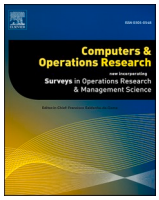
Computers & Operations Research 173 (2025) 106828

Contents lists available at ScienceDirect

Computers and Operations Research

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



Vehicle routing problem for omnichannel retailing including multiple types of time windows and products

Ning Li [a,b], Zheng Wang [a,b,*]

[a] School of Automation, Southeast University, Nanjing 210096, China
[b] Key Laboratory of Measure and Control of Complex Engineering Systems, Ministry of Education, Southeast University, Nanjing 210096, China

ARTICLE INFO

Keywords:
Vehicle routing problem
Omnichannel retailing
Multiple types of products
Split delivery
Multiple types of time windows

ABSTRACT

This paper addresses the capacitated vehicle routing problem for omnichannel retailing with multiple types of time windows (hard time windows and soft time windows) and multiple types of products simultaneously. The problem aims to transfer multiple products from a central warehouse to stores and satellites, where split delivery is allowed since the demands of stores or satellites are large and the different product types have different volumes. Based on different characteristics of purchasing channels, the time window of a store is a hard constraint while the time window of a satellite is a soft one. This type of vehicle routing problem exists extensively in omnichannel retail distribution systems in the logistics industry of China, which considers multiple purchasing channels for customers. Since this type of vehicle routing problem is an NP-hard problem and more complicated than the conventional vehicle routing problem with time windows, we design an adaptive large neighborhood search (ALNS) method to solve it. Finally, numerical experiments are conducted to evaluate the effectiveness of the proposed algorithm and reveal some managerial insights. The numerical experiments show that the proposed algorithm can obtain a high-quality solution under a shorter computation time compared with the state-of-the-art MIP solver (Gurobi).

1. Introduction

With the development of e-commerce, customer behavior has been changed, which forces retailers to make a revolution of traditional retailing in order to attract more customers. As a consequence, traditional retailers adopt omnichannel retailing (i.e., online channel, offline channel and buy-online-and-pick-up-in-store (BOPS) channel) to satisfy multiple types of customer behavior. Although omnichannel retailing can provide a better shopping experience for customers, it makes the product distribution system more sophisticated and brings new assumptions compared with traditional product distribution systems.

In omnichannel retailing, the retailer always needs to consider the transportation of products for stores and the transportation of online orders for online customers. Investigating the omnichannel retailers in China, e.g., JD.com, Tmall, etc., the product distribution system in omnichannel retailing consists of a central warehouse, satellites, and physical stores. For physical stores, the products are delivered from a central warehouse to stores by vehicles. For online customers, the online orders are firstly delivered from a central warehouse to satellites by larger vehicles (e.g., trucks) and then from the satellites to online customers by smaller vehicles. Therefore, the vehicle routing problem for omnichannel retailing (VRPOR) is a two-echelon vehicle routing problem: the first echelon of VRPOR (FVRPOR) is the transportation of products and online orders from a central warehouse to stores or satellites; and the second echelon of VRPOR (SVRPOR) is the transportation of online orders from satellites to online customers. However, we can find that the SVRPOR is a traditional vehicle routing problem with time windows. Therefore, we only focus on the FVRPOR.

For the FVRPOR, due to multiple products and the large demands of satellites and stores, split delivery is usually adopted to reduce the total cost. Notably, the split delivery quantities of products and online orders must be an integer after the split operation in practice, which increases the difficulty in solving this problem (i.e., a packaged online order for a customer or a product cannot be split). However, most literature ignores this integer constraint for simplicity. Besides, to ensure the normal operation of stores and the secondary delivery of online orders every day, there are delivery time constraints (i.e., time windows) for satellites and stores (i.e., the arriving time of products or online orders needs to be earlier than the business time of the stores or the working time of the satellites every day). Based on different features of purchasing channels,

* Corresponding author.
E-mail addresses: 230208165@seu.edu.cn (N. Li), wangz@seu.edu.cn (Z. Wang).

https://doi.org/10.1016/j.cor.2024.106828
Received 20 January 2024; Received in revised form 27 August 2024; Accepted 28 August 2024
Available online 1 September 2024

when facing shortage of products, unsatisfied demand from stores will be lost whereas unsatisfied demand from online channel will be backlogged (Li and Wang, 2023). Therefore, the time windows of stores are hard constraints (i.e., the delivered products for stores must arrive within the time window). Moreover, for online orders exceeding the committed delivery time, omnichannel retailers usually provide some remedial policies (i.e., providing coupons, returning some cash and so on) for online customers, which prevents online customers from canceling their orders. Hence, the time windows of satellites are considered as soft constraints unlike physical stores. Briefly, there exist two types of time windows (i.e., the hard time windows for stores and the soft time windows for satellites) in the FVRPOR.

In this article, we investigate the FVRPOR (i.e., how to transfer products and online orders from a central warehouse to stores and satellites). First, we model this type of vehicle problem, considering multiple types of time windows and products. Due to the features of online and offline channels, the time windows for stores and satellites are different. Besides, split delivery can be allowed in the FVRPOR to reduce the total cost. To solve the FVRPOR, we propose a modified adaptive large neighborhood search method (ALNS) with specially designed destroy and repair operations.

To the best of our knowledge, only a limited part of vehicle routing literature considers omnichannel retailing. However, the problems in these studies are inconsistent with the vehicle routing problem for omnichannel retailing in China. Our research contributions to existing literature are two-folds (model and methodology). On the model aspect, we jointly consider some realistic features in the FVRPOR: (1) multiple products/online orders where different types of products/online orders have different volumes; (2) multiple types of time windows (i.e., the time window of store as a hard constraint and the time window of satellite as a soft constraint); (3) split delivery and the extra cost caused by it. Especially, the split delivery quantity of products/online orders is an integer (i.e., split integer constraint), which is consistent with practice. On the methodology aspect, due to jointly considering those realistic features, existing methods cannot be used to solve the constructed model. Therefore, we propose a modified adaptive large neighborhood search (ALNS) method to solve this problem. Aimed at the realistic features, the modifications of this method based on the framework of ALNS are: (1) the method considers the penalty cost violating online soft time windows and the split delivery cost caused by split delivery; (2) we modify and design some destroy operations, based on the realistic features; (3) we modify and design some repair operations, where due to split integer constraint and multiple products/online orders in the FVRPOR, those repair operations need to consider two subproblems: Subproblem 1 (how to find the optimal feasible inserting positions and the corresponding allocated volumes for the node) and Subproblem 2 (how to allocate multiple products or online orders based on the inserting positions and allocated volumes). Finally, numerical experiments show that our proposed modified ALNS can solve the constructed FVRPOR effectively and efficiently, and can obtain a high-quality solution in a shorter computation compared with Gurobi. Notably, since the proposed method is a modified version of ALNS based on the realistic features, this method can still solve some standard vehicle routing problems (i.e., vehicle routing problem (VRP), vehicle routing problem with time windows (VRPTW) and the split delivery vehicle routing problem with time windows (SDVRPTW)) and has the same performance with the standard ALNS for those standard problems.

Notably, although our studied problem can be regarded as an extension of the split delivery vehicle routing problem with time windows (SDVRPTW), the existing methods of SDVRPTW cannot be used to solve our studied problem due to the features of our studied problem. Compared with SDVRPTW, our studied problem jointly considers multiple types of time windows, multiple products, split delivery and split integer constraint.

The rest of the paper is organized as follows. Section 2 reviews relevant literature. In Section 3, we formulate the FVRPOR including multiple products and some resource constraints (multiple types of time windows and vehicle capacity). In Section 4, we propose a modified ALNS method with specifically designed destroy and repair operations to solve this vehicle routing problem. In Section 5, numerical experiments are conducted to evaluate the performance of the proposed method. Finally, Section 6 concludes the paper and proposes some issues for future research.

2. Literature review

The problem studied in this paper is mainly related to three lines of research, namely vehicle routing problem with time windows, omnichannel vehicle routing problem and related methodology. To position this study, we first present the differences between our study and related literature as shown in Table 1 and then further review the related literature.

2.1. Vehicle routing problem with time windows

The transportation cost of products takes up a large proportion of the total cost. Higher efficiency and productivity of transportation systems are helpful to reduce the total cost. A vehicle routing problem with time windows is the one of optimizing transportation efficiency (Pan et al., 2021a; Huang et al., 2023; Song et al, 2020). Generally, there are two types of time windows (hard time window and soft time window) in this problem. Zhang et al. (2020) proposed a novel reinforcement learning method called Muli-Agent attention Model to solve a multi-vehicle routing problem with soft time windows. Aiming at the multi-depot vehicle routing problem with soft time windows, Fan et al. (2021) introduced a temporal-spatial distance to generate an initial solution and then employed an adaptive neighborhood search method to solve it. Pan et al. (2021b) proposed a hybrid algorithm based on tabu searchand adaptive large neighborhood search to solve a time-dependent vehicle routing problem with hard windows. Under autonomous transportation, Li et al. (2023) studied a vehicle routing problem considering pickup-delivery and hard time windows by employing a multitask-based evolutionary method to solve it.

The split delivery vehicle routing problem with time windows (SDVRPTW) is a variation of the classical vehicle routing problem, where the delivery demand can be split between two or more vehicles. The SDVRPTW has been paid much attention from researchers (Yan et al., 2015; Desaulniers, 2010; Wu et al., 2024). Luo et al. (2017) proposed an exact branch-and-price-and-cut method to solve a new SDVRPTW with linear weight-related cost. Li et al. (2020) modeled a vehicle routing problem considering split delivery and multiple time windows to be a three-index vehicle flow model and proposed a branch-and-price-and-cut method to solve it. Zhao et al. (2023) proposed a two-stage search quantum particle swarm optimization method to solve a green split vehicle routing problem including multiple products and soft time windows. Considering the limitations of split delivery and customer inconvenience, Bianchessi et al. (2019) studied a split delivery vehicle routing problem with time windows and designed an extended branch-and-cut method to solve it. Wang and Wen (2020) proposed an adaptive genetic algorithm method to study a low-carbon vehicle routing problem including mixed time windows (i.e., hard time window and soft time window).

