Electronic Research Archive

http://www.aimspress.com/journal/ERA

ERA, 33(4): 2668–2697.
DOI: 10.3934/era.2025118
Received: 23 February 2025
Revised: 15 April 2025
Accepted: 18 April 2025
Published: 28 April 2025

Research article

Optimizing routing for autonomous delivery and pickup vehicles in three-dimensional space

Qi Hong[1], Hongyi Zhao[1], Shiyu Chen[1], Aya Selmoune[2] and Kai Huang[3,4,*]

[1] School of Transportation, Southeast University, Nanjing 210096, China
[2] The Cho Chun Shik Graduate School of Mobility, Korea Advanced Institute of Science and Technology, South Korea
[3] School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China
[4] Wuxi Campus, Southeast University, Wuxi 214000, China

* **Correspondence:** Email: kaihuang@seu.edu.cn.

Abstract: The last-mile delivery challenge in three-dimensional (3D) multi-floor building environments has a significant impact on logistics efficiency. Although autonomous delivery robots (ADRs) have been widely adopted to address last-mile logistics, most existing studies focus on optimizing ADR routing in simplified two-dimensional environments. Moreover, optimal layout of goods within the robot's physical containers brings another challenge. Hence, this paper formalizes the Discrete Capacity Vehicle Routing Problem with Simultaneous Delivery-Pickup and Soft Time Windows (DCVRP-SDP-STW) in a 3D environment. To achieve high-quality solutions, we propose an improved ant colony optimization algorithm that considers spatiotemporal multi-stage clustering characteristics, leading to a significant reduction in computation time. A data preprocessing framework is also developed to convert real-world architectural topologies into navigable 3D routing networks. To validate the proposed model and algorithm, we conducted a case study in Nanjing. The results show that the algorithm can improve optimization outcomes by 23% to 94% compared to pre-optimization results based on the proximity principle, which can contribute to the advancement of efficient and intelligent autonomous delivery systems.

Keywords: last-mile delivery; urban logistics; spatiotemporal clustering; hybrid algorithm; vehicle routing problem

1. Introduction

With the rapid development of the logistics industry, the demand for express delivery is growing significantly creating challenges in the last mile of logistics, such as difficulty in effectively coordinating terminal distribution resources with dispersed, small-batch, and high-frequency consumer demand in urban areas [1,2]. Autonomous delivery robots (ADRs) are innovative tools to address this last-mile problem by reducing the freight cost [3,4]. ADRs can utilize facilities including staircases and elevator networks to achieve door-to-door delivery across multi-level urban infrastructures [5].

The vehicle routing problem (VRP) framework is introduced to solve the last-mile logistics problem using ADRs. By optimizing routes, VRP reduces delivery distances and minimizes customer waiting times [6]. Many existing studies have applied VRP into real delivery tasks [7,8]. Most focus on trucks deliveries while overlooking the advantages of ADRs, which can offer door-to-door service. A distinguishing feature of ADRs is their handling of goods with 3D characteristics (length, width, and height). Wang et al. considered this feature in their modeling constrains, but this resulted in a complex two-stage model [9]. However, for campus deliveries, package sizes are standard. Thus, ADRs containers can be split into smaller cells designed for the standard package sizes. This design can help merchants quickly load goods and allows customers to pick up their items from designated containers [10]. In this case, the best way is to transform 3D spatial constraints into one-dimensional matching constraints, thereby reducing model complexity [11].

The ability of door-to-door delivery provides a strong foundation for extending the VRP framework into 3D environments in real campus settings [12]. The complexity of real-world 3D delivery scenarios, characterized by multi-building distributions and vertical floor transitions, presents significant operational challenges [13]. Compared with the previous studies involving two-dimensional environments, usually using Euclidean or Manhattan distances [14], we enhance the 3DCityNet architecture through a rigorous 3D spatial parameterization [15]. Our algorithm generates a 3D routing matrix that simultaneously accounts for: horizontal street-level networks, vertical circulation systems (elevators/stairwells), and house layout in the same floor.

In this paper, we propose an innovative method to fulfill goods delivery. ADRs are employed in multi-floor buildings to offer door-to-door service in the campus. This paper introduces the Discrete Capacity Vehicle Routing Problem with Simultaneous Delivery-Pickup and Soft Time Windows (DCVRP-SDP-STW) model, which can capture the features of a real-world ADR container. Given the NP-hard nature of the VRP, managing large-scale instances remains a significant challenge [16,17]. This paper proposes the Spatiotemporal Multi-Stage Clustering Tabu Ant Colony Search Algorithm with Temporal Pheromones (SMC-TACSATP) to efficiently obtain high-quality solutions. The contributions of this paper are as follows:

- We propose a novel VRP variant that considers both supply and demand features. This model introduces a discrete capacity to account for the physical characteristics of ADRs containers.
- We propose the SMC-TACSATP algorithm for large-scale delivery scenarios. Such a method can provide high-quality solutions within a limited time frame by utilizing a clustering approach to capture spatiotemporal characteristics and proposing an improved ant colony optimization (ACO) algorithm.
- We conduct a case study using map data. An algorithm is developed to calculate 3D distances between points, and the dummy node method is introduced to enable split delivery, which, for the

first time, applies VRP into last-mile delivery of goods in urban 3D buildings.

The remainder of this paper is organized as follows: Section 2 reviews related literature. Section 3 presents the optimization model and methods for its simplification, while Section 4 describes the solution algorithm. Section 5 provides a case study, demonstrating the application and advantages of the proposed model and solution algorithm. Finally, Section 6 concludes the paper.

2. Literature review

One of the main challenges in this paper is how to optimize the VRP under complex constraints [18,19]. The first one is modelling the VRP with distinct good sizes in vehicles. A classic variation is the capacitated vehicle routing problem (CVRP), which can be divided into two types. The first type, CVRP, was put forward by Clark and Wright, and assumed identical vehicles with equal capacity [20]. The second type heterogeneous fleet VRP, proposed by Quak and de Koster, which considers varying the types of vehicles [21]. Lin et al. and Luiz Usberti et al. considered trucks and trailers as part of the fleet [22]. In the past decade, VRP involving loading constraints for real containers has gained huge attention regarding the sizes (e.g., length, width, and height) of goods [9].

The second challenge is to consider customer-specific time windows in the VRP model. Solomon first considered the VRP with time window (VRPTW) [23]. When vehicles arrive before the desired time, they must wait; conversely, arriving late is prohibited. Moreover, Balakrishnan set soft time windows, which allow arriving late at an additional cost, thus adding flexibility to routing decisions [24]. Guertin et al. extended this concept by transitioning from static to dynamic scheduling under soft time window constraints [25]. Hashimoto et al. contributed a dynamic programming framework for situations involving convex and non-convex penalties [26].

Another significant modeling challenge lies in involving both deliveries and pickups. To handle such a problem, Angelelli and Mansini considered the simultaneous pickup and delivery, enabling a vehicle to achieve pickup and delivery tasks within a single trip [27]. Goetschalckx and Jacobs-Blecha considered VRP with backhauls, which simplifies the problem by enforcing a strict sequence where all deliveries precede pickups [28].

Finding the optimal solution for the VRP is challenging due to its NP-hard nature. One of the foundational methods was introduced by Clark and Wright, which focuses on combining individual routes after ensuring all operational constraints are satisfied [20]. Altabeeb et al. also introduced a firefly algorithm to tackle capacity constraints [29]. For VRP with time window, Gambardella et al. were the first to successfully apply the ACO algorithm to VRPTW [30]. Wang et al. further enhanced this approach by designing a hybrid multi-ant system to solve the VRPTW with service time constraints [30]. Zhang et al. integrated time penalties into the updating process of pheromone [31]. A typical VRP model, which is similar to our model for ADRs, also addressed this model using ACO [32].

