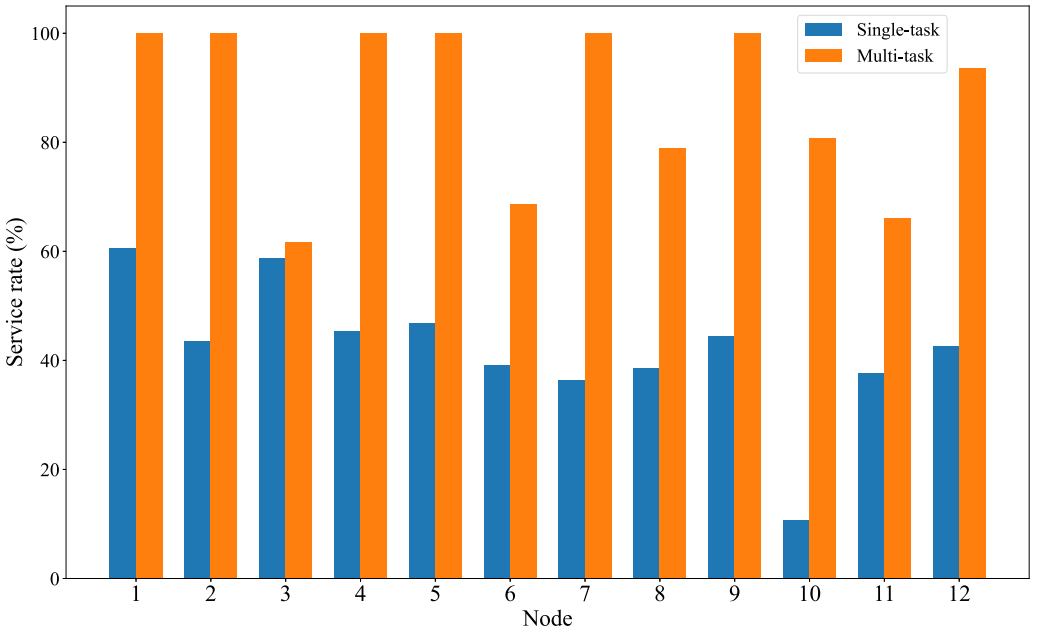


Fig. 9. Comparison of shared bicycle service rate between single-task and multi-task scheduling.

As shown in Fig. 9, under single-task scheduling, bicycles show weak service performance at most nodes. For example, service rates fall to only 10.6% at node 10 and remain below 40% at nodes 7 and 8, while even relatively better-performing nodes such as nodes 1 and 3 stay around 60%. These results suggest that a large share of the fleet remains unavailable to meet demand. With multi-task scheduling, however, the service rate changes dramatically. Several nodes (e.g., nodes 1, 2, 4, 7, and 9) achieve 100% service rates, while others record remarkable improvements, such as node 10 rising from 10.6% to over 80% and node 8 from 38.6% to nearly 79%. The failure to achieve full coverage at certain nodes, such as nodes 3 or 6, is not due to operational inefficiencies but rather to structural imbalances, where user demand surpasses the maximum fleet capacity that the system can provide.

For mopeds, the contrast between the two strategies is even sharper (Fig. 10). Under single-task scheduling, most nodes operate at low to moderate service rates, with typical values around 30–45%; for instance, node 10 records only 29.4% and node 4 approximately 31.9%. In multi-task scheduling, these deficiencies are almost completely resolved: nodes such as nodes 1, 2, 4, 7, and 10 reach 100%

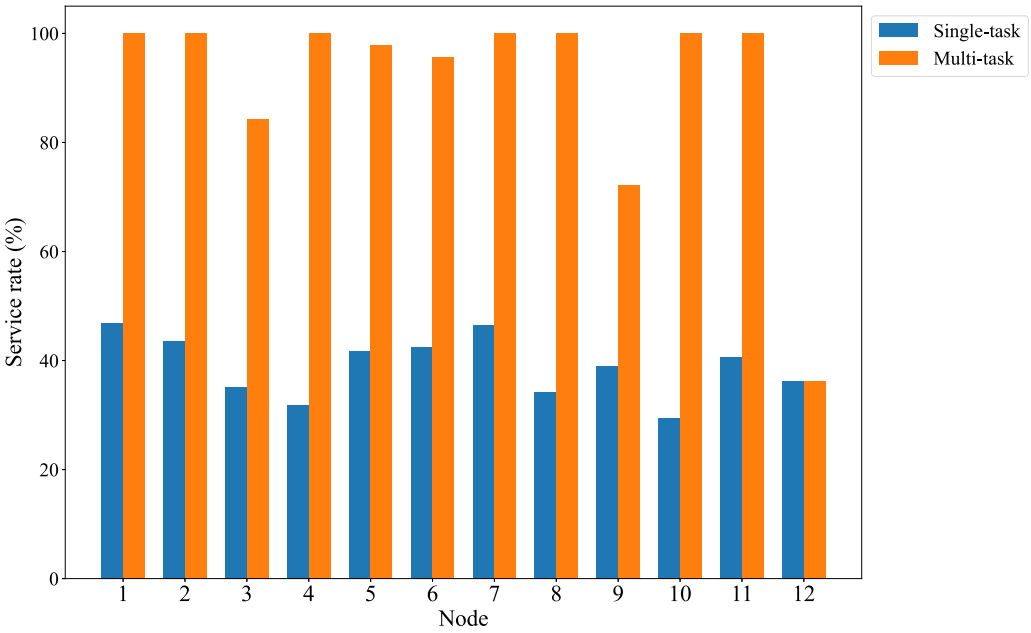


Fig. 10. Comparison of shared moped service rate between single-task and multi-task scheduling.

service rates, while node 6 improves from 42.6% to 95.7% and node 3 from 35.1% to 84.2%. The only notable exception is node 12, where the service rate remains at 36.2%, a result driven by demand levels that exceed the total fleet capacity.

6.3. Performance under different high-priority request ratios

This section investigates the influence of varying high-priority request ratios, providing critical evidence for the rationality and superiority of the proposed priority-based mechanism. In the experimental design, we consider four ratio settings for high-priority requests: 20%, 30%, 40%, and 50%. For each setting, the system is tested under four request scales of 10, 20, 30, and 50 requests. The resulting service rates are illustrated in Fig. 11.

The experimental results show that when the total number of requests is 10, our model ensures a 100% service rate for high-priority requests regardless of their proportion. This is mainly because the total demand is far below the system's capacity threshold, and resources are sufficient. As the number of requests increases, the model can maintain full service rate only when the high-priority request ratio is 20% or 30%; once the proportion exceeds 30%, the service rate drops to 93% or even below 90%. In particular, under the scenario of 50 requests, the service rate reaches its highest level (93%) when the high-priority request ratio is 30%, but falls sharply to 72% when the proportion reaches 50% (Fig. 11(b)). This indicates that a moderate proportion of high-priority requests prompts the model to allocate resources more efficiently, whereas an excessive proportion leads to rapid depletion of limited vehicle capacity, leaving some high-priority requests unserved. Additionally, as the total number of requests grows, the overall service rate of high-priority requests declines, reflecting inherent resource constraints such as limited fleet size and vehicle capacity.

Compared with low-priority requests, high-priority requests consistently show a significant service advantage. When the high-priority request ratio does not exceed 40%, this advantage remains stable across different request scales, demonstrating the model's scalability and stability. The only exception occurs when the ratio reaches 50% with 50 total requests, where their service rate (72%) falls below that of low-priority requests (76%). Meanwhile, the service rate of low-priority requests fluctuates considerably when the high-priority request ratio is 40% or 50%, but remains relatively stable when the ratio is 20% or 30%.

Fig. 12 illustrates the impact of different proportions of high-priority requests on the major sub-costs under varying request sizes, with all sub-costs normalized. The detailed cost data under various request sizes and high-priority ratio instances are summarized in Table 7.