For the literature about vehicle routing with time windows, they do not allow the split delivery for products, where only Huang et al. (2023) and Song et al. (2020) consider multiple products or tasks and others focus on single product or single tasks. For the literature about the split delivery vehicle routing problem with time windows, Yan et al. (2015), Desaulniers (2010), Wu et al. (2021), Luo et al. (2017), and Li et al. (2020) consider the split delivery vehicle routing problem for single product and Desaulniers (2010), Wu et al. (2024), Luo et al. (2017), and Zhao et al. (2023) ignore the constraint that the split delivery quantities of products are integers. Except for Wang and Wen (2020), the above

Table 1
Review of related literature.

	Omnichannel	Split delivery	Split integer constraint[1]	Multiple types of products	Multiple types of time windows[2]
Pan et al. (2021a), Zhang et al. (2020), Fan et al. (2021), Li et al. (2023)					
Huang et al. (2023), Song et al. (2020)				√	
Wang and Wen (2020)					√
Desaulniers (2010), Wu et al. (2024), Luo et al. (2017)		√			
Yan et al. (2015), Li et al. (2020), Bianchessi et al. (2019)		√	√		
Zhao et al. (2023)		√		√	
Guo et al. (2021), Schubert et al. (2021), Paul et al. (2019)	√				
Gao et al. (2023), Liu et al. (2021), Yang and Li (2023), Bayliss et al. (2020), Abdulkader et al. (2018), Akyüz et al. (2022), Arslan et al. (2021)	√			√	
Our research	√	√	√	√	√

[1] **Split integer constraint**: split delivery quantities must be an integer (i.e., a packaged online order for a customer or a product cannot be split).
[2] **Multiple types of time windows**: there are different types of time windows (i.e., hard time window and soft time window) in this problem at the same time.

literature only considers a single type of time window (hard window or soft window). Although Wang and Wen (2020) consider multiple time windows, they only focus on a single product and also do not allow the split delivery for products. Different from the above literature, we study a split delivery vehicle routing problem for omnichannel retailing with multiple types of time windows and products, where different types of demands have different types of time windows.

2.2. Omnichannel vehicle routing problem

In recent years, retailers have employed omnichannel retailing to improve the shopping experience. Therefore, omnichannel retailing has been a mainstream retailing model. As a significant part of omnichannel retailing, the omnichannel transportation of products has gotten attention from lots of researchers (Guo et al., 2021; Gao et al., 2023; Schubert et al., 2021). Generally, omnichannel vehicle routing can be regarded as a two-echelon vehicle routing problem. Based on different practical situations, researchers focus on one or both of the echelons. Sluijk et al. (2023) conducted a literature review for two-echelon vehicle routing problems and their real-world inspired variants. Akyüz et al. (2022) proposed an iterative matheuristic to solve an omnichannel logistics problem including multi-item orders, where a concave pricing policy was used to solve the transportation between the stores and the customer. Arslan et al. (2021) modeled an online channel-driven distribution network deployment problem to be a two-stage stochastic program and proposed an exact solution method based on scenario sampling and the integer L-shaped method to solve it. Liu et al. (2021) proposed a multi-objective approach to solve a vehicle routing problem in omnichannel retailing by jointly considering the cost of distribution network and customer convenience. Yang and Li (2023) proposed two heuristic methods (a multirestart randomized tabu thresholding method and a memetic method) to solve a multiproduct pickup and delivery vehicle routing problem with time windows under omnichannel retailing. Bayliss et al. (2020) studied a pickup and delivery omnichannel vehicle routing problem and proposed a two-phase local search to solve this problem. Abdulkader et al. (2018) proposed two solution approaches (two-phase method and multi-ant colony method) to solve an omnichannel retailing vehicle routing problem based on inventory availability. Paul et al. (2019) considered the shared capacity among different sales channels for omnichannel routing problem and proposed a knapsack-based heuristic method to solve the capacity vehicle routing problem.

In the above mentioned studies, split delivery is not allowed. Guo et al. (2021), Schubert et al. (2021), and Paul et al. (2019) focus on the omnichannel vehicle routing problem for single product. Besides, the literature ignores different types of time windows for products and online orders under omnichannel retailing. Different from the literature, our study researches the split delivery omnichannel vehicle routing problem with multiple types of time windows, where the split delivery is helpful to reduce the total cost and the time windows of products (or online orders) are hard (or soft) constraints. Besides, we also consider vehicle capacity allocation caused by multiple products.

2.3. Methodology

In terms of methodology, our proposed *meta*-heuristic method is related to the literature about the adaptive large neighborhood search (ALNS) method (Shaw, 1998; Ropke and Pisinger, 2006; Chen et al., 2021). Liu et al. (2023) developed an efficient algorithm based on ALNS method to solve a rebalancing problem for a free-floating electric vehicle sharing systems. In Dönmez et al., (2022), the ALNS method is used to solve a mixed fleet vehicle routing problem with time windows. Žulj et al. (2018) proposed a hybrid heuristic method based on ALNS and tabu search to solve an order-batching problem. Yu et al. (2017) researched a robust traditional gate assignment problem considering a wider scope and designed an ALNS to solve it. Wen et al. (2016) employed the ALNS method to research an electric vehicle scheduling problem.

Notably, since our problem is different from the problems in the above literature, the ALNS method in the literature cannot be used to solve our studied problem directly. Therefore, we design a modified ALNS method based on the features of our problem. In particular, we design some destroy and repair operations of this search method based on the features of our problem.

3. Problem description

In this section, we describe the first echelon of VRPOR (FVRPOR) and formulate this problem to be a mathematical model, based on which we propose a modified ALNS method to minimize the total cost in Section 4. To facilitate the construction of the FVRPOR, some notations to be used are listed in Table 2.

For omnichannel retailing, a customer can purchase some products in three ways: visiting the physical store, purchasing products online and buy-online-and-pick-up-in-store way. Therefore, an omnichannel retailer usually needs to consider the transportation of products for stores and the transportation of online orders for online customers. For physical stores, large vehicles are used to deliver products from a central warehouse to stores. For online customers, large vehicles deliver products from a central warehouse to satellites, and then small vehicles deliver products from satellites to online customers. Based on the above analysis, the VRPOR can be regarded as a two-echelon distribution network with a central warehouse, a set of stores, a set of satellites and a set of customers. The first echelon distribution network FVRPOR is the transportation of products and online orders from a central warehouse to the stores and the satellites by some large vehicles; and the second echelon distribution network SVRPOR is the transportation of online orders from the satellites to the online customers by some small vehicles. Compared with the FVRPOR, the SVRPOR only needs to consider online orders. Moreover, the SVRPOR can be divided into some independent

Table 2
Notations.

Symbol	Definition
S	The set of stores
M	The set of satellites where 0 denotes the central warehouse
L	The set of arcs,$L = \{(i,j) : (i,j) \in (S \cup M) \times (S \cup M), i \neq j\}$
R	The set of product type
B	The set of packaging boxes (online orders) type
VF	A fleet of large vehicles
Y	A direct grap,$Y = \{S \cup M, L\}$
d_{ij}	The transportation distance from node i to node j on arc (i,j)
t_{ij}	The traveling time from node i to node j on arc (i,j)
o_i	The service time at node i
v_r	The volume of product type r, $r \in R$
v_b	The volume of packaging box (online order) type b, $b \in B$
V^*	The volume capacity of vehicle
D_i^r	The demand of product type r at the store i, $i \in S$, $r \in R$
D_i^b	The demand of online order type b at the satellite i, $i \in M/\{0\}$, $b \in B$
l_i	The delivery time for node i, $i \in \{S \cup M\}/\{0\}$, where the demand of store has a hard time constratint l_i, $i \in S$ and the demand of satellite has a soft time constraint l_i, $i \in M/\{0\}$
p_u	Unit penalty cost for delivery time (i.e., the penalty for per unit of delayed time and per volume of online orders violating time windows)
p_s	Extra cost caused by the split delivery
$(a)^+$	$max(a,0)$
Decision variables:	
x_{ij}^k	Binary variable that equals 1 if the network edge (i,j) is used by vehicle k, $(i,j) \in L$, $k \in VF$
Q_j^{kr}	Nonnegative integer variable indicating the quantity of product type r carried by vehicle k to satisfy the demand of store j, $j \in S$, $k \in VF$, $r \in R$
Y_j^{kb}	Nonnegative integer variable indicating the quantity of online order type b carried by vehicle k to satisfy the demand of satellite j, $j \in M$, $k \in VF$, $b \in B$
t_j^k	Arriving time of the large vehicle k at the store or satellite j, $j \in M \cup S$, $k \in VF$. For $j \in M \cup S \backslash \{0\}$, if $t_j^k = 0$, node j is not visited by vehicle k

distribution subnetworks centered around satellite nodes, where each subnetwork is composed of satellite node and the online customers serviced by this satellite. Therefore, the SVRPOR is simpler than the FVRPOR. Besides, the SVRPOR can be seen as a traditional vehicle routing problem with time windows, which has been researched by lots of literature. Based on the above analysis, we only research the FVRPOR (i.e., the transportation of products and online orders from a central warehouse to the stores and satellites), where a homogeneous vehicle fleet is used and every vehicle has a specific capacity.