The VRP in a 3D environment is a newly-raised problem. Door-to-door delivery belongs to a 3D vehicle routing problem, in which ADRs travel between buildings. Traditional VRP models simply assumed that the nodes are in the same plane and used time or fuel consumption as the weight between nodes [33]. To overcome this limitation, Thill et al. put forward 3DCityNet, which integrates an object-oriented GIS model to seamlessly connect indoor and outdoor transportation networks [15].

How to solve large-scale optimization is a challenging topic [34]. Recent studies show promise in combining machine learning models with the heuristic algorithm to solve the VRP efficiently [35]. To solve VRP with time window models, Wang et al. incorporated time information into the distance metric and proposed the 3D k-means approach [36]. Erdogan and Miller-Hooks used the DBSACN clustering algorithm to partition nodes and quickly generate an initial solution [37].

The following Table 1 lists the literature discussed in this paper, and the methods used.

Table 1. Summary of related literatures.

Literature	Capacity	Time-window	SDP	3D environment
Solomon MM [23]		√		
Balakrishnan N [24]		√		
Gambardella LM, et al. [30]		√		
Gendreau M, et al. [25]		√		
Angelelli E, et al. [27]		√	√	
Hashimoto H, et al. [26]		√		
Quak H, de Koster MB [21]				√
Thill et al. [15]				√
Erdogan and Miller-Hooks [37]	√	√		
Mańdziuk J, Nejman C [18]	√			
Reil S, et al. [17]	√	√		
Altabeeb AM, et al. [29]	√			
Wang Y, et al. [31]		√		
Zhang H, et al. [32]		√		
Wang Y, et al. [9]	√	√		√
Zheng X, et al. [33]			√	
Wang Y. et al. [37]	√	√	√	
Ours	√	√	√	√

3. Problem description

This paper aims to minimize the weighted total distance, number of vehicles, and time window penalties.

The operational area is a campus environment with $|I|$ nodes, comprised by a logistics center and multiple customer nodes. The travel time between node $i \in I$ and node $j \in I$ is denoted by g_{ij}. Among these nodes, we define P as the subset with pick-up demands and D as those requiring deliveries. There are $|V|$ vehicles in depot to conduct pick-up and delivery tasks. Each customer $i \in I$ sets a time window $[te_i, tl_i]$ where te_i is the earliest arriving time permitted and tl_i is the latest. If a vehicle's arriving time ta_i is not in the time window, a penalty w_i will be added to the objective function. It also considers the discrete capacity feature of ADRs. For goods in depot, we use $|S|$ to denote the number of sizes of goods, where the physical size of goods decreases as the index s increases. Considering the containers with different sizes, a vehicle owns f_t t-th-sized containers, where $t \in S$. A t-th-sized container can only store 1 t-th-sized good. We also allow cross-level storage, which means an $s-1$-th-sized container is capable of being loaded with n s-th goods. In order to identify the size of goods, we denote by cd_i^t the goods that need to be delivered in node $i \in I$ with the size $t \in S$, while cp_i^t is for pick-up. In the delivery process, md_i^t is the number of containers used for $t \in S$ size delivery container in node $i \in I$, and mp_i^t is the number of containers used for picking up goods. We optimize the decision variable X_{ij}^v, which is 1 if vehicle $v \in V$

passes the link from node $i \in I$ to node $j \in I$, and 0 otherwise. As shown in Figure 1, different vehicle colors represent different tasks, while the colors of the nodes indicate various demand types: red for delivery, blue for pick up, and green for combined delivery and pick up. The numbers denote the sequence of vehicle routing, which is also the output of our model.

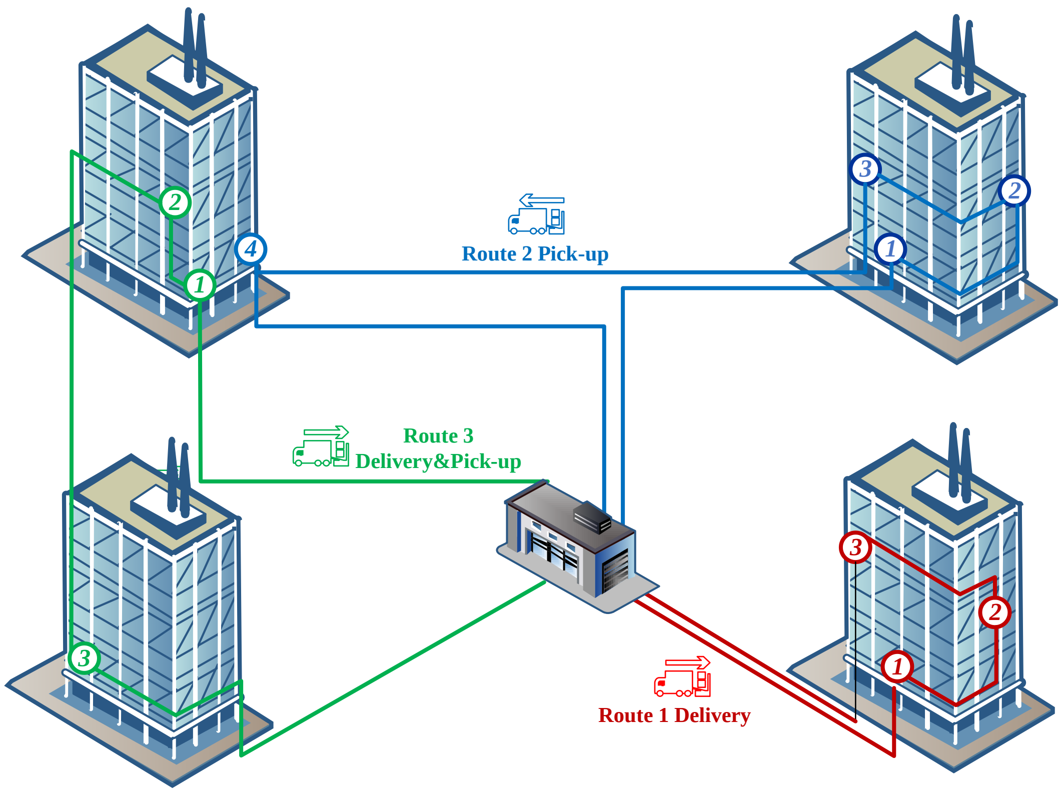


Figure 1. Example of the autonomous delivery and pickup vehicles in 3D environment.

3.1. Notation

Table 2 presents the notations in this model.

Table 2. Notation table of model.