Among all sub-costs, transport cost has the greatest impact on the total cost, consistently accounting for at least 76% and usually exceeding 80% (Table 7). As shown in Fig. 12, transport cost under the 30% high-priority request ratio tends to be lower than most other ratios across various request sizes. The only exceptions occur in the 10-request case, where the cost is 0.7% higher than that under 40%, and in the 50-request case, where the 50% ratio yields an artificially low cost due to a sharp decline in service rate. On average, the transport cost under the 30% ratio is merely 93.6% of that under the 20% ratio and 90.6% of that under the 40% ratio. The gap reaches its maximum at the 30-request scale, where the 30% ratio corresponds to only 84.1% of the cost under 20% and 81.9% under 40%.

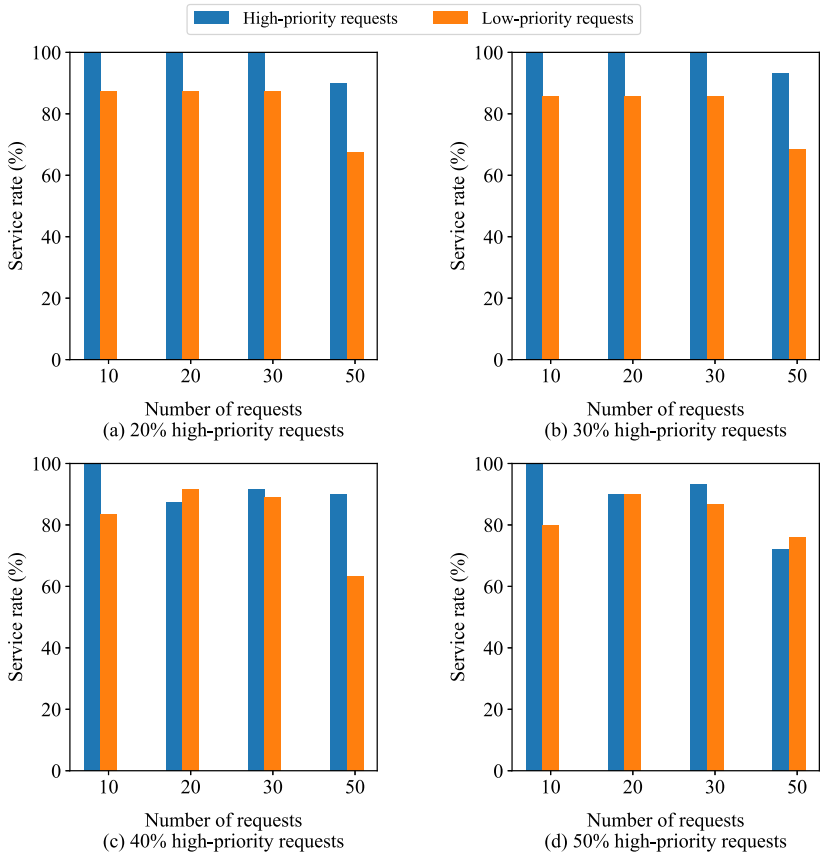


Fig. 11. Request service rates under different high-priority ratio settings.

Given the critical role of timeliness in shared mobility systems, delay penalty is another important sub-cost. The results indicate that delay penalties are absent when the number of requests is small, and remain minor at moderate scales. However, once the request volume reaches higher levels, particularly with 40%–50% high-priority shares, delay penalties rise sharply. This is mainly due to excessive prioritization: to fulfill a disproportionate number of high-priority requests, too many trucks are dispatched simultaneously, and lower-priority requests are inevitably left unfulfilled. Additionally, high-priority requests are assigned larger penalty coefficients, further amplifying the overall penalty. These findings suggest that setting the high-priority ratio at 20%–30% achieves a better balance between service differentiation and timeliness.

Similarly, in small-scale request instances, the 20%–30% high-priority scenarios exhibit higher waiting and storage costs. As the request scale increases, however, the 50% high-priority scenario becomes the most costly in these two categories. The reason is that with fewer requests, a higher share of high-priority demand leads to faster dispatching and thus minimal waiting or storage. In contrast, when the request scale is large, an excessively high share of high-priority requests causes backlogs and accumulation, driving up both costs.

To address the concern that the observed conclusion may be instance- or parameter-dependent, we conducted additional sensitivity analyses on three key operational parameters: vehicle capacity, travel speed, and fleet size. For vehicle capacity and speed, we considered five levels ranging from 60% to 140% of the baseline value. For fleet size, we varied the number of available trucks to 200% and 400% of the baseline level, under three request volumes (5, 10, and 20). All scenarios were evaluated under the four high-priority request ratios. The detailed cost results are presented in Tables 8–10, respectively.

Under the capacity sensitivity analysis (Table 8), the 30% ratio achieves the lowest total cost in four out of five capacity states (C1–C4), while the 20% ratio also performs competitively in several instances (e.g., C5). Similarly, under the speed sensitivity analysis (Table 9), the 30% ratio yields the lowest total cost across all speed states, with the 20% ratio again showing strong performance in multiple scenarios (e.g., V2, V3). These findings suggest that, while the performance of specific ratios is inherently influenced by instance characteristics and system parameters, the competitiveness of the 20%–30% ratio interval emerges as a relatively common pattern across a wide range of experimental settings.

However, Table 10 reveals a clear interaction between fleet size and the optimal high-priority ratio. For the 20-order scenario under the baseline fleet (N1), the 30% ratio yields the lowest total cost . However, when the fleet size is doubled (N2), the 40% ratio becomes the most cost-effective, outperforming both 30%. With a fourfold increase in fleet size (N3), the 50% ratio achieves the

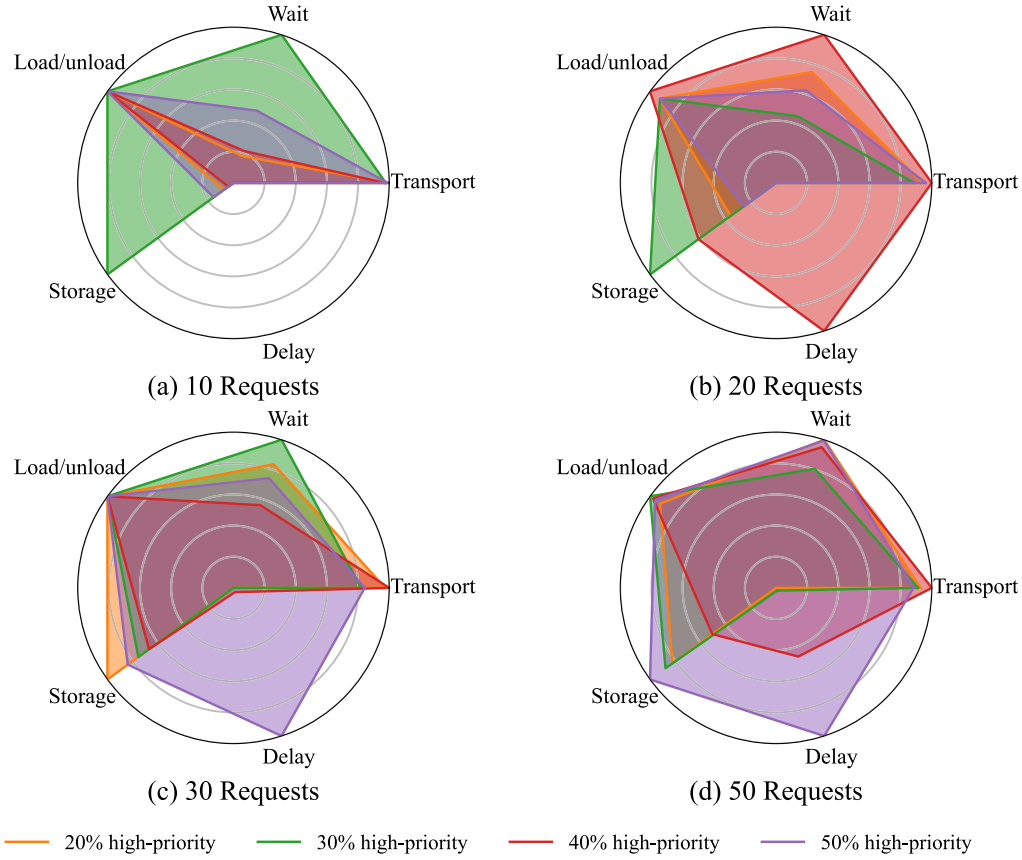


Fig. 12. Radar chart of costs under varying high-priority request ratios.

minimum total cost. This pattern suggests that as more trucks become available, the system can accommodate a higher proportion of high-priority requests without incurring excessive costs. For smaller order volumes (10 and 5 requests), the cost differences across ratios are relatively minor due to the abundance of capacity relative to demand. Nevertheless, similar trends can be observed: under N2 and N3, higher ratios (40% or 50%) consistently lead to lower costs than lower ones, although the variations are less substantial. These findings carry practical implications for enterprise fleet management. In settings where a large share of high-priority requests is anticipated, operators should ensure a sufficiently large fleet to maintain cost efficiency. Conversely, when the fleet size is constrained, it may be advisable to limit the proportion of high-priority requests to avoid excessive costs. In other words, the optimal high-priority ratio should be aligned with the available transport resource.