For the transportation of products and online orders from a central warehouse to the satellites and the stores, the vehicle route starts from a central warehouse, visits a subset of satellites and stores, and ends at the central warehouse. Generally, the stores and the satellites require the arriving time of products or online orders before the delivery time (e.g., the business time of the stores or the working time of the satellites). In omnichannel retailing, based on the intensity of desired products, customers select their favorite channels to purchase products. When facing shortage of products, unsatisfied demand from stores will be lost whereas unsatisfied demand from online channel will be backlogged (Li and Wang, 2023). Therefore, the delivery time constraints for stores are hard time windows (i.e., the delivery products for stores must arrive within the time constraints). For online products violating delivery time constraints, omnichannel retailers provide some remedial methods (i.e., reduction of shipping cost or online shopping voucher) to compensate consumers, which makes customers accept those products. Therefore, the delivery time constraints for satellites (i.e., the time constraints for online orders) are soft time windows.

Further, since the demands of stores or satellites are larger and the different product types have different volumes, split delivery often emerges in the FVRPOR, which makes the stores or satellites be visited by multiple vehicles until the demand of these stores or satellites can be satisfied. It is noted that since the online order is packaged at the warehouse, the omnichannel retailer actually transports the packaged products based on the online order into satellites in the FVRPOR. By investigating omnichannel retailers in China, the retailers generally provide multiple types of packaging boxes to package online orders. In our studied FVRPOR, we only consider the vehicle volume restrictions. Therefore, different online orders packaged by the same type of packaging box can be seen as the same type of online order having the same volume (i.e., the volume of used packaging box). Besides, since the satellite node first gathers different online orders from different online customers and then delivers those online orders to online customers in the second echelon, there are multiple types of online orders in the satellite node and the quantity of each type of online order may be greater than one. Due to allowing split delivery, those multiple types of online orders in the satellite node can be split and carried by different vehicles. However, the split delivery quantity of each type of online order by different vehicles in the satellite node (i.e., Y_j^{kb}) must be an integer (i.e., split integer constraint: single online order (the packaged products based on the single online order) cannot be split in the transportation process). Fig. 1 shows the split delivery and split integer constraint for satellite node with three types of online orders. Similar to the satellite node, the demands of multiple types of products in the store node can also be split and carried by different vehicles where the split delivery quantity of each type of product in the store node (i.e., Q_j^{kr}) must be an integer (i.e., split integer constraint: single product cannot be split in the transportation process). Although the two types of node both consider split delivery and split integer constraint, they are not the same. In terms of volume, since the online order actually is composed of different types of products, the volume of online order is different from the volume of product. In terms of time window, the time window of store is a hard constraint while the time window of satellite is a soft constraint. Besides, due to the split delivery, the frequency of loading and unloading processes for products increases with the number of splitting times, which leads to an extra cost.

Notably, based on the above analysis, the FVRPOR jointly considers some real-life features (i..e, multiple types of products/online orders, multiple types of time windows, split delivery and split integer constraint). Therefore, the FVRPOR cannot be simply regarded as a traditional vehicle routing problem. Besides, from Table 1 in Section 2, we can find the differences between the variations of traditional VRP in related literature and our studied problem.

We define the vehicle routing problem on a direct graph $Y = \{S \cup M, L\}$. For online orders, the omnichannel retailer provides some packaging boxes with different volumes to package the online order. Therefore, the volume of online orders during transportation actually equals the volume of the packaging box. Further, based on the volume of online orders, we can classify the online orders. Then, we can assume that the set of online order types equals the set of packaging box type B and the volume of online order type b equals the volume of the packaging box v_b.

Further, we present the mathematical formulation of this transportation. In the transportation of products online orders from a central warehouse to the satellites and the stores, there is a fleet of large vehicles represented by VF, with a volume capacity of V^*. $|VF|$ is the total number of large vehicles at the central warehouse and satisfies the following inequality:

$$|VF| \geqslant \left\lceil \frac{\sum_{r \in R} \sum_{j \in S} v_r D_j^r + \sum_{j \in M/\{0\}} \sum_{b \in B} v_b D_j^b}{V^*} \right\rceil \tag{1}$$

Then, we model the distribution network for the FVRPOR by the following mixed-integer programming formulation:

$$\begin{aligned} \min \sum_{(i,j) \in L} \sum_{k \in VF} d_{ij} x_{ij}^k + \sum_{j \in M/\{0\}} \sum_{k \in VF} \left[p_u \left(t_j^k - l_j \right)^+ \sum_{b \in B} v_b Y_j^{kb} \right] \\ + p_s \sum_{j \in S \cup M \backslash \{0\}} \left[\left(\sum_{i \in S \cup M \backslash \{j\}} \sum_{k \in VF} x_{ij}^k \right) - 1 \right] \end{aligned} \tag{2}$$

subject to

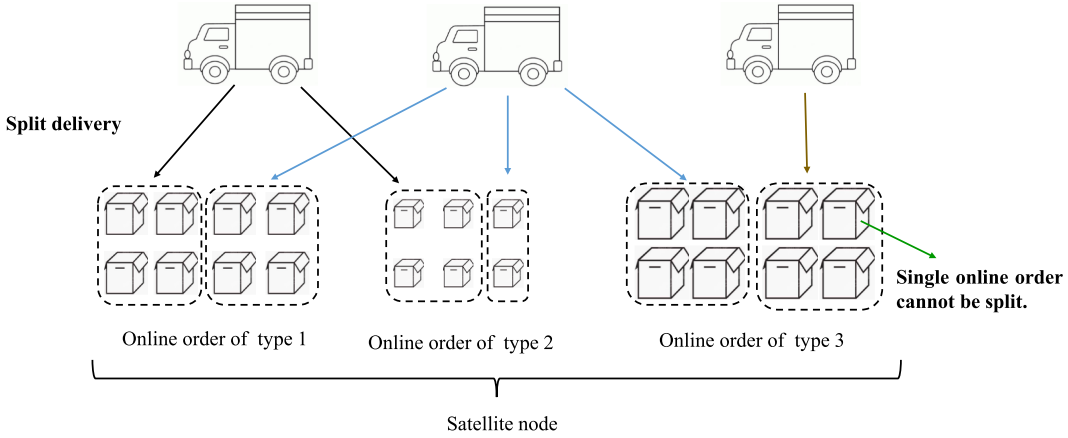


Fig. 1. The split delivery and split integer constraint for satellite node.

$$\sum_{j\in S\cup M\setminus\{0\}}\sum_{k\in VF}x_{0j}^{k}\leqslant|VF| \tag{3}$$

$$\sum_{j\in S\cup M\setminus\{0\}}x_{0j}^{k}=\sum_{i\in S\cup M\{0\}}x_{i0}^{k}\leqslant 1,\forall k\in\ VF \tag{4}$$

$$\sum_{i\in S\cup M}x_{ij}^{k}\leqslant 1,\forall k\in\ VF,j\in S\cup M\setminus\{0\} \tag{5}$$

$$\sum_{i\in S\cup M}x_{ij}^{k}-\sum_{i\in S\cup M}x_{ji}^{k}=0,\forall k\in\ VF,j\in S\cup M\setminus\{0\} \tag{6}$$

$$Q_{j}^{kr}\leqslant D_{j}^{r}\sum_{i\in S\cup M}x_{ij}^{k},\forall k\in\ VF,j\in S,r\in R \tag{7}$$

$$Y_{j}^{kb}\leqslant D_{j}^{b}\sum_{i\in S\cup M}x_{ij}^{k},\forall k\in\ VF,j\in M,b\in B \tag{8}$$

$$\sum_{j\in S}\sum_{r\in R}v_{r}Q_{j}^{kr}+\sum_{j\in M}\sum_{b\in B}v_{b}Y_{j}^{kb}\leqslant V^{*},\forall k\in\ VF \tag{9}$$

$$\sum_{k\in VF}Q_{j}^{kr}=D_{j}^{r},\forall j\in S,r\in R \tag{10}$$

$$\sum_{k\in VF}Y_{j}^{kb}=D_{j}^{b},\forall j\in M\setminus\{0\},b\in B \tag{11}$$

$$t_{j}^{k}=\sum_{i\in S\cup M}\left(t_{i}^{k}+t_{ij}+o_{i}\right)x_{ij}^{k},\forall k\in\ VF,j\in S\cup M\setminus\{0\} \tag{12}$$

$$t_{j}^{k}\leqslant l_{j},\forall k\in\ VF,j\in S \tag{13}$$

$$x_{ij}^{k}\in\{0,1\},\forall k\in\ VF,i,j\in S\cup M \tag{14}$$

$$Y_{j}^{kb}\text{nonnegative integer},\forall k\in\ VF,j\in M\setminus\{0\},b\in B \tag{15}$$

$$Q_{j}^{kr}\text{nonnegative integer},\forall k\in\ VF,j\in S,r\in R \tag{16}$$

The objective function Eq. (2) is to minimize the total cost, which includes three types of costs: the transportation distance (cost) traveled by all vehicles, the penalty cost violating online time windows and the split delivery cost. Notably, the last term of Eq. (2)denotes the split delivery cost, which equals the total increased number of vehicle visits for nodes due to split delivery multiplied by the extra cost p_s. If split delivery is not allowed, the number of vehicle visits for the node j is one. Therefore, the increased number of vehicle visits for node j caused by split delivery equals the total number of vehicle visits for node j minus one. Constraint (3) ensures that the number of vehicle departures from the central warehouse equals the total number of vehicles at the central warehouse. Constraints (4)-(6) confirm that each route of the large vehicle starts from the central warehouse, ends at the central warehouse, and visits each node at most once. However, every node can be visited by different vehicles. Constraints (7)-(9) impose vehicle volume restrictions. Constraints (10)-(11) confirm that the demand of each node in this network can be satisfied. Due to the constraint (5), the node $j\in M\cup S\setminus\{0\}$ is visited at most once in the route of vehicle $k\in VF$ (i.e., there is at most one decision variable that is equal one among $x_{ij}^{k}, i\in M\cup S\setminus\{0\}$). Therefore, the arriving time of the vehicle takes the cumulative sum formulation in constraint (11). Constraint (13) ensures the arriving time before the delivery time. Constraints (14)-(16) are nonnegative and integrality constraints on decision variables.