Sets	
$I:\{i\}\{j\}$	Set of nodes, where i_0 represents the logistics center, and $i_1, i_2, i_3 \ldots i_{\lvert I \rvert}$ represent the customer location to be served
$P:\{p\}$	Set of nodes, which is a subset of the I set, represents the nodes with pick up needs
$D:\{d\}$	Set of nodes, which is a subset of the I set, represents the nodes with drop-off needs
$V:\{v\}$	Set of vehicles
$S\{s\}$	Set of sizes of goods
$CD:\{cd_i^t\}$	Set of goods states, 1 if node i is delivered to t size goods, 0 otherwise
$CP:\{cp_i^t\}$	Set of goods states, 1 if node i picks up t size goods, 0 otherwise
$F:\{f_t\}$	Set of maximum number of cells for $t \in S$ size container in one vehicle
Parameters	
c_f	Vehicle fixed cost
c_s	Vehicle travel cost per time unit
c_t	Customer waiting cost
g_{ij}	Travel time from i node to j node
te_i	Allowed arrive time in advance to i node
tl_i	Allowed delayed arrive time to i node
n	The number of level s (smaller) goods that can be accommodated by one level $s-1$ (larger) container
Decision variables	
X_{ij}^v	1 if vehicle v travels from node i to node j, 0 otherwise
Y_i^v	1 if the v vehicle serves the demand of node i, 0 otherwise
$\mu_i^e, \nu_i^e, \mu_i^l, \nu_i^l$	Binary variables involved in the linearization of the time penalty functions
Auxiliary variable	
ta_i	Time of the actual arrival of i node
vn	Number of vehicles activated
w_i	The penalty cost when vehicle does not reach the desired arrival time of i node
md_i^t	The number of containers used for t size delivery container in i node
mp_i^t	The number of containers used for t size pick up container in i node

3.2. *Mathematical model*

- **P1: Delivery-Only Model**

In this scenario, all goods are stored at the depot and must be delivered to various customer locations. This model is applicable in situations where a logistics center serves multiple customers with no return goods.

- **P2: Pick-Up-Only Model**

All goods are collected from customer locations and transported back to the logistics center. This model is applicable in situations where customer locations are primarily suppliers or the focus is on returning goods to a central location.

- **P3: Combined Delivery and Pick-Up Model**

This model combines the scenarios of P1 and P2, where goods are both delivered from the depot to customers and collected from customers back to the logistics center. It is a more complex situation where vehicles handle both deliveries and pick-ups simultaneously.

3.2.1. Delivery model

The objective function (1) aims to minimize the weighted total distance, number of vehicles, and time window penalties.

$$\textbf{P1} \ \min_{X_{ij}^{v}} p = \sum_{v=0}^{|V|}\sum_{i=0}^{|I|}\sum_{j=0}^{|I|} c_s \cdot g_{ij} \cdot X_{ij}^{v} + c_f \cdot vn + \sum_{i=1}^{|I|} w_i \tag{1}$$

Subject to:

$$X_{ij}^{v} \in \{0,1\}, \qquad \forall i, j \in I, \forall v \in V \tag{2}$$

$$Y_i^{v} = \sum_{i=0}^{|I|} X_{ij}^{v}, \qquad \forall j \in I \setminus i_0, \forall v \in V \tag{3}$$

$$\sum_{j=0}^{|I|} X_{0j}^{v} = 1, \qquad \forall v \in V \tag{4}$$

$$\sum_{i=0}^{|I|} X_{i0}^{v} = 1, \qquad \forall v \in V \tag{5}$$

$$X_{ii}^{v} = 0, \qquad \forall i \in I \setminus i_0, \forall v \in V \tag{6}$$

$$vn = |V| - \sum_{v=1}^{|V|} X_{00}^{v} \tag{7}$$

$$\sum_{v=1}^{|V|}\sum_{i=0}^{|I|} X_{ij}^{v} = 1, \qquad \forall j \in I \setminus \{i_0\} \tag{8}$$

$$\sum_{v=1}^{|V|}\sum_{j=0}^{|I|} X_{ij}^{v} = 1, \qquad \forall i \in I \setminus \{i_0\} \tag{9}$$

$$\sum_{i \in I, i \neq h} X_{ih}^{v} = \sum_{j \in I, j \neq h} X_{hj}^{v}, \qquad \forall v \in V, \forall h \in I \tag{10}$$

$$ta_j \geq ta_i + g_{ij} + ts_i - (1 - X_{ij}^{v}) \cdot T_{\max}, \qquad \forall i \in I, \forall j \in I \setminus \{i_0\}, i \neq j, \forall v \in V \tag{11}$$

$$w_i = \max\{0, \alpha(te_i - ta_i)\} + \max\{0, \beta(ta_i - tl_i\}, \qquad \forall i \in I \setminus \{i_0\} \tag{12}$$

$$\sum_{t=1}^{s}\sum_{i=0}^{|I|} n^{s-t} \cdot Y_i^{v} \cdot cd_i^{t} \leq \sum_{t=1}^{s} n^{s-t} \cdot f_t, \qquad \forall v \in V, \forall s \in S \tag{13}$$

Constraints (2) and (3) define the decision variables. Constraint (4) ensures that all vehicles leave from a fixed depot. Constraint (5) requires that all vehicles return to the depot. Constraint (6) indicates that the vehicles cannot travel to the same place. Constraint (7) calculates the number of activated vehicles.

$\sum_{v=1}^{|V|} X_{00}^{v}$ represents the vehicles that were not activated, as they returned directly to the depot after leaving. Constraints (8) and (9) require that each node can only receive one service from one vehicle per application. It is important to note here that Constraints (8) and (9) are not incompatible with split delivery of discrete containers, as the dummy node technique will be introduced below, and this step will deal with discrete goods as multiple virtual nodes so that they can be delivered in a manner that satisfies Constraints (8) and (9) to satisfy a customer's split delivery. Constraint (10) ensures that vehicles must leave the same node after visiting it. Constraint (11) is used to restrict the time to reach node j. Constraint (12) defines the penalties incurred when a vehicle fails to meet the customer's requested time window. Constraint (13) ensures that the number of goods of different sizes loaded on the robot while starting from depot cannot exceed the capacity of the different size containers of the robot. n^{s-t} represents how many $s-th$ goods the $t-th$ containers can accommodate. Thus, the left side of the inequality (13) calculates how many equivalent $s-th$ goods have been loaded onto vehicle v. The right side of the inequality (13) calculates the maximum amount of equivalent $s-th$ goods that the vehicle can load.

3.2.2. Pick up model

The model in which all goods are requested to be delivered to the depot is formulated as follows:

$$\textbf{P2} \ \min_{X_{ij}^{v}} p = \sum_{v=0}^{|V|}\sum_{i=0}^{|I|}\sum_{j=0}^{|I|} c_s \cdot g_{ij} \cdot X_{ij}^{v} + c_f \cdot vn + \sum_{i=1}^{|I|} w_i \tag{14}$$

Constraints (2)–(12), and

$$\sum_{t=1}^{s}\sum_{i=0}^{|I|} n^{s-t} \cdot Y_i^{v} \cdot cp_i^{t} \le \sum_{t=1}^{s} n^{s-t} \cdot f_t,, \qquad \forall v \in V, s \in S \tag{15}$$

Constraint (15) ensures that the number of goods of different sizes loaded on the robot while coming back to depot cannot exceed the capacity of containers. The principle is consistent with Constraint (13).

3.2.3. Delivery and pick up model

The model in which all goods are sent from or to the depot, usually expressed as One-to-Many-to-One Single Vehicle Pickup and Delivery Problem, is formulated as follows:

$$\textbf{P3} \ \min_{X_{ij}^{v}} p = \sum_{v=0}^{|V|}\sum_{i=0}^{|I|}\sum_{j=0}^{|I|} c_s \cdot g_{ij} \cdot X_{ij}^{v} + c_f \cdot vn + \sum_{i=1}^{|I|} w_i \tag{16}$$

Constraints (2)–(12), and

$$md_i^t + cd_i^t - (1 - X_{ij}^{v})M \le md_j^t \le md_i^t + cd_i^t + (1 - X_{ij}^{v})M, \qquad \forall t \in S, \forall v \in V, \forall i, j \in I, i \ne j \tag{17}$$

$$mp_i^t + cp_i^t - (1 - X_{ij}^{v})M \le mp_j^t \le mp_i^t + cp_i^t + (1 - X_{ij}^{v})M, \qquad \forall t \in S, \forall v \in V, \forall i, j \in I, i \ne j \tag{18}$$

$$\sum_{t=1}^{s} n^{s-t} \cdot (md_{|I|}^t - md_i^t) + \sum_{t=1}^{s} n^{s-t} \cdot mp_i^t \le \sum_{t=1}^{s} n^{s-t} \cdot f_t, \qquad \forall i \in I, s \in S \tag{19}$$

If the delivery and pick up tasks are mixed, the complexity of the problem will increase dramatically. It is necessary to check whether all nodes meet the capacity constraint. Constraints (17) and (18) calculate the number of goods to delivery and the number of goods to be picked up before this node. Figure 2 describes the concrete explanation of Constraint (19), which determines whether the capacity constraints for different sizes are satisfied for each node, by calculating the quantity of goods that still need to be delivered and the goods received earlier, summing them up to obtain the current load.