The findings underscore a key managerial insight: operators need to monitor the relationship between the proportion of high-priority requests and the available transport resources, and respond accordingly, either by expanding the fleet size or by controlling the high-priority request ratio.

6.4. Performance across concentrated or dispersed inventory and demand scenarios

To evaluate the performance of the proposed model, we design four instances: (a) 80% of the demand is concentrated in 20% of the nodes; (b) 50% of the demand is concentrated in 20% of the nodes; (c) 80% of the inventory is concentrated in 20% of the nodes; and (d) 50% of the inventory is concentrated in 20% of the nodes. The results are visualized in Figs. 13–16. We compare the net demand of all 12 nodes before and after scheduling, considering shared bicycles requiring rebalancing, shared mopeds requiring rebalancing, and vehicles requiring maintenance or battery replacement.

As shown in Fig. 13, the model demonstrates excellent performance under highly concentrated demand scenarios. Before scheduling, the absolute net demand for various vehicles is significantly high across multiple nodes. After scheduling, complete supply-demand balance is achieved for shared bicycles and mopeds in 75% of the nodes, and for malfunctioning vehicles requiring maintenance in 92% of the nodes. Moreover, the absolute net demand in unbalanced nodes decreased substantially, with only Node 8 for shared bicycles and Node 2 for shared mopeds exhibiting relatively high residual net demand.

When demand becomes more dispersed, the model performs well (Fig. 14). Following the scheduling optimization, complete balance was achieved across 50% of nodes for shared bicycles, 100% of nodes for shared mopeds, and 83% of nodes for malfunctioning

Table 7
Costs under varying high-priority request ratios.

$\|R\|$	Total cost	Transport cost (%)	Waiting cost (%)	loading/unloading cost (%)	Storage cost (%)	Delay penalty (%)	High-priority ratios (%)
10	4697	3822 (81.4)	3 (0.1)	870 (18.5)	2 (0.0)	0	20
10	4651	3738 (80.3)	16 (0.3)	870 (18.7)	27 (0.6)	0	30
10	4587	3712 (80.9)	4 (0.1)	870 (19.0)	1 (0.0)	0	40
10	4668	3786 (81.1)	8 (0.2)	870 (18.6)	4 (0.1)	0	50
20	9406	7631 (81.1)	30 (0.3)	1740 (18.5)	5 (0.1)	0	20
20	8983	7212 (80.3)	18 (0.2)	1740 (19.4)	13 (0.1)	0	30
20	10383	8144 (78.4)	40 (0.4)	1890 (18.2)	8 (0.1)	301 (2.9)	40
20	9596	7828 (81.6)	25 (0.3)	1740 (18.1)	3 (0.0)	0	50
30	16322	13618 (83.4)	50 (0.3)	2640 (16.2)	14 (0.1)	0	20
30	14161	11451 (80.9)	60 (0.4)	2640 (18.6)	10 (0.1)	0	30
30	16674	13989 (83.9)	33 (0.2)	2640 (15.8)	9 (0.1)	3 (0.0)	40
30	14592	11789 (80.8)	44 (0.3)	2640 (18.1)	12 (0.1)	107 (0.7)	50
50	21298	17725 (83.2)	52 (0.2)	3450 (16.2)	71 (0.3)	0	20
50	21308	17415 (81.7)	41 (0.2)	3750 (17.6)	76 (0.4)	26 (0.1)	30
50	23521	19115 (81.3)	49 (0.2)	3660 (15.6)	44 (0.2)	653 (2.8)	40
50	22031	16885 (76.6)	51 (0.2)	3600 (16.3)	86 (0.4)	1409 (6.4)	50

$|R|$: number of requests; Priority (%): Proportion of high-priority requests; All costs are measured in CNY.

Table 8
Vehicle capacity related sensitivity analyses under different high-priority request ratios.

Capacity state	High-priority ratio (%)	Total cost	Transport cost	Waiting cost	Loading/ unloading cost	Storage cost	Delay penalty	Service rate (%)
C1 (60%)	20	6850	5985	21	840	4	0	60
C1 (60%)	30	6785	5937	6	840	2	0	60
C1 (60%)	40	7389	6450	38	900	2	0	60
C1 (60%)	50	7036	6175	19	840	2	0	60
C2 (80%)	20	9553	7782	28	1740	3	0	90
C2 (80%)	30	9480	7714	21	1740	4	0	90
C2 (80%)	40	9973	8286	34	1650	3	0	80
C2 (80%)	50	9616	7850	22	1740	4	0	90
C3 (100%)	20	8881	7110	27	1740	5	0	90
C3 (100%)	30	8741	6966	30	1740	4	0	90
C3 (100%)	40	9430	7653	30	1740	7	0	90
C3 (100%)	50	9931	8162	25	1740	4	0	90
C4 (120%)	20	9300	7529	22	1740	9	0	90
C4 (120%)	30	8668	6889	34	1740	5	0	90
C4 (120%)	40	9381	7602	36	1740	3	0	90
C4 (120%)	50	9236	7474	17	1740	5	0	90
C5 (140%)	20	8747	6979	25	1740	3	0	90
C5 (140%)	30	8753	6980	27	1740	6	0	90
C5 (140%)	40	9268	7497	28	1740	3	0	90
C5 (140%)	50	9298	7521	37	1740	0	0	90

All costs are measured in CNY. Capacity state indicates the level of truck capacity relative to baseline (e.g., 60% means 60% of baseline capacity).

vehicles. While rebalancing effectiveness experienced a moderate decline, most notably for shared bicycles, the model consistent performs well across diverse node configurations and vehicle types.

Similarly, under highly concentrated inventory conditions (Fig. 15), nearly 100% of nodes exhibited supply-demand imbalances across all vehicle types prior to optimization. Particularly concerning, at least 25% of nodes experienced severe imbalances for each vehicle type category. Following optimization, complete balance was successfully achieved for shared bicycles in 67% of nodes, shared mopeds in 100% of nodes, and malfunctioning vehicles requiring maintenance in 75% of nodes. Remarkably, only Node 6 displayed significant residual imbalance, demonstrating the optimization algorithm's effectiveness even under challenging inventory concentration scenarios.

When inventory distribution becomes dispersed (Fig. 16), the model's rebalancing performance experiences moderate decline while maintaining relatively high effectiveness levels. Specifically, complete balance is attained for shared bicycles in 67% of nodes, shared mopeds in 75% of nodes, and malfunctioning vehicles in 58% of nodes. A notable challenge emerges in the increased prevalence of nodes exhibiting substantial supply-demand imbalances under these conditions.

Table 9
Vehicle speed related sensitivity analyses under different high-priority request ratios.