In the above formulation of the FVRPOR, the store deliveries and the satellite deliveries are not treated in the same way. Since the volume of online order is different from the volume of product, we set the set of product type R and the set of packaging box (online order) type B separately. Besides, due to the hard time windows for stores and the soft time windows for satellites, the second term of the objective function (i.e, the penalty cost violating online time windows) only punishes the online order violating time windows, and the constraint (13) only ensures the arriving time of store nodes before the delivery time of store nodes.

4. Adaptive large neighborhood search

In practice, the mathematical model for the FVRPOR is not tractable, even if the size of the problem is reasonable. This vehicle routing problem (i.e., Eq. (2)) is a split delivery capacitated vehicle routing problem with multiple types of products, packaged online orders and multiple types of time windows. The framework of the ALNS introduced by Ropke and Pisinger (2006) is an extension of the large neighborhood search introduced by Shaw (1998), which is often used to solve many types of vehicle routing problems. Therefore, we propose a modified ALNS method to solve this vehicle routing problem.

The main idea of ALNS framework is to improve the solution at each iteration by employing multiple destroy and repair operators. Destroy and repair operators are the key components of ALNS framework, which can destroy the current solution and then rebuild it to obtain a new

solution. The general ALNS framework with simulated annealing is illustrated in Algorithm 1.

Algorithm 1: The framework of the ALNS

Initialization:
Set the iteration index $k \leftarrow 0$;
Initialize the solution G by Initial Solution process;
Set the optimal solution $G^* \leftarrow G$;
Loop:
Select a destroy method and a repair method from set Ω^d and set Ω^r based on their weights;
Generate a new solution G' by employing the selected destroy operator and repair operator for the solution G;
Updating the solution G based on the simulated annealing acceptance criterion;
If $f(G) < f(G^*)$:
Updating the optimal solution $G^* \leftarrow G$;
Updating the scores of set operator Ω^d and Ω^r based on an adaptive acceptance criteria;
Set $k \leftarrow k + 1$;
Until the termination condition can be met.

Let G, G' and G^* denotes the current solution, new solution and optimal solution for the FVRPOR. $f(G)$ denotes the objective value under the solution G. Ω^d denotes a set of destroy operators, and Ω^r denotes a set of repair operators. To avoid getting trapped into a local optimal solution, the acceptance criteria based on simulated annealing is used for the new solution after destroy and repair operators (Chen et al., 2021), where the temperature T is initialized by a constant value T_{st} and is linearly decreased towards to zero by $T = \beta T$, the cooling rate $\beta \in (0, 1)$. Instead of setting appropriate the initial temperature T_{st}, this parameter is calculated by using the cost of initial solution G_o (i.e., $T_{st} = hf(G_o)$, h is the start temperature control parameter) (Chen et al., 2021; Ropke and Pisinger, 2006). Besides, the weights of operators are updated by an adaptive criteria at the end of each segment, where a segment is a number of iterations of ALNS (Max_{seg}). The termination condition is that the ALNS stops after a fixed number of iterations Max_{iter}. The weight of operator o in segment m is updated by the following equation:

$$w_o^{m+1} = (1-\gamma)w_o^m + \gamma \frac{\pi_o^m}{\mu_o^m} \tag{17}$$

where w_o^m is the weight of operator o in segment m, μ_o^m is the number of times that the operator o is employed in segment m, the reaction factor $\gamma \in [0, 1]$. π_o^m is the sum of scores obtained by operator o in segment m. Similar to Ropke and Pisinger (2006) and Chen et al. (2021), the operator has three scores (i.e., σ_1, σ_2 and σ_3). When the new solution is accepted, there scores (i.e., σ_1, σ_2 and σ_3) are used under $f(G') < f(G^*)$, $f(G') < f(G)$ and $f(G') \geq f(G)$, respectively.

4.1. Initial solution process

In this subsection, we propose two methods to generate the initial solution. The first method is that we solve the relaxed problem (the problem Eq. (2) without considering the integer constraints Eq. (15) and (16)) by using the commercial solver and then use the solution of the relaxed problem as the initial solution. Another one is using a greedy insertion method to generate the initial solution, similar to Wen et al. (2016) and Bodin et al. (1978). There are three features in this initial solution process: (1) the nodes of stores are firstly selected to insert into routes and after that the nodes of satellites are selected to insert into routes; (2) the nodes selection are sorted in non-decreasing order of the delivery constraints l_i; (3) when the total volume of demand of the selected node exceeds the remaining volume capacity of vehicle, the split delivery for this node is allowable. The detail of the second initial solution method is described by Algorithm 2, where $\overline{V}_i$ denotes the total volume of demands for node i and z_ϑ denotes the residual volume of route ϑ.

Algorithm 2. The initial solution method

Initialization:
Sort set of store S in non-decreasing order of delivery constraints l_i;
Sort set of satellite M in non-decreasing order of delivery constraints l_i;
Construct a sorted node set $NS = S \cup M$, where the store nodes are placed before the satellite nodes;
Loop:
Sequentially select a node i from the set NS;
While $\overline{V}_i > 0$:
Find the feasible position of route ϑ for the node i from vehicle set V_l, which leads to the minimum increase of the objective function;
If $z_\vartheta > \overline{V}_i$:
$\overline{V}_i \leftarrow 0$, $z_\vartheta \leftarrow z_\vartheta - \overline{V}_i$;
else:
$\overline{V}_i \leftarrow \overline{V}_i - z_\vartheta$, $z_\vartheta \leftarrow 0$;
Insert the node i into the feasible position of route ϑ;
Remove the node i from the set NS;
Until the node set NS is empty.

Note that the sorted nodes of stores are inserted into the feasible positions in the routes, which leads to the minimum increase of the objective function. For the node of stores, the feasible position needs to make sure that the time constraints of the selected node and existing nodes in the route must be satisfied after inserting into this position. For the node of satellites, the feasible position denotes the one that can ensure that the delivery time constraints of the existing nodes of stores in the route are still satisfied after inserting the satellite node. In the two methods, the first one is time-consuming. Therefore, we first employ the second method to obtain the initial solution and will employ the first one only if the second one cannot obtain the feasible initial solution. Besides, although the initial solution generated by the two methods ignores the integer constraint, the destroy and repair operators of the ALNS can ensure the integer property of split quantities of products and packaged online orders after a large number of iterations.

4.2. Destroy operators

Different destroy operators defined in this section are used to remove a number λ of nodes of the current solution. To diversify the search process, we construct a random number $\lceil \eta^\rho E \rceil$, where η is a random variable between zero and one, a determinism parameter ρ is used to control the degree of randomness, and E denotes the number of removable nodes under the current solution G. The $\lceil \eta^\rho E \rceil$ th node on the sorted list based on those destroy operators is selected to remove. Finally, we can obtain a removal set Ψ including λ removal nodes. Inspired by some related work (i.e., Shaw (1998), Ropke and Pisinger, (2006), and Chen et al. (2021)), we present the seven destroy operators, where the first five operators are relatedness removal, the sixth operator is the random removal and the seventh operator is the worst removal.

1. **Improved Shaw's removal**: Inspired by Shaw (1998), we proposed a destroy operator based on the conventional Shaw removal, in order to suit this problem. The main idea of this operator is still to remove a predetermined number λ of relatedness nodes. The relatedness between node i and node j is measured by $RM(i,j)$ as follows

$$RM(i,j) = \varphi_1 d_{ij} + \varphi_2 \left|\overline{V}_i - \overline{V}_j\right| + \varphi_3 \left|l_i - l_j\right| + \varphi_4 \left|\Gamma_i - \Gamma_j\right| + \varphi_5 SN_{ij} \tag{18}$$

The relatedness measure $RM(i,j)$ includes five terms: the distance term, the volume term, the time window term, the number of split delivery term and the node type term. $\overline{V}_i$ is the total volume of demands for node i. Γ_i is the number of split deliveries for the node i. SN_{ij} equals zero if the type of node i is the same as node j and equals one otherwise. φ_1, φ_2, φ_3, φ_4 and φ_5 are the weights for those terms respectively. We assume that d_{ij}, $\overline{V}_i$ and l_i are normalized, such that $RM(i,j) \in [0, \varphi_1 + \varphi_2 + \varphi_3 + \varphi_4 + \varphi_5]$. Smaller $RM(i,j)$ denotes higher relatedness between node i and node j.

The detailed process of this operator is that (1) randomly remove a node from the current solution to the removal set Ψ; (2) randomly select a node j from the removal set Ψ; (3) obtain the sorted node list under the current solution G by increasing the relatedness measure $RM(i,j)$, $i \in G$; (4) generate a random number η, and move the $\lceil \eta^{\rho} E \rceil$ th node under the sorted list from the current solution G to the removal set; (5) repeat the steps (2)-(4) until the removal set includes λ nodes.