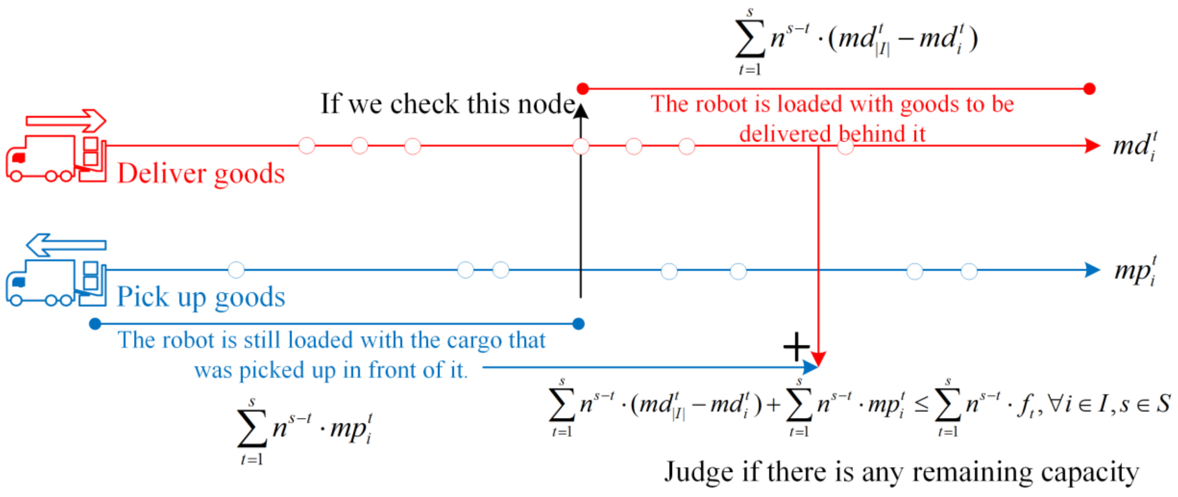

$\sum_{t=1}^{s} n^{s-t} \cdot (md_{|I|}^{t} - md_{i}^{t})$
If we check this node
The robot is loaded with goods to be delivered behind it
Deliver goods
md_i^t
Pick up goods
mp_i^t
The robot is still loaded with the cargo that was picked up in front of it.
$\sum_{t=1}^{s} n^{s-t} \cdot mp_i^t$
$\sum_{t=1}^{s} n^{s-t} \cdot (md_{|I|}^{t} - md_{i}^{t}) + \sum_{t=1}^{s} n^{s-t} \cdot mp_i^t \le \sum_{t=1}^{s} n^{s-t} \cdot f_t, \forall i \in I, s \in S$
Judge if there is any remaining capacity

Figure 2. The concrete explanation of Constraint (19).

3.3. *Revised model*

3.3.1. Model linearization

The non-linear Constraint (12) caused by calculating the time penalty function incurs a great computation challenge; as such, series of linearization methods are proposed as follows. Since this is a general linearization technique, we omit (ω) in all decision variables for simplicity of notation. Constraint (12) can be divided into two parts. The question is transformed into two subproblems.

$$w_{ci} = w_{ci}^{e} + w_{ci}^{l}, \qquad \forall i \in I \setminus \{i_0\} \tag{20}$$

$$w_{ci}^{e} = \max\{0, \alpha(t_{ei} - t_{ai})\}, \qquad \forall i \in I \setminus \{i_0\} \tag{21}$$

$$w_{ci}^{l} = \max\{0, \beta(t_{ai} - t_{li})\}, \qquad \forall i \in I \setminus \{i_0\} \tag{22}$$

Constraint (21) is replaced by Constraints (23)–(28) to achieve model linearization.

$$w_{ci}^{e} \ge 0, \qquad \forall i \in I \setminus \{i_0\} \tag{23}$$

$$w_{ci}^{e} \ge \alpha(t_{ei} - t_{ai}), \qquad \forall i \in I \setminus \{i_0\} \tag{24}$$

$$w_{ci}^{e} - M(1 - \mu_i^{e}) \le 0, \qquad \forall i \in I \setminus \{i_0\} \tag{25}$$

$$w_{ci}^{e} - M(1-\nu_i^e) \le \alpha(t_{ei} - t_{ai}), \qquad \forall i \in I \setminus \{i_0\} \tag{26}$$

$$\mu_i^e + \nu_i^e \ge 1, \qquad \forall i \in I \setminus \{i_0\} \tag{27}$$

$$\mu_i^e, \nu_i^e \in \{0,1\}, \qquad \forall i \in I \setminus \{i_0\} \tag{28}$$

Constraint (20) is first divided into two subproblems. When $\mu_i^e = 1$ and $\nu_i^e = 0$, Constraint (25) reduces to $w_{ci}^e \le 0$ and Constraint (26) is redundant. Together with Constraint (23), we have $w_{ci}^e = 0$ and Constraint (24) reduces to $t_{ei} \le t_{ai}$. This indicates that vehicle arrives at i node later than the earliest time. The penalty for being early is equal to 0. When $\mu_i^e = 0$ and $\nu_i^e = 1$, Constraint (26) reduces to $w_{ci}^e \le \alpha(t_{ei} - t_{ai})$ and Constraint (25) is redundant. Together with Constraints (23) and (24), we have $w_{ci}^e = \alpha(t_{ei} - t_{ai})$ and $t_{ei} \ge t_{ai}$. This represents that vehicle arrives at i node earlier than the earliest time. The penalty for being early is equal to $\alpha(t_{ei} - t_{ai})$. When $\mu_i^e = 0$ and $\nu_i^e = 0$, we have $w_{ci}^e = \alpha(t_{ei} - t_{ai})$ and $w_{ci}^e = 0$. This represents the arrival of the vehicle just at the earliest allowed arrival time.