Speed state	High-priority ratio (%)	Total cost	Transport cost	Waiting cost	Loading/ unloading cost	Storage cost	Delay penalty	Service rate (%)
V1 (60%)	20	9191	7413	28	1740	10	0	90
V1 (60%)	30	9012	7233	25	1740	14	0	90
V1 (60%)	40	9326	7556	26	1740	4	0	90
V1 (60%)	50	9576	7806	24	1740	6	0	90
V2 (80%)	20	9014	7254	16	1740	4	0	90
V2 (80%)	30	9019	7237	31	1740	11	0	90
V2 (80%)	40	9308	7530	34	1740	4	0	90
V2 (80%)	50	9143	7373	23	1740	6	0	90
V3 (100%)	20	8873	7099	30	1740	4	0	90
V3 (100%)	30	8749	6978	27	1740	5	0	90
V3 (100%)	40	9430	7653	30	1740	7	0	90
V3 (100%)	50	9931	8162	25	1740	4	0	90
V4 (120%)	20	9029	7265	17	1740	6	0	90
V4 (120%)	30	8896	7123	24	1740	9	0	90
V4 (120%)	40	9510	7731	37	1740	2	0	90
V4 (120%)	50	9079	7302	32	1740	5	0	90
V5 (140%)	20	9379	7589	45	1740	4	0	90
V5 (140%)	30	9128	7359	25	1740	4	0	90
V5 (140%)	40	9404	7619	38	1740	7	0	90
V5 (140%)	50	9549	7768	37	1740	3	0	90

All costs are measured in CNY. Speed state indicates the level of truck speed relative to baseline (e.g., 60% means 60% of baseline speed).

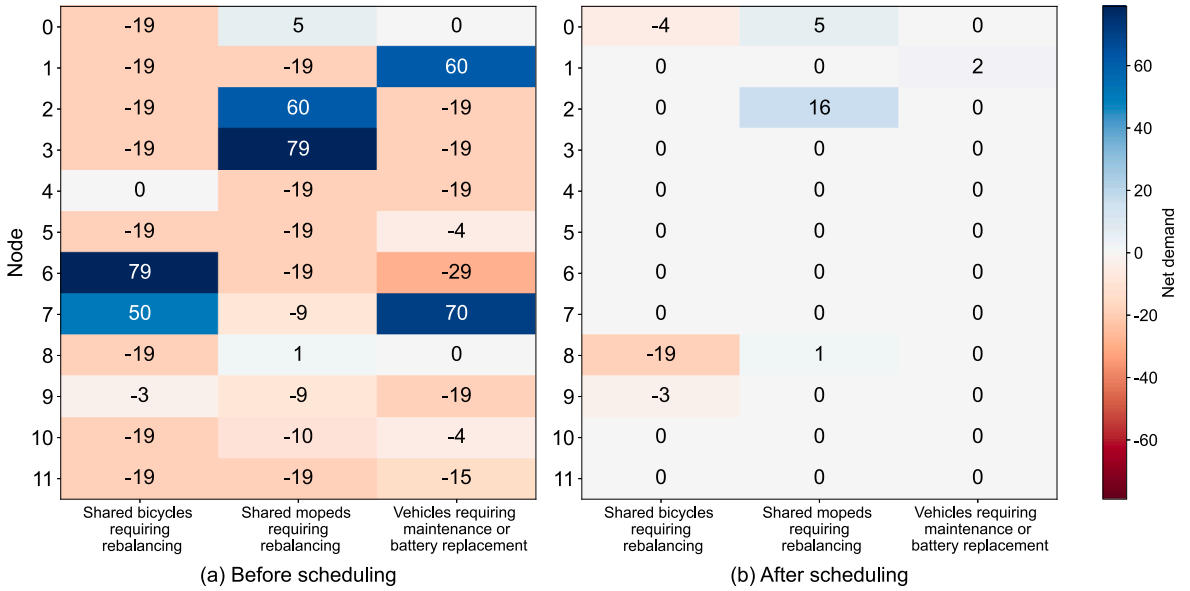


Fig. 13. 80% demands concentrated in 20% nodes.

Table 10
Sensitivity analysis of high-priority ratio under different order volumes and fleet sizes.

Number of orders	High-priority ratio (%)	Fleet size (% of baseline)	Total cost	Transport cost	Waiting cost	Loading/ unloading cost	Storage cost	Delay penalty	Service rate (%)
20	20	N1 (100%)	8881	7110	27	1740	5	0	90
20	30	N1 (100%)	8741	6966	30	1740	4	0	90
20	40	N1 (100%)	9430	7653	30	1740	7	0	90
20	50	N1 (100%)	9931	8162	25	1740	4	0	90
20	20	N2 (200%)	10,092	8306	43	1740	3	0	90
20	30	N2 (200%)	10,049	8273	33	1740	3	0	90
20	40	N2 (200%)	8885	7107	28	1740	10	0	90
20	50	N2 (200%)	9128	7349	36	1740	3	0	90
20	20	N3 (400%)	9119	7355	22	1740	2	0	90
20	30	N3 (400%)	9297	7527	27	1740	4	0	90
20	40	N3 (400%)	9219	7437	41	1740	1	0	90
20	50	N3 (400%)	9018	7252	21	1740	5	0	90
10	20	N1 (100%)	4897	4019	3	870	5	0	100
10	30	N1 (100%)	4594	3716	7	870	1	0	100
10	40	N1 (100%)	4936	4056	9	870	1	0	100
10	50	N1 (100%)	4747	3870	4	870	3	0	100
10	20	N2 (200%)	4961	4081	6	870	5	0	100
10	30	N2 (200%)	4624	3746	4	870	4	0	100
10	40	N2 (200%)	4547	3672	4	870	1	0	100
10	50	N2 (200%)	4554	3676	6	870	2	0	100
10	20	N3 (400%)	4897	4019	3	870	5	0	100
10	30	N3 (400%)	4961	4081	6	870	5	0	100
10	40	N3 (400%)	4747	3870	4	870	3	0	100
10	50	N3 (400%)	4695	3814	8	870	4	0	100
5	20	N1 (100%)	1902	1341	0	510	1	0	100
5	30	N1 (100%)	1890	1329	0	510	0	0	100
5	40	N1 (100%)	1902	1341	0	510	1	0	100
5	50	N1 (100%)	1966	1405	0	510	1	0	100
5	20	N2 (200%)	1966	1405	0	510	1	0	100
5	30	N2 (200%)	1966	1405	0	510	1	0	100
5	40	N2 (200%)	1814	1252	0	510	2	0	100
5	50	N2 (200%)	1902	1341	0	510	1	0	100
5	20	N3 (400%)	1966	1405	0	510	1	0	100
5	30	N3 (400%)	1908	1346	0	510	1	0	100
5	40	N3 (400%)	1899	1338	0	510	1	0	100
5	50	N3 (400%)	1899	1338	0	510	1	0	100

All costs are measured in CNY. Fleet size is expressed as a percentage of the baseline number of trucks (N1 = 100%).

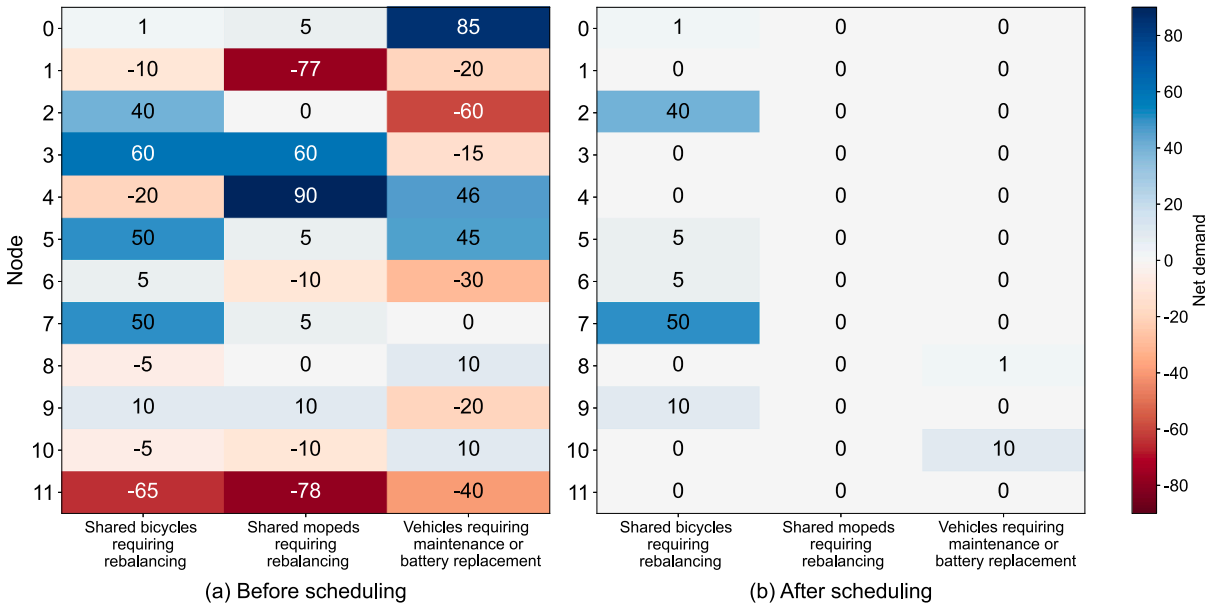


Fig. 14. 50% demands concentrated in 20% nodes.