2. **Distance removal**: In the FVRPOR, there are two types of nodes (i.e., store and satellite) and different types of nodes have different features. Therefore, in this destroy operator, the relatedness between node i and node j is measured only by the distance d_{ij} and the node type SN_{ij}. The distance removal is actually the improved *Shaw's* removal with $\varphi_2 = \varphi_3 = \varphi_4 = 0$
3. **Volume removal**: In this destroy operator, the total volume of $\overline{V}_i$ and the node type SN_{ij} are used to measure the relatedness of nodes. The improved *Shaw's* removal with $\varphi_1 = \varphi_3 = \varphi_4 = 0$ is this operator
4. **Time window removal**: In this destroy operator, the delivery time l_j and the node type SN_{ij} are used to measure the relatedness of nodes. The improved *Shaw's* removal with $\varphi_1 = \varphi_2 = \varphi_4 = 0$ is this operator
5. **Split delivery removal**: Similar to the first four operators, the split delivery's number Γ_i and the node type SN_{ij} are used to measure the relatedness of nodes. The improved *Shaw's* removal with $\varphi_1 = \varphi_2 = \varphi_3 = 0$ is this operator
6. **Random removal**: The simple operator removes λ nodes from the current solution randomly and adds the removal nodes to the removal set. The operator mainly is used to increase the diversity of the search
7. **Worst removal**: The main idea of this operator is to remove the node with the most influence on the objective function. We define the influence function $C(i,G) = f(G) - f(G/i)$, where $f(G)$ denotes the objective function under current solution G and $f(G/i)$ denotes the objective function under the solution G without node i. The detailed process of this operator is that: (1) obtain the sorted node list under the current solution G by descending influence value function $C(i,G)$; (2) generate a random number η, and move the $\lceil \eta^{\rho} E \rceil$ th node under the sorted list from the current solution G to a removal set; (3) repeat the steps (1) and (2) until the removal set includes λ nodes

4.3. Repair operators

The destroy operator has removed λ of nodes from the current solution to form the removal set Ψ. In this section, we present three repair operators to rebuild the partial solution. The repair operators are mainly used to reinsert the nodes of removal set into the partial solution one by one, in order to generate a new current solution. Notably, since there are split integer constraint and multiple products and online orders in the FVRPOR, we need to consider how to allocate those products and online orders.

1. **Greedy repair**: The greedy repair operator performs λ iterations to reinsert the nodes in the removal set into the solution. In each iteration, the node in the removal set with the minimum insertion cost is selected to be reinserted at the corresponding position of the current solution. For each selected node, there are two subproblems to solve: (1) find the optimal feasible inserting positions and the corresponding allocated volumes for the node; (2) based on the inserting positions and allocated volumes, allocate multiple products or online orders

For the first subproblem, let $\Delta RC(i,\vartheta)$ denote the minimum change of objective cost incurred by inserting node i into route ϑ at the feasible position. The feasible position of route ϑ for node i has two features: (1) there is the residual volume in route (i.e., the total volume of original products and online orders in route is less than the vehicle capacity); (2) after reinserting the node at the position, the delivery time constraints of stores in the route ϑ need to be satisfied. Let $\Delta RC(i,G)$ denote the minimum change of objective cost incurred by inserting node i into solution G. Due to the split delivery, $\Delta RC(i,G) \neq \Delta RC(i,\vartheta)$, $\vartheta \in G$, we need to solve the following optimization problem to obtain $\Delta RC(i,G)$ (i.e., subproblem 1).

Subproblem 1:

$$\Delta RC(i,G) = \min \sum_{\vartheta \in G} \Delta RC(i,\vartheta)\eta_i^{\vartheta} + p_s\left(\sum_{\vartheta \in G} \eta_i^{\vartheta} - 1\right) \tag{19}$$

subject to

$$\sum_{\vartheta \in G} \eta_i^{\vartheta} z_{\vartheta} \geqslant \overline{V}_i \tag{20}$$

$$\eta_i^{\vartheta} \in \{0,1\} \forall \vartheta \in G \tag{21}$$

where η_i^{ϑ} is a binary variable that equals 1 if the node i is inserted into route ϑ and z_{ϑ} denotes the residual volume of route ϑ. By solving the optimization problem, we can obtain the optimal insertion route set G_i^* (i.e., $\eta_i^{\vartheta} = 1$, $\vartheta \in G_i^*$) and the allocated volumes for node i are z_{ϑ}, $\vartheta \in G_i^*$. Further, we discuss the computational requirements for solving subproblem 1. By observing subproblem 1 and considering the number of routes is limited to $|VF|$, there are $2^{|VF|} - 1$ (i.e., $C_{|VF|}^1 + C_{|VF|}^2 + \cdots + C_{|VF|}^{VF}$) combinations for the decision variable η_i^{ϑ} in Subproblem 1. Therefore, when the value of $|VF|$ is small, the enumeration method is sufficient to solve subproblem 1 as we also preferred.

Subproblem 2 is to allocate multiple products or online orders for node i under the given allocated volumes. Since we only need to obtain a feasible solution about allocating multiple products or online orders into the given volumes, this problem can described by the following constraint problem:

Subproblem 2: Find the feasible solution satisfying

if the node i is store node

$$\sum_{\vartheta \in G_i^*} \overline{Q}_i^{r\vartheta} = D_i^r \forall r \in R \tag{22}$$

$$\sum_{r \in R} \overline{Q}_i^{r\vartheta} v_r \leqslant z_{\vartheta} \forall \vartheta \in G_i^* \tag{23}$$

if the node i is satellite node

$$\sum_{\vartheta \in G_i^*} \overline{Y}_i^{b\vartheta} = D_i^b \forall b \in B \tag{24}$$

$$\sum_{b \in B} \overline{Y}_i^{b\vartheta} v_b \leqslant z_{\vartheta} \forall \vartheta \in G_i^* \tag{25}$$

where $\overline{Q}_i^{r\vartheta}$ is the nonnegative integer variable indicating the quantity of product type r allocated by route ϑ for the node i, $i \in S$, $r \in R$, $\vartheta \in G_i^*$. $\overline{Y}_i^{b\vartheta}$ is the nonnegative integer variable indicating the quantity of online order type b allocated by route ϑ for the node i, $i \in M$, $b \in B$, $\vartheta \in G_i^*$. Constraints (22) or (24) ensure that the demands of store node or satellite node are satisfied. Constraints (23) or (25) ensure that the allocated multiple products or online orders satisfy the given volumes z_{ϑ}. Actually, the solution of subproblem 2 is a feasible solution of a classical bin packing problem. However, we can find that the small limited allocated volume (i.e., z_{ϑ}, $\vartheta \in G_i^*$) from subproblem 1 leads to the unfeasible solution of subproblem 2. Therefore, we need to reformulate subproblem 1 by adding a new constraint:

$$\sum_{\vartheta \in G} \eta_i^{\vartheta} z_{\vartheta} > \sum_{\vartheta \in G_i^*} z_{\vartheta} \tag{26}$$

where the right side of constraint (26) denotes the total allocated volume obtained by solving the original subproblem 1. When adding

constraint (26) into the current subproblem 1, the total allocated volume obtained by solving the current subproblem 1 is larger than the result obtained by solving the original subproblem 1. Further, we solve the current subproblem 1 to obtain a new optimal solution. Based on the new optimal solution, we resolve subproblem 2 until obtaining a feasible solution.

Based on the above analysis, the details of the operator are described by Algorithm 3.

Algorithm 3. Greedy repair

Initialization:
Set the iteration index $k \leftarrow 0$;
Loop:
For all node $i \in \Psi$ and route $\vartheta \in G$:
Computing $\Delta RC(i, \vartheta)$;
For all node $i \in \Psi$:
Computing $\Delta RC(i, G)$ to obtain the optimal insertion route set G_i^*;
Solve the optimization problem $\min_{i \in \Psi} \Delta RC(i, G)$ to obtain the optimal node i^*;
Solve the subproblem 2 for the node i^* based on $G_{i^*}^*$;
While the solution of subproblem 2 does not exist:
Add the new constraint (26) to subproblem 1;
Solve the new subproblem 1 to obtain new $G_{i^*}^*$;
Solve the subproblem 2 based on the new $G_{i^*}^*$;
Insert the node i^* into the corresponding position of solution G;
Remove the node i^* from the removal set Ψ;
Until the removal set Ψ is empty.

2. **Greedy repair with random noise**: A random version for the above operator can smash the myopic of this operator (i.e., the selected node and inserted positions are not best locally). Therefore, when computing the minimum change $\Delta RC(i, \vartheta)$, we generate a random number ξ in the interval $[-Rn, Rn]$ and add it to the minimum change to obtain a new minimum change $\Delta RC'(i, \vartheta) = \Delta RC(i, \vartheta) + \xi$. Since the distance d_{ij} has an important influence on the objective function, we define the limit of random number $Rn = \sigma \max_{ij} d_{ij}$, where σ is a parameter to control the amount of random number. Other details of the operator are similar to the greedy repair operator
3. **Regret repair**: Based on the greedy repair operator, the regret repair operator employs a look-ahead information to alleviate the myopia of the greedy repair operator and improve solution quality. Let $\Delta RC^1(i, G)$ and $\Delta RC^2(i, G)$ denote the minimum and second minimum change of objective cost incurred by inserting node i into solution G, where $\Delta RC^1(i, G)$ is the optimal value of subproblem 1 (i.e., eqs. (19)-(21)). The second minimum change $\Delta RC^2(i, G)$ is the optimal value of subproblem 1 with the following additional constraint

$$\sum_{\vartheta \in G} \eta_i^{\vartheta} \eta_i^{\vartheta^*} - \sum_{\vartheta \in G} \eta_i^{\vartheta} \left(1 - \eta_i^{\vartheta^*}\right) \leqslant \sum_{\vartheta \in G} \eta_i^{\vartheta^*} - 1 \tag{27}$$

where $\eta_i^{\vartheta^*}$ is the optimal solution of subproblem 1 (i.e., the optimal solution of $\Delta RC^1(i, G)$). The constraint (27) can remove the optimal solution from subproblem 1. Then, we can obtain the second minimum change $\Delta RC^2(i, G)$ by computing the following optimization problem:

$$\Delta RC^2(i, G) = \min \sum_{\vartheta \in G} \Delta RC(i, \vartheta) \eta_i^{\vartheta} + p_s \left(\sum_{\vartheta \in G} \eta_i^{\vartheta} - 1 \right) \tag{28}$$

subject to (20), (21) and (27).