For Constraint (22), via the same linearization method, Constraint (22) can be substituted by Constraints (29)–(34).

$$w_{ci}^{l} \ge 0, \qquad \forall i \in I \setminus \{i_0\} \tag{29}$$

$$w_{ci}^{l} \ge \beta(ta_i - tla_i), \qquad \forall i \in I \setminus \{i_0\} \tag{30}$$

$$w_{ci}^{l} - M(1-\nu_i^l) \le 0, \qquad \forall i \in I \setminus \{i_0\} \tag{31}$$

$$w_{ci}^{l} - M(1-\mu_i^l) \le \beta(ta_i - tla_i), \qquad \forall i \in I \setminus \{i_0\} \tag{32}$$

$$\mu_i^l + \nu_i^l \ge 1, \qquad \forall i \in I \setminus \{i_0\} \tag{33}$$

$$\mu_i^l, \nu_i^l \in \{0,1\}, \qquad \forall i \in I \setminus \{i_0\} \tag{34}$$

3.3.2. Method for calculating the minimum number of activated vehicles

The number of activated vehicles plays a vital role in objective function. During the solving process, solutions with a higher number of activated vehicles will be discarded due to the dominant cost of vehicle activation. To reduce the search space and simplify the objective function, this paper transforms the traditional optimization problem into the following bi-level optimization problem. Constraint (35) gives the upper model objective function.

$$\min_{X_{ij}^v} p = c_f \cdot vn \tag{35}$$

The result vn of optimizing the upper equation will be input into the lower model as the number of vehicles to optimize the lower equation.

$$\min_{X_{ij}^v} p = \sum_{v=0}^{|V|}\sum_{i=0}^{|I|}\sum_{j=0}^{|I|} c_s \cdot g_{ij} \cdot X_{ij}^v + \sum_{i=1}^{|I|} w_i \tag{36}$$

The bi-level programming model is complicated to solve. However, in this paper, the number of startup vehicles can be uniquely determined in the upper model. This process is well-suited for P1 and P2.

However, for P3, where certain cells can be used iteratively during the delivery process, we can only obtain a relatively approximate minimum number of vehicles. The method to calculate the minimum number of activated vehicles vn is shown in Algorithm 1.

Algorithm 1. Pseudo-code for calculating the activated vehicle

```
Input: $CD:\{cd_i^t\}\backslash CP:\{cp_i^t\}\backslash F:\{F_s\}\backslash n$
Output: $vn$
1: determine the corresponding model category:
       If P1: $c_i^t = cd_i^t$
       If P2: $c_i^t = cp_i^t$
       If P3: $c_i^t = cp_i^t + cd_i^t$
2: equivalent conversion of how many $s-th$ size of goods $h_s$ can be loaded in a vehicle at most, calculated as follows:
       $h_s = \sum_{t=1}^{s} n^{s-t} \cdot f_t \ , s = 1,2,3 \cdots |S|$    (37)
3: equivalent conversion of number of vehicles $l_s$ required for goods in the entire service network $s-th$ size is calculated as follows:
       $l_s = \sum_{t=1}^{s} (n^{s-t} \cdot \sum_{i=1}^{|I|} c_i^t), s = 1,2,3 \cdots |S|$    (38)
4: calculate the number of vehicles required to transport equivalent $s-th$ size goods:
       $q_s = \left\lceil \frac{l_s}{h_s} \right\rceil, s = 1,2,3 \cdots |S|$    (39)
5: calculate the number of vehicles needed to distribute all the goods:
       $vn = \max\{q_s, s = 1,2,3 \cdots |S|\}$    (40)
```

4. Solution algorithm

The DCVRP-SDP-STW is a typical NP-hard problem. To address the large-scale delivery VRP challenge, we propose an improved ACO algorithm as depicted in Figure 3. Dummy nodes are introduced to achieve split deliveries, and a complete distance graph is generated to consider the 3D feature of buildings. As a key component of the improved ACO algorithm, the SMC algorithm clusters the nodes based on both time window distribution and spatial location, which reduces computing time. To improve the solution diversity of ACO, we put forward three mechanisms.

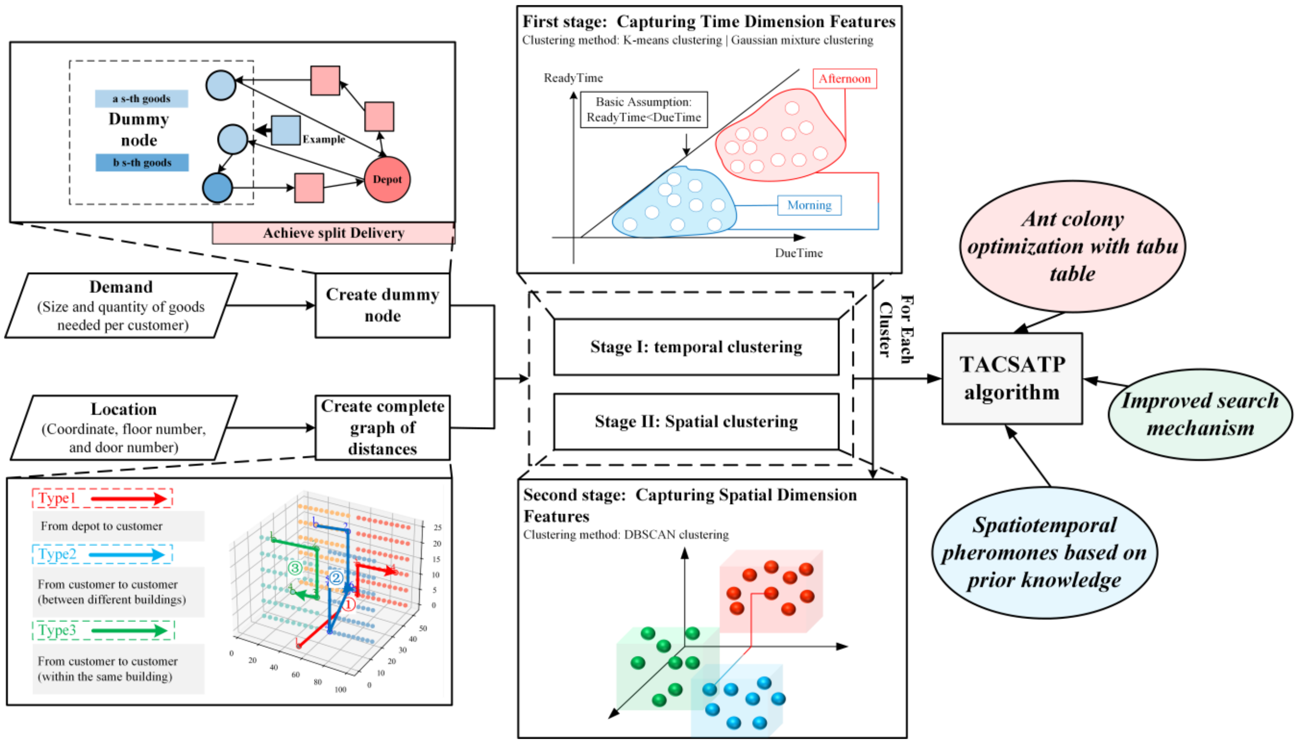

First stage: Capturing Time Dimension Features
Clustering method: K-means clustering | Gaussian mixture clustering
ReadyTime
Basic Assumption: ReadyTime<DueTime
Afternoon
Morning
DueTime
a s-th goods
Dummy node
b s-th goods
Example
Depot
Achieve split Delivery
Demand (Size and quantity of goods needed per customer)
Create dummy node
Location (Coordinate, floor number, and door number)
Create complete graph of distances
Stage I: temporal clustering
Stage II: Spatial clustering
For Each Cluster
TACSATP algorithm
Ant colony optimization with tabu table
Improved search mechanism
Spatiotemporal pheromones based on prior knowledge
Type1
From depot to customer
Type2
From customer to customer (between different buildings)
Type3
From customer to customer (within the same building)
Second stage: Capturing Spatial Dimension Features
Clustering method: DBSCAN clustering

Figure 3. Overall algorithm flowchart.