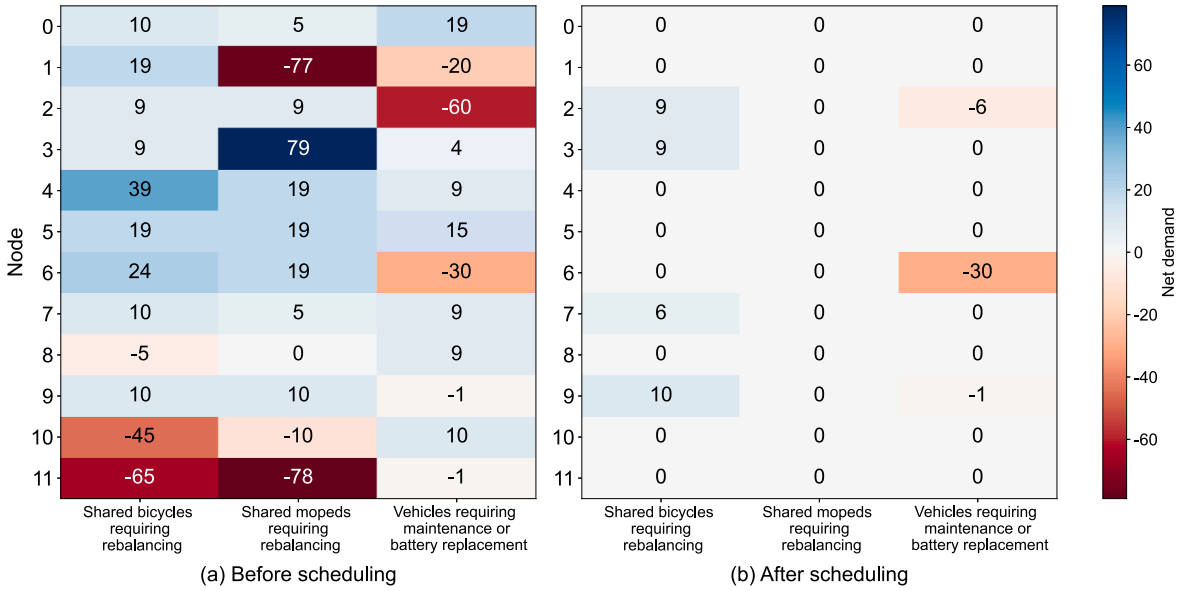


Fig. 15. 80% inventory concentrated in 20% nodes.

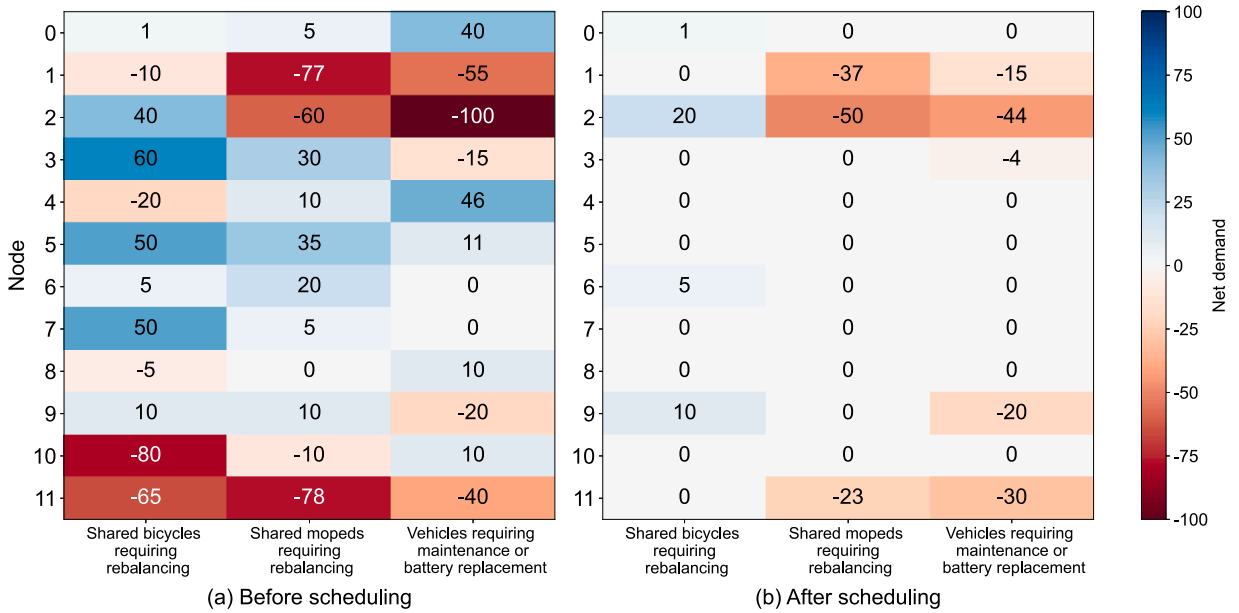


Fig. 16. 50% inventory concentrated in 20% nodes.

7. Conclusions

This study formulates a Multi-commodity Priority-based Multi-task Scheduling Problem (MPMSP) to address operational challenges in shared micro-mobility systems. An optimization model integrating multi-commodity flow, multi-visit route planning, multi-task scheduling mechanisms, and priority-based optimization is developed and solved using a customized Adaptive Large Neighborhood Search (ALNS) algorithm. To evaluate the proposed approach, numerical experiments are conducted that examine the influence of different sub-costs, the performance of ALNS compared to commercial solver and the impact of different operator designs. We also perform comparative experiments between single-task and multi-task scheduling mechanisms, and investigate how different ratios of high-priority requests influence the model. The model's adaptability and generalizability are evaluated under varying spatial distributions of inventory and demand.

The proposed framework demonstrates superior performance across all tested scenarios. Experimental results confirm the rationality and effectiveness of both the proposed cost function and the ALNS operator designs, while demonstrating the algorithm's

superior performance over Gurobi for solving MPMSP. Most notably, the multi-task scheduling mechanism achieves substantial improvements in system efficiency, demonstrating remarkable increases in both shared vehicle availability and service rates compared to conventional single-task approaches. Additionally, the analysis suggests that the formulation of high- and low-priority requests should carefully balance service quality and operational costs. Furthermore, the model maintains efficient and reliable performance under diverse demand distributions and inventory configurations.

In practice, the proposed model provides a decision-support tool for shared micro-mobility system operators that substantially improves system efficiency, stability, convenience, and user satisfaction compared to traditional approaches. Specifically, operators can directly utilize this model to generate operational plans based on available system information, including node locations, inventory and demand distributions, maintenance requirements, and vehicle battery status. By integrating vehicle rebalancing, maintenance, and battery replacement into a unified scheduling framework, the model produces optimized allocation plans that redistribute vehicles to mitigate spatial supply–demand mismatches while addressing necessary maintenance and battery replacement operations, thereby maintaining the shared mobility system in an optimal operational state.