It is noted that before computing $\Delta RC^2(i, G)$, we first need to obtain the optimal solution of subproblem 1 (i.e., $\eta_i^{\vartheta^*}$) by solving the subproblem 1. Then based on the given $\eta_i^{\vartheta^*}$, by solving the problem (28) we can obtain the second minimum change $\Delta RC^2(i, G)$. Therefore, the decision variable of this optimization problem is only η_i^{ϑ} and the adding constraint (27) is actually linear.

Further, we define a regret value $\Delta RR(i, G) = \Delta RC^2(i, G) - \Delta RC^1(i, G)$ (the difference cost of the minimum and second minimum change of objective cost incurred by inserting node i into solution G). In each iteration, the regret value is used to select the inserted node, where the selected node is $\arg\max_{i \in \Psi} \Delta RR(i, G)$. Then the selected node needs to be inserted at its minimum cost position. Similar to the greedy repair operator, this operator still needs to consider how to allocate multiple products or online orders for node i under the given allocated volumes (i.e., subproblem 2). The details of greedy repair operator are similar to Algorithm 3. However, compared with Algorithm 3, the selection of inserted node in the greedy repair operator depends on the $\arg\max_{i \in \Psi} \Delta RR(i, G)$.

5. Numerical experiments

This section provides some numerical experiments to evaluate the performance of our proposed modified ALNS method. We also conduct sensitivity analysis for the impact of key parameters to reveal some managerial insights. First, we describe the generation of the test instances and how to tune the parameters of the proposed method. Finally, the results of the numerical experiments are presented. This method is implemented in Python 3.9 with the solver Gurobi 9.5.1 on a PC with an Intel I7-8700 CPU and 16.0 GB RAM.

5.1. Instance generation

There are no available benchmark instances for our studied FVRPOR. Therefore, we need to create benchmark instances to test our proposed method. In all instances, the X and Y coordinates of stores or satellites are randomly generated in the range $[1, 50]$, where the X and Y coordinates of the central warehouse equal zeros. The service time of node o_j equals 10. The delivery time of node l_j is randomly generated in the range $[60, 150]$. We set $p_u = p_s = 10$. The demand of store or satellite D_j^r or D_j^b is randomly created between 10 and 20. By investigating the sizes of packaging boxes in China, the basic volume of packaging boxes is nearly 30cm $\times$ 15cm $\times$ 10cm, which can be taken as the volume unit. Due to the limitations of urban traffic regulations and conditions, the vehicle transporting products and online orders generally can load 500 $\sim$ 1000 products or online orders with basic volume. Therefore, the capacity of vehicle V^* is fixed to 500 vol units. Since different types of products or packaged online orders have different volumes, the volume of product type r (v_r) or online order type b (v_b) is generated in $[1, 10]$. The number of vehicle $|VF|$ is set to $\left\lceil \left(\sum_{r \in R} \sum_{j \in S} v_r D_j^r + \sum_{j \in M} \sum_{b \in B} v_b D_j^b \right) / V^* \right\rceil$. For simplicity, we assume that the number of product types equals the number of online order types (i.e., $|R| = |B| \in [3, 10]$), and the distance d_{ij} equals the time t_{ij}. For the satellite node, we set the number of satellite node $|M| = |\delta N|$, where N denotes the number of all nodes and $\delta \in [0.3, 0.7]$. We randomly select $|M|$ nodes from all generated nodes as satellite nodes.

5.2. Parameter tuning

Based on the setting of instance, we generate 5 instances with different numbers of nodes to tune the parameter of the modified ALNS method (Instance 1 with 5 nodes, Instance 2 with 10 nodes, Instance 3 with 15 nodes, Instance 4 with 20 nodes and Instance 5 with 25 nodes). We employ the methodology presented by Ropke and Pisinger (2006) and Chen et al (2021) to set the main parameters of this method. When tuning a main parameter, we allow the parameter to take several values, while others are kept fixed. Besides, for each instance, the *meta*-heuristic method is applied to test the instance five times and we select the best setting based on the average value of those five solutions.

For our proposed *meta*-heuristic method, all parameters used in this method are: (1) the acceptance criteria's parameters (i.e., h and β); (2) the ALNS framework parameters (i.e., Max_{iter}, Max_{seg}, γ, σ_1, σ_2 and σ_3);

(3) the operators' parameter (i.e.,λ, ρ, φ_1, φ_2, φ_3, φ_4, φ_5 and σ). The number of removed nodes λ is a crucial parameter for the proposed modified ALNS method. The smaller λ may lead to the suboptimal solution, while the larger λ may fail to obtain a better solution from the current solution and take higher computation effort. Therefore, we randomly select a value from the range $\left[\underline{\lambda}, \overline{\lambda}\right]$,$\underline{\lambda} = \varepsilon_1 N$ and $\overline{\lambda} = \varepsilon_2 N$ each iteration. We only need to determine the parameters ε_1 and ε_2 instead of specifying λ. After tuning those parameters, we list those tuning parameters along with their values in Table 3.

5.3. Performance of the modified ALNS method

To the best of our knowledge, due to jointly considering some realistic features (i.e, multiple types of products/online orders, multiple types of time windows, split delivery and split integer constraint), our studied problem has not been researched in existing related literature, which leads that the proposed methods in the related literature cannot solve our studied problem. Moreover, since our proposed method based on the framework of the ALNS is customized to our studied problem, the proposed modified ALNS can solve our studied problem effectively and efficiently. Further, to study the performance of the proposed method, we compare the results obtained by the proposed method and the exact solution based on the commercial solver (i.e., Gurobi). Table 4 shows the comparisons between the modified ALNS method and Gurobi for different sizes of instances. The instances having different sizes are randomly generated based on the parameters in Section 5.1. Without losing reasonability, we set a 3600-second time limit as the maximum computation time of Gurobi. T_{ALNS} denotes the computation time of the proposed method and T_G denotes the computation time of the commercial solver Gurobi. TC_{ALNS} denotes the objective function values obtained by the proposed method and TC_G denotes the objective function values obtained by the commercial solver Gurobi. The gap of objective function value Gap_{AG} is defined by $Gap_{AG} = 100 \times (TC_{ALNS} - TC_G)/TC_G$.

As shown in Table 4, the performance of Gurobi is related to the characteristics of data. Since the studied problem is a mixed integer programming problem, the commercial solver Gurobi employs a branch-and-cut framework combined with lots of heuristic methods to solve it, which leads to the quality of solution depending on how to branch integer decision variables. Especially for large-scale problems, Gurobi usually cannot find a feasible solution under the limited computation time. However, our proposed method is more stable compared with Gurobi. In terms of solution quality, the solutions obtained by our proposed method are better than those obtained by Gurobi in most of instances. Only in some instances with extremely less types of products and online orders (2 types of products and online orders), the quality of the solution obtained by the proposed method is similar to the solution obtained by Gurobi, but Gurobi cannot obtain feasible solutions for those instances under the same computation time as our proposed method (i.e., 42 s for Instance 7, 101 s for Instance 13 and 207 s for Instance 19). Especially, for large-scale problems, our proposed method can obtain a high-quality solution. In terms of computation time, no matter what scale the instance is, the computation time of the proposed method is much shorter than Gurobi. Therefore, the proposed method is more efficient compared with Gurobi, and it can obtain a high-quality near-optimal solution in a much shorter computation time.

Table 3
Values for parameters after tuning.

Description	Value
The start temperature control parameter h	0.01
The cooling rate β	0.9999
Total number of iteration Max_{iter}	2000
The number of iteration in each segment Max_{seg}	100
The reaction factor γ	0.1
The scores of solution (σ_1, σ_2, σ_3)	(16, 4, 8)
The higher limit number of removal nodes $\overline{\lambda}$	$0.5N$
The lower limit number of removal nodes $\underline{\lambda}$	$0.3N$
The parameter controlling the degree of randomness in removal operations ρ	9
The first weight for improved Shaw removal φ_1	1
The second weight for improved Shaw removal φ_2	1
The third weight for improved Shaw removal φ_3	1
The fourth weight for improved Shaw removal φ_4	6
The fifth weight for improved Shaw removal φ_5	1
The parameter controlling the random noise σ	0.002

Further, we conduct a specific analysis of the computation time for different sizes of instances. Fig. 2 shows the computation time for different sizes of instances. As shown in Fig. 2, the computation time of our proposed method increases dramatically with the increase of the sizes of instances. Firstly, since our problem is aimed at the first echelon distribution network FVRPOR (the transportation of products from a central warehouse to the satellites and the stores), the number of nodes is the sum of the number of stores and satellites without considering the customer node, which leads to that the number of nodes is not particularly large. Secondly, the results obtained by running the proposed method at once are daily planned vehicle routes for omnichannel retailers. Therefore, the computation time of our proposed method is still a reasonable time for our studied problem.