4.1. *Spatiotemporal multi-stage clustering*

We propose a SMC method, integrating spatiotemporal dimensions for more effective classification of customer points. As illustrated in Figure 4, the SMC approach divides the problem into two stages. In the first stage, we focus on temporal information, specifically the time window of each customer. By employing Euclidean distance, initial clustering is performed to divide the delivery process into multiple periods. This process creates a distance matrix, as described in Eq (41), to cluster the time feature. The second stage involves clustering based on spatial features, such as e (enclosure), f (floor), and d (door) information, which describe the customer's spatial characteristics. The distance function between two nodes is computed according to Eq (42), ensuring that both spatial proximity and structural features are considered. The proposed spatiotemporal SMC method can reduce computing time by effectively decomposing large-scale routing problems into smaller, manageable sub-problems.

$$T_{ij} = \sqrt{(te_i - te_j)^2 + (tl_i - tl_j)^2}, \forall i \in I, \forall j \in I \tag{41}$$

$$H_{ij} = \alpha | e_i - e_j | + \beta | f_i - f_j | + | d_i - d_j |, \forall i \in I, \forall j \in I \tag{42}$$

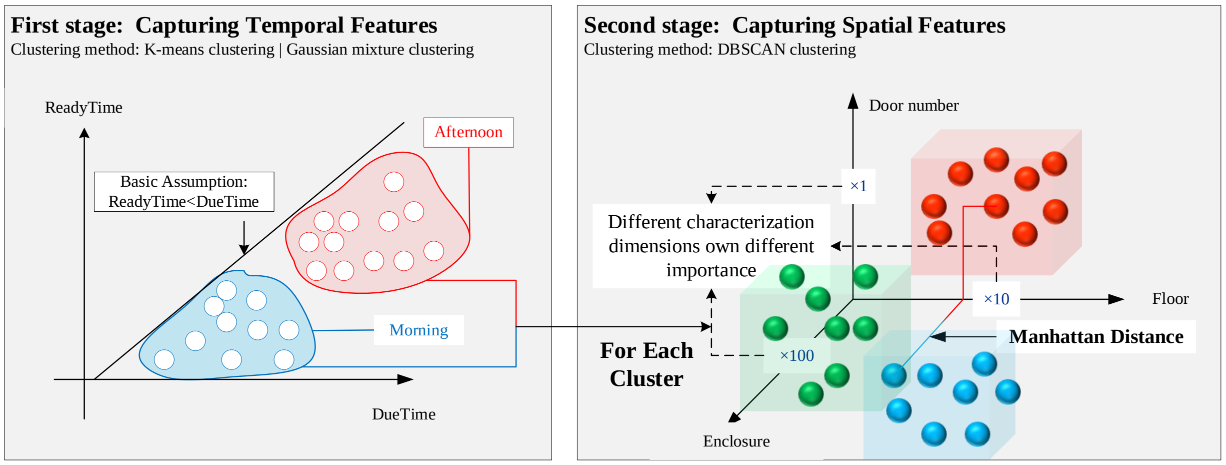

First stage: Capturing Temporal Features
Clustering method: K-means clustering | Gaussian mixture clustering
ReadyTime
Basic Assumption: ReadyTime<DueTime
Afternoon
Morning
DueTime
Second stage: Capturing Spatial Features
Clustering method: DBSCAN clustering
Door number
×1
Different characterization dimensions own different importance
×10
Floor
Manhattan Distance
For Each Cluster
×100
Enclosure

Figure 4. SMC algorithm.

4.2. *TACSATP algorithm*

This section introduces the details of the TACSATP algorithm. Algorithm 2 gives the pseudo-code. The TACSATP algorithm is an ACO-based metaheuristic algorithm. It employs multiple ants to iteratively construct solutions by dynamically managing capacity, cluster constraints, and node selections via a tabu table, while guiding search efficiency through spatiotemporal pheromone updates. The three subsections introduce three mechanisms used in this algorithm. To address the bi-level optimization problem, we propose a hierarchical solution update strategy that sequentially evaluates: 1) the number of activated vehicles as the primary criterion, and 2) the objective function value defined in Eq (36) as the secondary criterion.

4.2.1. Ant colony optimization with tabu table

In this section, we discuss the combination of ACO and TS. To alleviate the issue of ACO easily falling into local optima, we introduce a tabu table during the search process to maintain solution diversity by storing previously visited solutions. Unlike strict prohibition mechanisms, the tabu table in TACSATP implements a soft penalty mechanism, allowing some flexibility in revisiting solutions under certain conditions. Algorithm 3 illustrates the process of searching methods with tabu tables.

Algorithm 2. TACSATP algorithm

Input: $num\_iterations$, num_ants, *distance graph and demand of customers*, *clusters alternative set*

Output: optimal solution sol_{opt} and objective function value obj_{opt}

```
1:  for each iteration $iter \in \{1, \cdots, num\_iterations\}$ do
2:      if $iter$ =1: initialize optimal solution $sol_{opt}$, activated vehicles number $veh_{opt} = \inf$, and
        objective function value $obj_{opt} = \inf$
3:      initialize current solution $sol_{iter}$ ,activated vehicles number $veh_{iter} = \inf$, and objective
        function value $obj_{iter} = \inf$
4:      for each $ant \in \{1, \cdots, num\_ants\}$ do
5:          initialize capacity $cap$, clusters alternative set $all\_clusters$, nodes left $left\_node$, an
            empty routing $rou$, tabu table $tabu$, a solution $sol_{ant}^{iter}$ to store solution for ant
6:          randomly select a cluster $cur\_cluster$, remove it from $all\_clusters$ ,select a node
            $next\_node$ from $left\_node$
7:          while $node_{left}$ is not empty do
8:              if the Constraint (13)/(15)/(19) not be fitted after adding $next\_node$ into $rou_{ant}^{iter}$:
9:                  add $rou$ into $sol_{ant}^{iter}$, and refresh $rou$ into an empty routing
10:             else:
11:                 make $cur\_node = next\_node$, add $cur\_node$ into $rou$, update $cap$, remove
                    $cur\_node$ from $cur\_cluster$ and $left\_node$
12:                 if $cur\_cluster$ is empty:
13:                     select next node $next\_node$ from left nodes of all node set,
                        $cur\_cluster = Cluster\ which\ next\_node\ is\ located\ in$ , and remove
                        $cur\_cluster$ from $all\_clusters$
14:                 else: Select next node $next\_node$ and update $tabu$ based on Algorithm 3
15:         End while
16:         calculate the activated vehicles and objective function value $veh_{ant}^{iter}$ and $obj_{ant}^{iter}$ of $sol_{ant}^{iter}$
17:         if $veh_{ant} > veh_{ant}^{iter}$: Refresh $sol_{iter} = sol_{ant}^{iter}$, $veh_{ant} = veh_{ant}^{iter}$, and $obj_{iter} = obj_{ant}^{iter}$
18:         else if $veh_{ant} = veh_{ant}^{iter}$:
19:             if $obj_{iter} > obj_{ant}^{iter}$: Refresh $sol_{iter} = sol_{ant}^{iter}$, $veh_{ant} = veh_{ant}^{iter}$, and $obj_{iter} = obj_{ant}^{iter}$
20:     End for
17:     if $veh_{opt} > veh_{iter}$: Refresh $sol_{opt} = sol_{iter}$, $veh_{opt} = veh_{iter}$, and $obj_{opt} = obj_{iter}$
18:     else if $veh_{opt} = veh_{iter}$:
19:         if $obj_{opt} > obj_{iter}$: Refresh $sol_{opt} = sol_{iter}$, $veh_{opt} = veh_{iter}$, and $obj_{opt} = obj_{iter}$
22:     refresh the spatiotemporal pheromones based on Eq (44)
23: End for
Return $sol_{opt}$ and $obj_{opt}$
```

Algorithm 3. Searching methods with tabu tables

Input: current node cur_node, alternative point set $cur\_cluster$ (set of remaining points in the cluster), tabu table $tabu$, max