Moreover, the model incorporates critical real-world operational constraints, including limited fleet size and truck capacity, time window restrictions, node capacity limitations, and request heterogeneity. These practical considerations enable the model to address common challenges in shared micro-mobility management, such as vehicle shortages at high-demand nodes, inefficiencies caused by unavailable vehicles, heterogeneous urgent demands, and service delays imposed by time window constraints. The model's outputs provide actionable guidance for resource allocation, scheduling, and operational coordination, ultimately ensuring enhanced service availability and continuous improvement of user experience throughout the network.

Future research directions encompass several promising extensions. First, strengthening collaboration is essential for shared micro-mobility systems to enhance urban sustainable development and improve residents' quality of life. At the multimodal coordination level, integrated scheduling frameworks connecting shared micro-mobility with broader public transport networks merit further development to enhance system-wide efficiency. Incorporating dynamic planning capabilities offers significant potential for addressing time-dependent decision-making and improving system responsiveness. The adoption of uncertainty-based modeling approaches would enable the framework to better accommodate operational disruptions, demand fluctuations, and data variability encountered in real-world operations. Finally, advancing multi-objective optimization methodologies represents a crucial research priority.

CRediT authorship contribution statement

Yimeng Zhang: Writing – original draft, Validation, Software, Methodology, Funding acquisition, Conceptualization; **Shuyang Zhu:** Writing – original draft, Software; **Yuchun Chen:** Writing – original draft; **Peimin Li:** Writing – original draft; **Jie Feng:** Software; **Ruijie Li:** Writing – review & editing; **Xiaobo Liu:** Writing – review & editing; **Mi Gan:** Writing – review & editing; **Shuwen Yang:** Writing – original draft; **Nanxi Chen:** Writing – original draft; **Ruixue Ai:** Writing – original draft, Data curation.

Data availability

Data will be made available on request.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 52402521). This work is also partly co-funded by Sichuan Science and Technology Program (No. 2025NSFSC1969, 2025HJPJ0011), Fundamental Research Funds for the Central Universities of China (No. 2682025CX056), National Natural Science Foundation of China (No. U2568219), Scientific and Technological Research and Development Program of China State Railway Group Co., Ltd. (No. N2025X022-B(JB)), and National Key R&D Program of China (No. 2025YFB2606500).

References

Alvarez-Valdes, R., Belenguer, J.M., Benavent, E., Bermudez, J.D., Muñoz, F., Vercher, E., Verdejo, F., 2016. Optimizing the level of service quality of a bike-sharing system. Omega 62, 163–175.

Benchimol, M., Benchimol, P., Chappert, B., De La Taille, A., Laroche, F., Meunier, F., Robinet, L., 2011. Balancing the stations of a self service "bike hire" system. RAIRO-Oper. Res.-Rech. Op/'eration nelle 45 (1), 37–61.

Brinkmann, J., Ulmer, M.W., Mattfeld, D.C., 2019. Dynamic lookahead policies for stochastic-dynamic inventory routing in bike sharing systems. Comput. Oper. Res. 106, 260–279.

Brinkmann, J., Ulmer, M.W., Mattfeld, D.C., 2020. The multi-vehicle stochastic-dynamic inventory routing problem for bike sharing systems. Bus. Res. 13 (1), 69–92.

Caggiani, L., Camporeale, R., Ottomanelli, M., Szeto, W.Y., 2018. A modeling framework for the dynamic management of free-floating bike-sharing systems. Transp. Res. C: Emerg. Technol. 87, 159–182. https://doi.org/10.1016/j.trc.2018.01.001

Cavaliere, F., Accorsi, L., Laganà, D., Musmanno, R., Vigo, D., 2024. An efficient heuristic for very large-scale vehicle routing problems with simultaneous pickup and delivery. Transp. Res. E: Logist. Transp. Rev. 186, 103550.

Çelebi, D., Yörüsün, A., Işık, H., 2018. Bicycle sharing system design with capacity allocations. Transp. Res. B: Methodol. 114, 86–98.

Çelik, S., Schrotenboer, A.H., Martin, L., Van Woensel, T., 2026. Is waiting worth it? The value of delaying time window assignment in vehicle routing problems. Transp. Res. B: Methodol. 204, 103381.

Chang, S., Song, R., He, S., Qiu, G., 2018. Innovative bike-sharing in china: solving faulty bike-sharing recycling problem. J. Adv. Transp. 2018 (1), 4941029.

Chastre, M.R., Andrade, A.R., 2023. Rebalancing in bike sharing systems: application to the lisbon case study. Case Stud. Transp. Policy 13, 101071.

Chemla, D., Meunier, F., Calvo, R.W., 2013. Bike sharing systems: solving the static rebalancing problem. Discrete Optim. 10 (2), 120–146.

Chen, C., Demir, E., Huang, Y., 2021. An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots. Eur. J. Oper. Res. 294 (3), 1164–1180.