5.4. Sensitivity analysis

In this subsection, we conduct some numerical experiences based on the instance setting of Section 5.1. Fig. 3a illustrates the objective values of instances with different values of vehicle capacity V^*. From Fig. 3a, with the increase of vehicle capacity V^*, the objective value (total cost) decreases. The reason is that the increase of vehicle capacity implies that the number of nodes serviced by every vehicle increases, which leads to a decrease of total distance. Fig. 3b illustrates the computation time of instances with different values of vehicle capacity V^*. From Fig. 3b, with the increase of vehicle capacity V^*, the computation time decreases. The reason is that with the increase of vehicle capacity, the number of vehicles scheduled by the proposed method reduces, which leads to a decrease of the computation time. Besides, we can find that the vehicle capacity V^* has a greater impact on the large-scale problem (i.e., N=25) compared with the other problems. Therefore, the vehicle with a large capacity is helpful for the delivery of products under omnichannel retailing.

In our studied FVRPOR, there are two types of time windows (hard time windows for products and soft time windows for packaged online orders). To investigate the sensitivity analysis of the time window, we first set three scenarios with different widths of time windows (i.e., Scenario 1: $l_j \in [30, 75]$, Scenario 2: $l_j \in [42, 105]$ and Scenario 3: $l_j \in [60, 150]$) and then vary the unit penalty cost p_u for the three scenarios. We randomly generate an instance with $N = 20$ and $V^* = 750$ based on the instance setting of Section 5.1 and then conduct the sensitivity analysis of time windows based on this instance. Fig. 4 illustrates the objective value and computation time with different values of the unit penalty cost p_u under different scenarios. From Fig. 4a, the objective value decreases with the increase of widths of time windows. The reason is that widening time windows means weakening time window constraints, which leads to more combinations of vehicle routes. The increase of vehicle route combinations can help to improve the objective value. Besides, we can find that only under the tight time windows (Scenario 1), the unit penalty cost p_u has a greater impact on the objective value. Under tight time windows, the amount of satellites violating the corresponding soft time windows increases, which leads to a significant impact under tight time windows. Under this scenario, with the increase of the unit penalty cost p_u, the objective value increases. From Fig. 4b, neither the time window nor the parameter p_u have a significant impact on the computation time. Therefore, the change of the

Table 4
Performance of the proposed modified ALNS.

Instance number	N	$\|R\|$	$\|VF\|$	Gurobi		ALNS		Gap_{AG}
				$T_G(sec)$	TC_G	$T_{ALNS}(sec)$	TC_{ALNS}	
1	5	2	4	1	206.68	11	206.68	0.00
2	5	3	4	3	272.02	12	272.02	0.00
3	5	4	6	24	365.85	14	365.85	0.00
4	5	5	6	3600	540	17	462	−14.44
5	5	6	9	3600	502.49	17	462.09	−8.04
6	5	7	10	3600	744.65	19	607.77	−18.38
7	10	2	5	3600	438.82	42	441.41	0.59
8	10	3	9	3600	623.88	55	601.13	−3.65
9	10	4	13	3600	984.19	81	977.53	−0.68
10	10	5	15	3600	1158.17	92	1038.71	−10.31
11	10	6	16	3600	1200.55	89	1082.50	−9.83
12	10	7	20	3600	1474.59	101	1349.53	−8.48
13	15	2	7	3600	648.544	101	656.31	1.20
14	15	3	11	3600	–	165	1116.12	–
15	15	4	16	3600	1436.73	191	1418.72	−1.25
16	15	5	20	3600	1817.67	229	1786.15	−1.73
17	15	6	24	3600	2211.19	210	1910.18	−13.61
18	15	7	28	3600	–	240	2105.09	–
19	20	2	10	3600	818	207	836	2.20
20	20	3	14	3600	–	311	1302.26	–
21	20	4	21	3600	1821.06	370	1800.23	−1.14
22	20	5	25	3600	–	382	1988.27	–
23	20	6	28	3600	–	322	2022.74	–
24	20	7	36	3600	–	402	2411.73	–

−: No feasible solution or Gap_{AG} found in 3600 s.

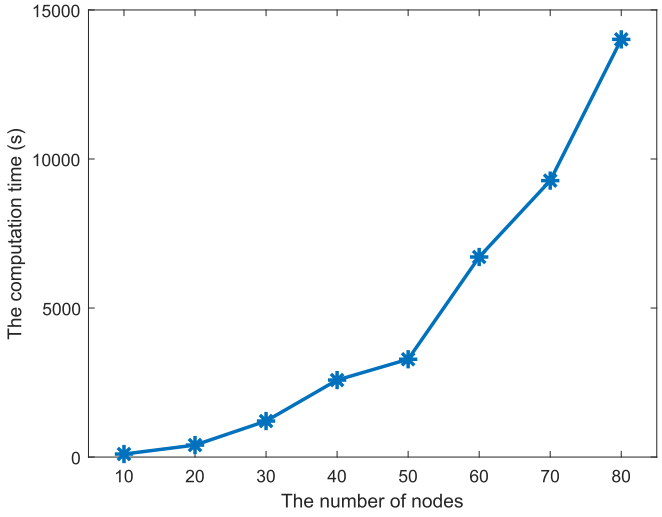


Fig. 2. The computation time for different sizes of instances.

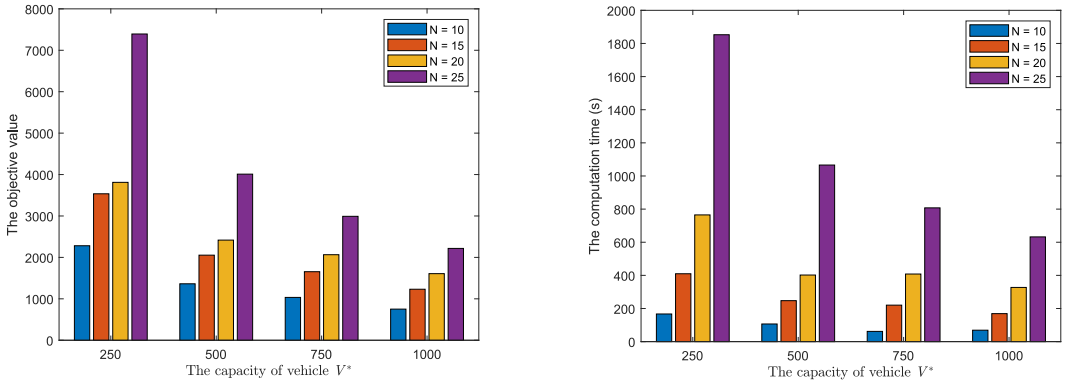
time window and the parameter p_u cannot increase the complexity of our proposed method.

Similar to the sensitivity analysis of parameter p_u, we conduct the sensitivity analysis of the split delivery cost p_s. Firstly, an instance with $N = 20$, $V^* = 750$ and $p_u = 0.1$ is generated based on the instance setting of Section 5.1 and then the sensitivity analysis of the split delivery cost p_s is conducted based on this instance under the three scenarios with different widths of time windows (i.e., Scenario 1, Scenario 2 and Scenario 3). Fig. 5 illustrates the objective value and computation time with different values of the penalty cost p_s under different scenarios. From Fig. 5, we can find that with the increase of the penalty split delivery cost p_u, the objective value increases while the computation time remains basically unchanged. Besides, similar to the unit penalty cost p_u, the penalty split delivery cost p_s also has a greater impact on the objective value under tight time windows (Scenario 1). However, compared with the unit penalty cost p_u, the objective value is more sensitive to the penalty split delivery cost p_s under the three scenarios.

6. Conclusion

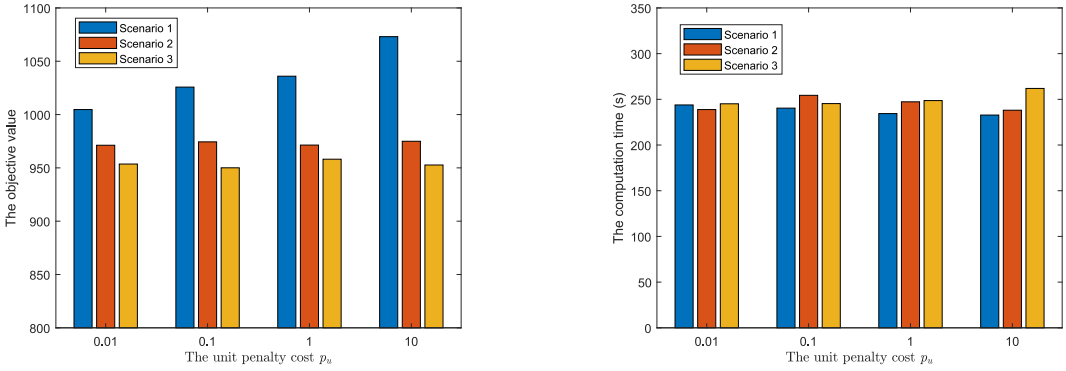
In this paper, we study the first echelon of VRPOR (FVRPOR) with multiple types of time windows (hard time windows and soft time windows) and multiple types of products simultaneously. Due to introducing omnichannel retailing, our studied problem jointly considers some features: (1) multiple products or packaged online orders have different types of volumes; (2) split delivery and integer constraints for the split delivery quantities of products are considered; (3) the time window of product is a hard constraint while the time window of online order is a soft constraint, based on the characteristics of different purchasing channels. Therefore, our constructed model is more complex but more practicable than existing models in related literature. Further, to cope with the high complexity of this constructed model, we propose a modified adaptive large neighborhood search method to solve this problem. Especially, based on the features of vehicle routing for omnichannel retailing, the destroy and repair operations of the searching method are modified and designed to adapt to our studied problem. Compared with the commercial solver Gurobi 9.5.1, the proposed method can obtain a high-quality near-optimal solution in a much shorter computation time.