Output: next node $next\_node$

```
1:  for each node $node \in cur\_cluster$ do
2:      Calculate the arriving possibility $p_{node}$ according to (43)
    End for
3:  // Roulette Selection
    construct cumulative probability arrays $cum\_prob$:
4:      $cum\_prob[0] = p_{node_0}$
5:      for each node $node \in \{cur\_cluster \setminus node_0\}$ do
6:          $cum\_prob[i] = cum\_prob[i-1] + p_{node_i}$
        End for
7:  generate random numbers $r \in [0, 1)$
8:  find the smallest $i$ such that $cum\_prob[i] \geq r$
9:  $next\_node = node_i$
10: if $tabu_{cur\_node}^{next\_node} = 0$: set $tabu_{cur\_node}^{next\_node} = T$
11: else: $tabu_{cur\_node}^{next\_node} = tabu_{cur\_node}^{next\_node} - 1$ and return to *step 3*
Return $next\_node$
```

In the ACO process, the next node is selected using a roulette wheel method to introduce randomness, ensuring diverse path exploration. After selecting the next node, the algorithm checks the tabu table: if the node is new, its corresponding edge is initialized with an iteration countdown before release; if it already exists, the countdown is decremented by one. If the node remains restricted, the selection repeats until a valid node is found. This approach balances solution diversity and feasibility, prevents premature convergence by avoiding repetitive path revisits, and enhances overall search efficiency.

4.2.2. Improved search mechanism

In the TACSATP algorithm, two mechanisms are designed to generate routes.

Mechanism 1: It adapts the search process to the clustering method by changing the range of the candidate set. During route generation, a node is randomly selected from multiple clusters. Nodes within the same cluster are then selected until the cluster is exhausted, a process called intra-cluster node selection. After that, the process returns to global node selection to determine the next node from a different cluster. This approach allows the ACO to focus on node traversal within clusters, significantly reducing computational complexity.

Mechanism 2: To address the issue of potential capacity waste with discrete capacity containers, this mechanism introduces a lookahead feature. When a delivery robot's large capacity container cannot be fully utilized, the robot would return to the depot, leaving the small capacity container underutilized. The lookahead mechanism allows the search to consider multiple steps forward if capacity constraints are not met, thus improving the overall utilization rate of the robot's containers.

In Figure 5, the relationship between two mechanisms is reflected in how they interact during the search process: Mechanism 1 affects the route construction by structuring the node selection order, whereas Mechanism 2 affects decision-making by improving capacity utilization.

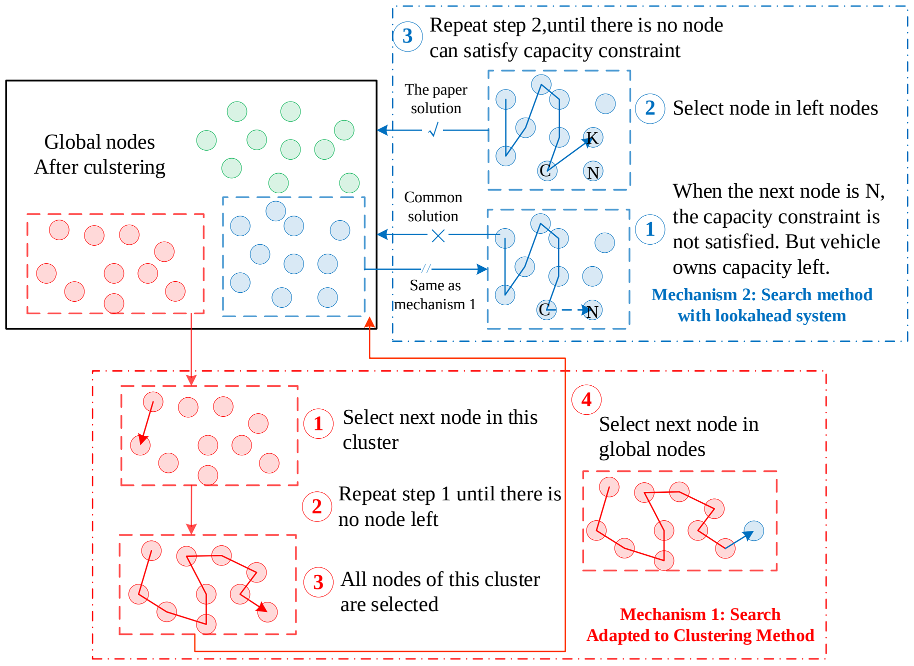

3 Repeat step 2,until there is no node can satisfy capacity constraint
The paper solution
√
2 Select node in left nodes
K
C
N
Global nodes After culstering
Common solution
×
When the next node is N, the capacity constraint is not satisfied. But vehicle owns capacity left.
1
Same as mechanism 1
Mechanism 2: Search method with lookahead system
1 Select next node in this cluster
2 Repeat step 1 until there is no node left
3 All nodes of this cluster are selected
4 Select next node in global nodes
Mechanism 1: Search Adapted to Clustering Method

Figure 5. Illustration of improved search mechanism.

4.2.3. Spatiotemporal pheromones based on prior knowledge

In the process of constructing the route selection, we propose the search method of long- and short-time pheromone. The specific search method is shown in Eq (43), where the time pheromone is calculated as Eq (44) . In this way, the time and distance penalty relationship can be jointly considered during the search process, allowing for the capture of spatiotemporal relationships.

$$p_{ij} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}(\phi_{ij})^{\gamma}}{\sum_{u \notin tabu_l}(\tau_{iu})^{\alpha}(\eta_{iu})^{\beta}(\phi_{iu})^{\gamma}}, & if\ \ j \notin tabu_l \\ 0, & elsewise \end{cases} \tag{43}$$

$$\phi_{ij} = \phi_{ij}^{(local)} \cdot \phi_{ij}^{(gobal)} \tag{44}$$

To consider the local information and a priori information in the search process, our approach is divided into two parts. The first part captures points with the current time window penalty, referred to as short-time pheromones, shown in Eq (45). z_{ij}^{l} represents the time from i node to j node for the l ant. The second part introduces a priori knowledge from previous iterations to capture long-term time penalty relationships named global time pheromone. The global time pheromone computation process is illustrated in Figure 6. In order to give ambiguity to vehicle arrivals to improve the performance of the solution, this paper constructs the possibility of vehicle arrivals in Eq (46).

$$\phi_{ij}^{(local)} = \frac{1}{\max(1, z_{ij}^{l} - tl_j, te_j - z_{ij}^{l})} \tag{45}$$

$$p_{ij} = \begin{cases} \frac{2}{(tl_j - te_j)^2} \times (t - ta_j + \frac{tl_j - te_j}{2}) \times X_{ij}, & ta_j - \frac{tl_j - te_j}{2} \le t < ta_j \\ -\frac{2}{(tl_j - te_j)^2} \times (t - ta_j - \frac{tl_j - te_j}{2}) \times X_{ij}, & ta_j \le t < t + \frac{tl_j - te_j}{2} \\ 0, & elsewise \end{cases} \tag{46}$$

After deriving the arrival density function, integrating over the range of te_j and tl_j, as in Eq (48) , the probability of the vehicle arriving on time can be computed, denoted as $\Delta\omega_j$. Inspired by the pheromone mechanism in the ACO algorithm, an iterative process also exists for the time pheromone:

$$\phi_{ij}^{(gobal)} = \omega_{ij}^{new} = \Delta\omega_{ij} + (1-\rho)\omega_{ij}^{old} \tag{47}$$

In this way, a memorizing and forgetting process for global time information is created.