Chen, Q., Ma, S., Li, H., Zhu, N., He, Q.-C., 2024. Optimizing bike rebalancing strategies in free-floating bike-sharing systems: an enhanced distributionally robust approach. Transp. Res. E: Logist. Transp. Rev. 184, 103477.
Chen, Z., Zhang, Z., Bian, Z., Dai, L., Hu, H., 2023. Subsidy policy optimization of multimodal transport on emission reduction considering carrier pricing game and shipping resilience: a case study of shanghai port. Ocean Coast. Manag. 243, 106760.
Chiariotti, F., Pielli, C., Zanella, A., Zorzi, M., 2018. A dynamic approach to rebalancing bike-sharing systems. Sensors 18, 512.
Côté, J.-F., Mansini, R., Raffaele, A., 2024. Multi-period time window assignment for attended home delivery. Eur. J. Oper. Res. 316 (1), 295–309.
Cruz, F., Subramanian, A., Bruck, B.P., Iori, M., 2017. A heuristic algorithm for a single vehicle static bike sharing rebalancing problem. Comput. Oper. Res. 79, 19–33.
De Chardon, C.M., Caruso, G., Thomas, I., 2016. Bike-share rebalancing strategies, patterns, and purpose. J. Transp. Geogr. 55, 22–39.
Dell'Amico, M., Hadjicostantinou, E., Iori, M., Novellani, S., 2014. The bike sharing rebalancing problem: mathematical formulations and benchmark instances. Omega 45, 7–19.
Dell'Amico, M., Montemanni, R., Novellani, S., 2023. Pickup and delivery with lockers. Transp. Res. C: Emerg. Technol. 148, 104022.
Dötterl, J., Bruns, R., Dunkel, J., Ossowski, S., 2017. Towards dynamic rebalancing of bike sharing systems: an event-driven agents approach. In: EPIA Conference on Artificial Intelligence. Springer, pp. 309–320.
Du, M., Cheng, L., Li, X., Tang, F., 2020. Static rebalancing optimization with considering the collection of malfunctioning bikes in free-floating bike sharing system. Transp. Res. E: Logist. Transp. Rev. 141, 102012.
Erdoğan, G., Battarra, M., Calvo, R.W., 2015. An exact algorithm for the static rebalancing problem arising in bicycle sharing systems. Eur. J. Oper. Res. 245 (3), 667–679.
François, V., Arda, Y., Crama, Y., 2019. Adaptive large neighborhood search for multitrip vehicle routing with time windows. Transp. Sci. 53 (6), 1706–1730.
Fricker, C., Gast, N., 2016. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. Euro J. Transp. Logist. 5 (3), 261–291.
Fu, C., Zhu, N., Ma, S., Liu, R., 2022. A two-stage robust approach to integrated station location and rebalancing vehicle service design in bike-sharing systems. Eur. J. Oper. Res. 298 (3), 915–938.
Gammelli, D., Wang, Y., Prak, D., Rodrigues, F., Minner, S., Pereira, F.C., 2022. Predictive and prescriptive performance of bike-sharing demand forecasts for inventory management. Transp. Res. C: Emerg. Technol. 138, 103571.
Gleditsch, M.D., Hagen, K., Andersson, H., Bakker, S.J., Fagerholt, K., 2024. A column generation heuristic for the dynamic bicycle rebalancing problem. Eur. J. Oper. Res. 317 (3), 762–775.
Gschwind, T., Drexl, M., 2019. Adaptive large neighborhood search with a constant-time feasibility test for the dial-a-ride problem. Transp. Sci. 53 (2), 480–491.
Guan, Q., Fan, Y., Wang, Y., Liang, L., Luo, P., Yao, Y., 2026. Real-time multi-depot urban logistics optimization in megacities via transformer-based deep reinforcement learning. Int. J. Geogr. Inf. Sci. 40 (4), 1107–1130.
Guerrero, A., Juan, A.A., Garcia-Sanchez, A., Pita-Romero, L., 2024. Optimizing maintenance of energy supply systems in city logistics with heuristics and reinforcement learning. Mathematics 12 (19), 3140.
Guo, F., Xu, Y., Huang, Z., Wu, Y., 2024. Collaborative optimization of routing and storage strategy of multi-period multimodal transport in an uncertain environment. Comput. Oper. Res. 167, 106676.
Guo, Y., Yang, L., Chen, Y., 2022. Bike share usage and the built environment: a review. Front Public Health 10, 848169. https://doi.org/10.3389/fpubh.2022.848169
Han, Y., Yaman, H., 2024. Formulations and branch-and-cut algorithms for the heterogeneous fleet vehicle routing problem with soft time deadlines. Transp. Res. B: Methodol. 190, 103104.
He, D., Ceder, A.A., Zhang, W., Guan, W., Qi, G., 2023. Optimization of a rural bus service integrated with e-commerce deliveries guided by a new sustainable policy in china. Transp. Res. E: Logist. Transp. Rev. 172, 103069.
Ho, S.C., Szeto, W.Y., 2017. A hybrid large neighborhood search for the static multi-vehicle bike-repositioning problem. Transp. Res. B: Methodol. 95, 340–363.
Hsu, Y.-T., Yan, S., Huang, P., 2021. The depot and charging facility location problem for electrifying urban bus services. Transp. Res. D: Transp. Environ. 100, 103053.
Hu, R., Zhang, Z., Ma, X., Jin, Y., 2021. Dynamic rebalancing optimization for bike-sharing system using priority-based MOEA/d algorithm. IEEE Access 9, 27067–27084.
Idris, M. F.M., Saad, N.H., Yahaya, M.I., Shuib, A., Mohamed, W. M.W., Amin, A. N.M., 2022. Cost of rolling stock maintenance in urban railway operation: literature review and direction. Pertanika J. Sci. Technol. 30 (2).
International Energy Agency, 2023. World energy outlook 2023. https://www.iea.org/reports/world-energy-outlook-2023.
Janovec, M., Kohání, M., 2019. Exact approach to the electric bus fleet scheduling. Transp. Res. Procedia 40, 1380–1387.
Jiang, H., Hong, S., Zhang, K., Yuan, J., Yu, Q., 2026. Two-stage optimization approach for dynamic routing and charging scheduling in electrified-autonomous flexible transit. Transp. Res. E: Logist. Transp. Rev. 207, 104600.
Jiang, J., Dai, Y., Yang, F., Ma, Z., 2024. A multi-visit flexible-docking vehicle routing problem with drones for simultaneous pickup and delivery services. Eur. J. Oper. Res. 312 (1), 125–137.
Jin, Y., Ruiz, C., Liao, H., 2022. A simulation framework for optimizing bike rebalancing and maintenance in large-scale bike-sharing systems. Simul. Model. Pract. Theory 115, 102422. https://doi.org/10.1016/j.simpat.2021.102422
Jin, Y., Ruiz, C., Liao, H., Pierson, H., 2019. A simulation framework for the rebalancing and maintenance of bicycle-sharing systems. In: 2019 Winter Simulation Conference (WSC). IEEE, pp. 819–829.
Jin, Z., Ma, D., Li, P., Li, Y., Zhang, L., 2025. Managing shared electric micromobility systems: allocation planning and battery swapping. Transp. Res. E: Logist. Transp. Rev. 198, 104108.
Kadri, A.A., Kacem, I., Labadi, K., 2016. A branch-and-bound algorithm for solving the static rebalancing problem in bicycle-sharing systems. Comput. Ind. Eng. 95, 41–52.
Kim, E.W., Kim, S., 2021. Optimum location analysis for an infrastructure maintenance depot in urban railway networks. KSCE J. Civ. Eng. 25 (6), 1919–1930.
Kloimüllner, C., Raidl, G.R., 2017. Full-load route planning for balancing bike sharing systems by logic-based benders decomposition. Networks 69 (3), 270–289.
Kraiem, A., Audy, J.-F., Lamghari, A., 2025. Adaptive large neighbourhood search for the multi-depot arc routing problem with flexible assignment of end depot and different arc types. J. Oper. Res. Soc. 76 (7), 1319–1337.
Kuhn, H., Schubert, D., Holzapfel, A., 2021. Integrated order batching and vehicle routing operations in grocery retail–a general adaptive large neighborhood search algorithm. Eur. J. Oper. Res. 294 (3), 1003–1021.
Legros, B., 2019. Dynamic repositioning strategy in a bike-sharing system; how to prioritize and how to rebalance a bike station. Eur. J. Oper. Res. 272 (2), 740–753.
Lehmann, J., Winkenbach, M., 2024. A matheuristic for the two-echelon multi-trip vehicle routing problem with mixed pickup and delivery demand and time windows. Transp. Res. C: Emerg. Technol. 160, 104522.
Li, L., Al Chami, Z., Manier, H., Manier, M.-A., Xue, J., 2021. Incorporating fuel delivery in network design for hydrogen fueling stations: formulation and two metaheuristic approaches. Transp. Res. E: Logist. Transp. Rev. 152, 102384.
Liang, S., Ye, Y., Wu, J., 2025. An effective hybrid optimization algorithm for static rebalance problem of bicycle-sharing system. IET Intell. Transp. Syst. 19 (1), e70050.
López-Santana, E., Méndez, G., Franco, C., 2023. On the multi-period combined maintenance and routing optimisation problem. Int. J. Prod. Res. 61 (23), 8265–8290.
Lu, L., Zhao, S., He, Q.-C., Zhu, N., 2022. Task assignment in predictive maintenance for free-float bicycle sharing systems. Comput. Ind. Eng. 169, 108214.
Ma, H., Pei, W., Zhang, Q., Xu, D., Li, Y., 2023. Location of electric vehicle charging stations based on game theory. World Electr. Veh. J. 14 (5), 128.
Maggioni, F., Cagnolari, M., Bertazzi, L., Wallace, S.W., 2019. Stochastic optimization models for a bike-sharing problem with transshipment. Eur. J. Oper. Res. 276 (1), 272–283.
Mara, S. T.W., Norcahyo, R., Jodiawan, P., Lusiantoro, L., Rifai, A.P., 2022. A survey of adaptive large neighborhood search algorithms and applications. Comput. Oper. Res. 146, 105903.
Martins, S., Ostermeier, M., Amorim, P., Hübner, A., Almada-Lobo, B., 2019. Product-oriented time window assignment for a multi-compartment vehicle routing problem. Eur. J. Oper. Res. 276 (3), 893–909.