Based on the results of our numerical experiments, we provide some managerial insights. On the one hand, the volume of vehicle has a greater impact on the FVRPOR. The larger volume of vehicle is beneficial to reduce the total cost. Especially for the large-scale vehicle routing problem, there are more cost reductions. On the other hand, widening time windows facilitates the route planning of vehicles, which can improve the quality of vehicle routes. Under tight time windows, the two penalty cost parameters (i.e., the unit penalty cost of violating online order time windows p_u and the split delivery cost p_s) have significant influences on the total cost. Therefore, omnichannel retailers need to focus on the change of the two parameters under tight time windows. However, when widening time windows, the influence of the unit penalty cost violating online order time windows is much weaker than the influence of the split delivery cost.



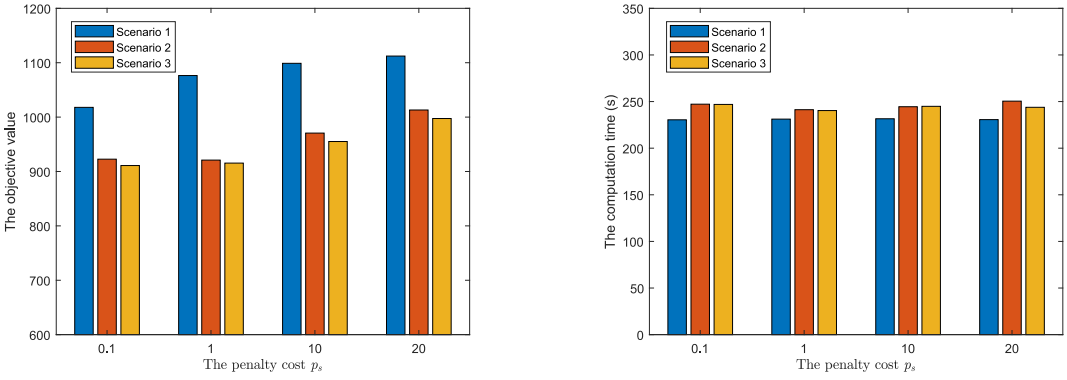
(a) The objective value (b) The computation time

Fig. 3. The objective values and computation time of instances with different values of V^*.



(a) The objective value (b) The computation time

Fig. 4. The objective value and computation time of instance with different values of p_u under different scenarios.



(a) The objective value (b) The computation time

Fig. 5. The objective value and computation time of instance with different values of p_s under different scenarios.

Our studied problem has some limitations. In our model, the demand of product or online order is deterministic. Although we consider the limited volume of vehicle during transportation of products and online orders, the volume and load capacity of vehicle jointly need to be taken into consideration in practice. For future work, we would like to extend our studied problem by considering those limitations.

CRediT authorship contribution statement

Ning Li: Writing – original draft, Methodology. **Zheng Wang:** Writing – review & editing.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgment

This research is supported by National Natural Science Foundation of China (Grant No. 61673109).

References

Abdulkader, M.M.S., Gajpal, Y., EIMekkawy, T.Y., 2018. Vehicle routing problem in omni-channel retailing distribution systems. Int. J. Prod. Econ. 196, 43–55.
Akyüz, M.H., Muter, İ., Erdoğan, G., Laporte, G., 2022. Minimum cost delivery of multi-item orders in e-commerce logistics. Comput. Oper. Rese. 138, 105613.
Arslan, A.N., Klibi, W., Montreuil, B., 2021. Distribution network deployment for omnichannel retailing. Eur. J. Oper. Res. 294 (3), 1042–1058.
Bayliss, C., Martins, L.C., Juan, A.A., 2020. A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem. Comput. Ind. Eng. 148, 1–14.
Bianchessi, N., Drexl, M., Irnich, S., 2019. The split delivery vehicle routing problem with time windows and customer inconvenience constraints. Transp. Sci. 53 (4), 1067–1084.
Bodin, L., Rosenfield, D., Kydes, A., 1978. UCOST: a micro approach to a transportation planning problem. J. Urban Anal. 5 (1).
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.
Desaulniers, G., 2010. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. Oper. Res. 58 (1), 179–192.
Dönmez, S., Koç, Ç., Alt1parmak, F., 2022. The mixed fleet vehicle routing problem with partial recharging by multiple chargers: Mathmematical model and adaptive large neighborhood search. Transport. Res. E Logist. Transport. Rev. 167, 1–23.
Fan, H., Zhang, Y., Tian, P., Lv, Y., Fan, H., 2021. Time-dependent multi-depot green vehicle routing problem with time windows considering temporal-spatial distance. Comput. Oper. Rese. 129, 105211–105224.
Gao, R., Ma, Y., Ralescu, D.A., 2023. Uncertain multilevel programming with application to omni-channel vehicle routing problem. J. Ambient Intell. Human. Comput. 14, 9159–9171.
Guo, C., Thompson, R.G., Foliente, G., Kong, X.T.R., 2021. An auction-enabled collaborative routing mechanism for omnichannel on-demand logistics through transshipment. Transport. Res. E Logist. Transport. Rev. 146, 1–22.
Huang, M., Du, B., Guo, J., 2023. A hybrid collaborative framework for integrated production scheduling and vehicle routing problem with batch manufacturing and soft time windows. Comput. Oper. Rese. 159, 1–17.
Li, J., Qin, H., Baldacci, R., Zhu, W., 2020. Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. Transport. Res. E Logist. Transport. Rev. 140, 1–22.
Li, J., Cai, J., Sun, T., Zhu, Q., Lin, Q., 2023. Multitask-based evolutionary optimization for vehicle routing problems in autonomous transportation. IEEE Trans. Autom. Sci. Eng. 1–12.
Li, N., Wang, Z., 2023. Inventory control for omnichannel retailing between one warehouse and multiple stores. IEEE Trans. Eng. Manage. 1–18.
Liu, P., Hendalianpour, A., Feylizadeh, M., Pedrycz, W., 2021. Mathematical modeling of vehicle routing problem in omni-channel retailing. Appl. Soft Comput. 131, 1–25.
Liu, X., Kim, S.W., Kwon, C., 2023. An adaptive large neighborhood search method for rebalancing free-floating electric vehicle sharing systems. Comput. Oper. Rese. 155, 1–21.
Luo, Z., Qin, H., Zhu, W., Lim, A., 2017. Branch and price and cut for the split-delivery vehicle routing problem with time windows and linear weight-related cost. Transp. Sci. 51 (2), 668–687.
Pan, B., Zhang, Z., Lim, A., 2021a. Multi-trip time-dependent vehicle routing problem with time windows. Eur. J. Oper. Res. 291 (1), 218–231.
Pan, B., Zhang, Z., Lim, A., 2021b. A hybrid algorithm for time-dependent vehicle routing with time windows. Comput. Oper. Rese. 128, 1–12.
Paul, J., Agatz, N., Spliet, R., Koster, R.D., 2019. Shared capacity routing problem – an omni-channel retail study. Eur. J. Oper. Res. 273 (2), 731–739.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp. Sci. 40 (4), 393–546.
Schubert, D., Kuhn, H., Holzapfel, A., 2021. Same-day deliveries in omnichannel retail: integrated order picking and vehicle routing with vehicle-site dependencies. Nav. Res. Logistics 68 (6), 721–744.
Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems. In: Proceedings If the 4th International Conference on Principles and Practice of Constraint Programming. Springer, New York, pp. 417–431.
Sluijk, N., Florio, A.M., Kinable, J., Dellaert, N., Van Woensel, T., 2023. Two-echelon vehicle routing problems: a literature review. Eur. J. Oper. Res. 12 (5), 1967.
Song, X., Jones, D., Asgari, N., Pigden, T., 2020. Multi-objective vehicle routing and loading with time window constraints: a real-life application. Ann. Oper. Res. 291, 799–825.
Wang, Z., Wen, P., 2020. Optimization of a low-carbon two-echelon heterogeneous-fleet vehicle routing for cold chain logistics under mixed time window. Sustainability 304 (3), 865–886.
Wen, M., Linde, E., Ropke, S., Mirchandani, P., Larsen, A., 2016. An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem. Comput. Oper. Rese. 76, 73–83.
Wu, S., Bo, H., Jin, C., Liu, X., 2024. Nested column generation for split pickup vehicle routing problem with time windows and time-dependent demand. Comput. Oper. Rese. 164, 1–18.
Yan, S., Chu, J.C., Hsiao, F.Y., Huang, H.J., 2015. A planning model and solution algorithm for multi-trip split-delivery vehicle routing and scheduling problems with time windows. Comput. Ind. Eng. 87, 383–393.
Yang, J., Li, Y., 2023. A multicommodity pickup and delivery problem with time windows and handling time in the omni-channel last-mile delivery. Int. Trans. Oper. Res. 1–42.
Yu, C., Zhang, D., Lau, H.Y.K., 2017. An adaptive large neighborhood search heuristic for solving a robust gate assignment problem. Expert Syst. Appl. 84 (30), 143–154.
Zhang, K., He, F., Zhang, Z., Lin, X., Li, M., 2020. Multi-vehicle routing problems with soft time windows: a multi-agent reinforcement learning approach. Transport. Res. Part c: Emerg. Technol. 121, 1–14.
Zhao, J., Dong, H., Wang, N., 2023. Green split multiple-commodity pickup and delivery vehicle routing problem. Comput. Oper. Rese. 159, 1–11.
Žulj, I., Kramer, S., Schneider, M., 2018. A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem. Eur. J. Oper. Res. 264 (2), 653–664.