Masmoudi, M.A., Hosny, M., Braekers, K., Dammak, A., 2016. Three effective metaheuristics to solve the multi-depot multi-trip heterogeneous dial-a-ride problem. Transp. Res. E: Logist. Transp. Rev. 96, 60–80.
Maximo, V.R., Cordeau, J.-F., Nascimento, M. C.V., 2025. A hybrid adaptive iterated local search heuristic for the maximal covering location problem. Int. Trans. Oper. Res. 32 (1), 176–193.
Mohri, S.S., Asgari, N., Farahani, R.Z., Bourlakis, M., Laker, B., 2020. Fairness in hazmat routing-scheduling: a bi-objective stackelberg game. Transp. Res. Part E: Logist. Transp. Rev. 140, 102006.
Nair, R., Miller-Hooks, E., 2011. Fleet management for vehicle sharing operations. Transp. Sci. 45 (4), 524–540.
Ndhaief, N., Bistorin, O., Rezg, N., 2017. An improved distribution policy with a maintenance aspect for an urban logistic problem. Appl. Sci. 7 (7), 703.
Pan, L., Cai, Q., Fang, Z., Tang, P., Huang, L., 2019. A deep reinforcement learning framework for rebalancing dockless bike sharing systems. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1393–1400.
Pereira, V.G., Alves-Junior, O.C., Baldo, F., 2024. An approach to solve the heterogeneous fixed fleet vehicle routing problem with time window based on adaptive large neighborhood search meta-heuristic. IEEE Trans. Intell. Transp. Syst. 25 (7), 8148–8157.
Pfrommer, J., Warrington, J., Schildbach, G., Morari, M., 2014. Dynamic vehicle redistribution and online price incentives in shared mobility systems. IEEE Trans. Intell. Transp. Syst. 15 (4), 1567–1578.
Raviv, T., Tzur, M., Forma, I.A., 2013. Static repositioning in a bike-sharing system: models and solution approaches. EURO J. Transp. Logist. 2 (3), 187–229.
Regue, R., Recker, W., 2014. Proactive vehicle routing with inferred demand to solve the bikesharing rebalancing problem. Transp. Res. E: Logist. Transp. Rev. 72, 192–209.
Ren, Y., Meng, L., Zhao, F., Zhang, C., Guo, H., Tian, Y., Tong, W., Sutherland, J.W., 2020. An improved general variable neighborhood search for a static bike-sharing rebalancing problem considering the depot inventory. Expert Syst. Appl. 160, 113752.
Santos, V. G.M., de Carvalho, M. A.M., 2021. Tailored heuristics in adaptive large neighborhood search applied to the cutwidth minimization problem. Eur. J. Oper. Res. 289 (3), 1056–1066.
Schiffer, M., Walther, G., 2018. An adaptive large neighborhood search for the location-routing problem with intra-route facilities. Transp. Sci. 52 (2), 331–352.
Shoushtari, F., Talebi, M., Rezvanjou, S., 2024. Electric vehicle charging station location by applying optimization approach. Int. J. Ind. Eng. Oper. Res. 6 (1), 1–15.
Shui, C.S., Szeto, W.Y., 2018. Dynamic green bike repositioning problem–a hybrid rolling horizon artificial bee colony algorithm approach. Transp. Res. D: Transp. Environ. 60, 119–136.
Singla, A., Santoni, M., Bartók, G., Mukerji, P., Meenen, M., Krause, A., 2015. Incentivizing users for balancing bike sharing systems. In: Proceedings of the AAAI Conference on Artificial Intelligence.
Statista, 2025. Bike-sharing - worldwide. https://www.statista.com/outlook/mmo/shared-mobility/bike-sharing/worldwide.
Sun, Z., Gao, W., Li, B., Wang, L., 2020. Locating charging stations for electric vehicles. Transp. Policy 98, 48–54.
Tang, L., Li, Y., Zhang, S., Wang, Z., Coelho, L.C., 2025. Mobile COVID-19 vaccination scheduling with capacity selection. Transp. Res. E: Logist. Transp. Rev. 193, 103826.
Voigt, S., 2025. A review and ranking of operators in adaptive large neighborhood search for vehicle routing problems. Eur. J. Oper. Res. 322 (2), 357–375.
Wang, S., Sun, W., Baldacci, R., Elomri, A., 2025a. Exact solution of location–routing problems with heterogeneous fleet and weight-based carbon emissions. Transp. Res. E: Logist. Transp. Rev. 193, 103862.
Wang, W., Adulyasak, Y., Cordeau, J.-F., He, G., 2025b. The heterogeneous-fleet electric vehicle routing problem with nonlinear charging functions. Transp. Res. C: Emerg. Technol. 170, 104932.
Wang, Y., Luo, S., Zhen, L., 2026. Collaborative multicenter delivery and pickup network design with transportation resource configuration and alliance synergy service. Transp. Res. E: Logist. Transp. Rev. 205, 104484.
Wang, Y., Wei, Z., Luo, S., Zhou, J., Zhen, L., 2024. Collaboration and resource sharing in the multidepot time-dependent vehicle routing problem with time windows. Transp. Res. E: Logist. Transp. Rev. 192, 103798.
Wang, Y.-W., Lin, C.-C., 2013. Locating multiple types of recharging stations for battery-powered electric vehicle transport. Transp. Res. E: Logist. Transp. Rev. 58, 76–87.
Warrington, J., Ruchti, D., 2019. Two-stage stochastic approximation for dynamic rebalancing of shared mobility systems. Transp. Res. C: Emerg. Technol. 104, 110–134.
Woo, S., Choi, E.Y., Moura, S.J., Borrelli, F., 2024. Saving energy with eco-friendly routing of an electric vehicle fleet. Transp. Res. E: Logist. Transp. Rev. 189, 103644.
Xu, D., Zhen, L., Chan, H.K., Wang, J., Cui, L., 2024. An exact algorithm for unpaired pickup and delivery vehicle routing problem with multiple commodities and multiple visits. Transp. Res. C: Emerg. Technol. 160, 104488.
Xu, M., Di, Y., Zhu, Z., Yang, H., Chen, X., 2022. Designing van-based mobile battery swapping and rebalancing services for dockless ebike-sharing systems based on the dueling double deep q-network. Transp. Res. C: Emerg. Technol. 138, 103620.
Xu, S., Ou, X., Govindan, K., Chen, M., Yang, W., 2025. An adaptive genetic hyper-heuristic algorithm for a two-echelon vehicle routing problem with dual-customer satisfaction in community group-buying. Transp. Res. E: Logist. Transp. Rev. 194, 103874.
Yan, Y., Shang, W.-L., Yan, J., Liao, Q., Wang, B., Song, H., Liu, Y., 2022. Logistic and scheduling optimization of the mobilized and distributed battery in urban energy systems. Resour. Conserv. Recycl. 187, 106608.
Yang, D., Hyland, M.F., Jayakrishnan, R., 2024. Tackling the crowdsourced shared-trip delivery problem at scale with a novel decomposition heuristic. Transp. Res. E: Logist. Transp. Rev. 188, 103633.
Zandieh, F., Ghannadpour, S.F., Mazdeh, M.M., 2023. Integrated ground vehicle and drone routing with simultaneous surveillance coverage for evading intentional disruption. Transp. Res. E: Logist. Transp. Rev. 178, 103266.
Zhang, C., Dong, M., Luan, T.H., Ota, K., 2020. Battery maintenance of pedelec sharing system: big data based usage prediction and replenishment scheduling. IEEE Trans. Netw. Sci. Eng. , 127–138. https://doi.org/10.1109/TNSE.2019.2901833
Zhang, C., Wu, F., Wang, H., Tang, B., Fan, W., Liu, Y., 2022. A meta-learning algorithm for rebalancing the bike-sharing system in iot smart city. IEEE Internet Things J. 9, 21073–21085.
Zhang, S., Xiang, G., Huang, Z., 2018. Bike-sharing static rebalancing by considering the collection of bicycles in need of repair. J. Adv. Transp. 2018 (1), 8086378.
Zhang, Y., Mi, Z., 2018. Environmental benefits of bike sharing: a big data-based analysis. Appl. Energy 220, 296–301.
Zhang, Y., Shao, Y., Bi, H., Aoyong, L., Ye, Z., 2023. Bike-sharing systems rebalancing considering redistribution proportions: a user-based repositioning approach. Phys. a: Stat. Mech. Appl. 610, 128409.
Zheng, Y.-J., Chen, X., Yan, H.-F., Zhang, M.-X., 2023. Evolutionary algorithm for vehicle routing for shared e-bicycle battery replacement and recycling. Appl. Soft Comput. 135, 110023.