$$\Delta\omega_{ij} = \int_{tl_j}^{te_j} p_{ij}\, dt \tag{48}$$

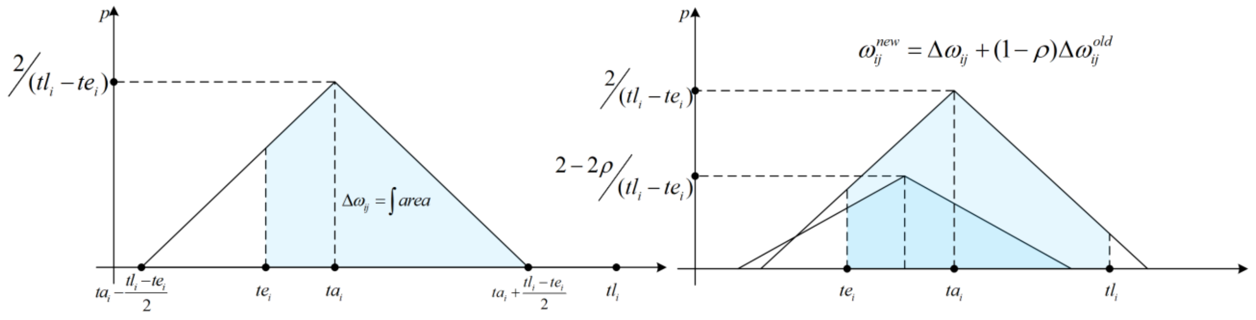

p
$2/(tl_i - te_i)$
$\Delta\omega_{ij} = \int area$
$ta_i - \frac{tl_i - te_i}{2}$
te_i
ta_i
$ta_i + \frac{tl_i - te_i}{2}$
tl_i
p
$\omega_{ij}^{new} = \Delta\omega_{ij} + (1-\rho)\Delta\omega_{ij}^{old}$
$2/(tl_i - te_i)$
$2-2\rho/(tl_i - te_i)$
te_i
ta_i
tl_i

Figure 6. Computation and iteration of global time pheromone.

5. Experiment study

5.1. *Experiment setting*

To demonstrate the proposed models and algorithms, the express order information and map of

Southeast University, Nanjing, China is used. The selected building information includes four buildings, A, B, C, and D, each with six floors and ten rooms per floor, with a total of 240 rooms. The specific map information of the park can be seen in Figure 7.

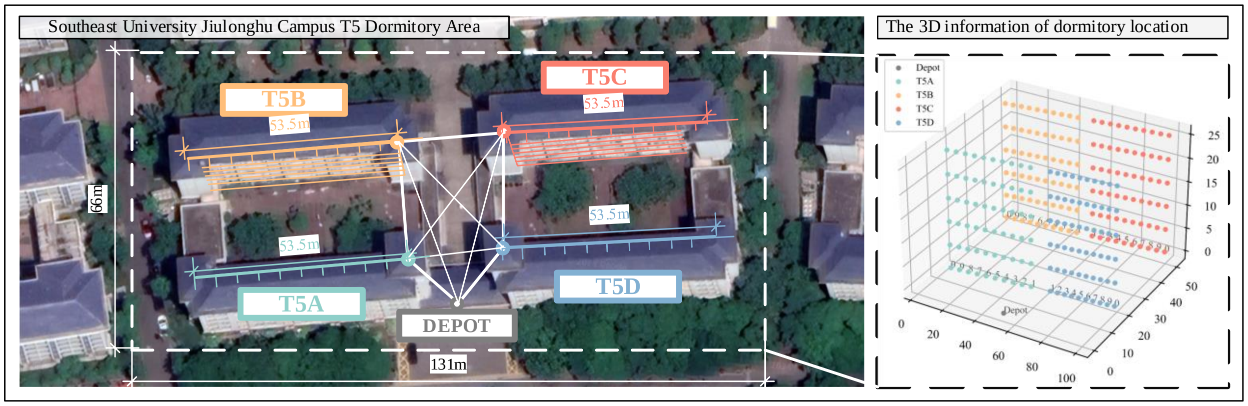

Southeast University Jiulonghu Campus T5 Dormitory Area
T5B
T5C
T5A
T5D
DEPOT
53.5m
66m
131m
The 3D information of dormitory location
Depot
T5A
T5B
T5C
T5D

Figure 7. 3D image of the data set.

The generated order information and time distribution are shown in Table 3 and Figure 8.

Table 3. Order information.

Order ID	House number	Number of orders	Number of large pieces	Number of small pieces	Earliest delivery time	Latest delivery time
2024012801	T5D101	2	1	1	11:45:00	12:30:00
2024012802	T5C101	3	2	1	11:30:00	12:30:00
2024012803	T5A203	4	0	4	11:00:00	12:00:00
……	……	……	……	……	……	……
2024012829	T5A105	6	5	1	13:10:00	15:45:00
2024012830	T5A108	4	2	2	11:10:00	13:15:00

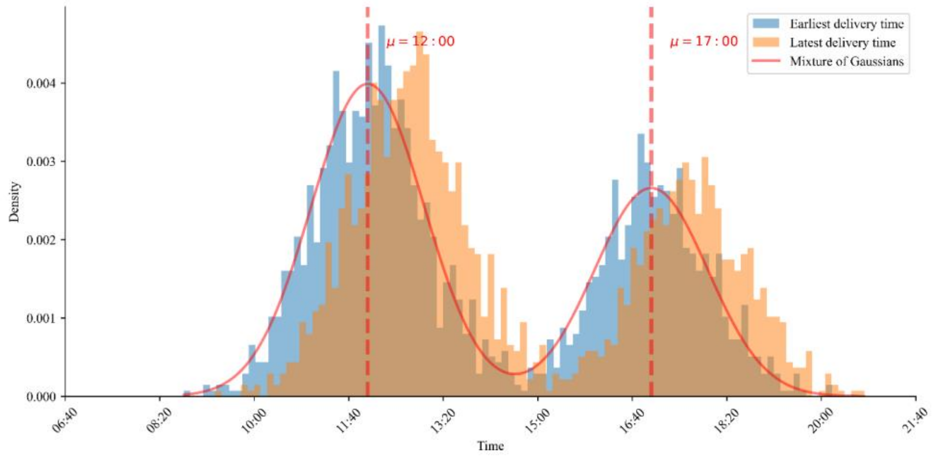


Figure 8. The distribution of user demand time windows.

5.2. *Data preprocessing*

5.2.1. Creating distance graph

As shown as Figure 9, the Euclidean distance and Manhattan distance are not applicable in a 3D environment, because ADRs can move between floors including climbing stairs or using an elevator. In this paper, the following algorithm is proposed, which assumes the complete graph of distances is $G=(V,E)$, where $V=\{v_1,v_2,...,v_n\}$ corresponds to customer nodes $I=\{i_1,i_2,...,i_n\}$, with n being the number of customers. Each node records its corresponding 3D spatial information (e_i, f_i, d_i), where e_i is the coordinate, f_i indicates the floor number, and d_i represents the corresponding door number. The purpose of the algorithm proposed in this paper is to calculate the weight matrix W.

Algorithm 4: The method to calculate the weight matrix

Input: $G=(V,E)$

Output: $G'=(V,E,W)$

```
1: for $i$ in $V$:
2:     for $j$ in $V$ V:
3:         if $e_i = e_j$ ://Determine if they are in the same building
4:             $w_{ij} = \tau \times | f_i - f_j | + D(d_i) + D(d_j)$
5:         else:
6:             $w_{ij} = \tau \times ( f_i + f_j ) + D(d_i) + D(d_j) + diatance\ between\ e_i\ and\ e_j$
7:     End for
8: End for
```

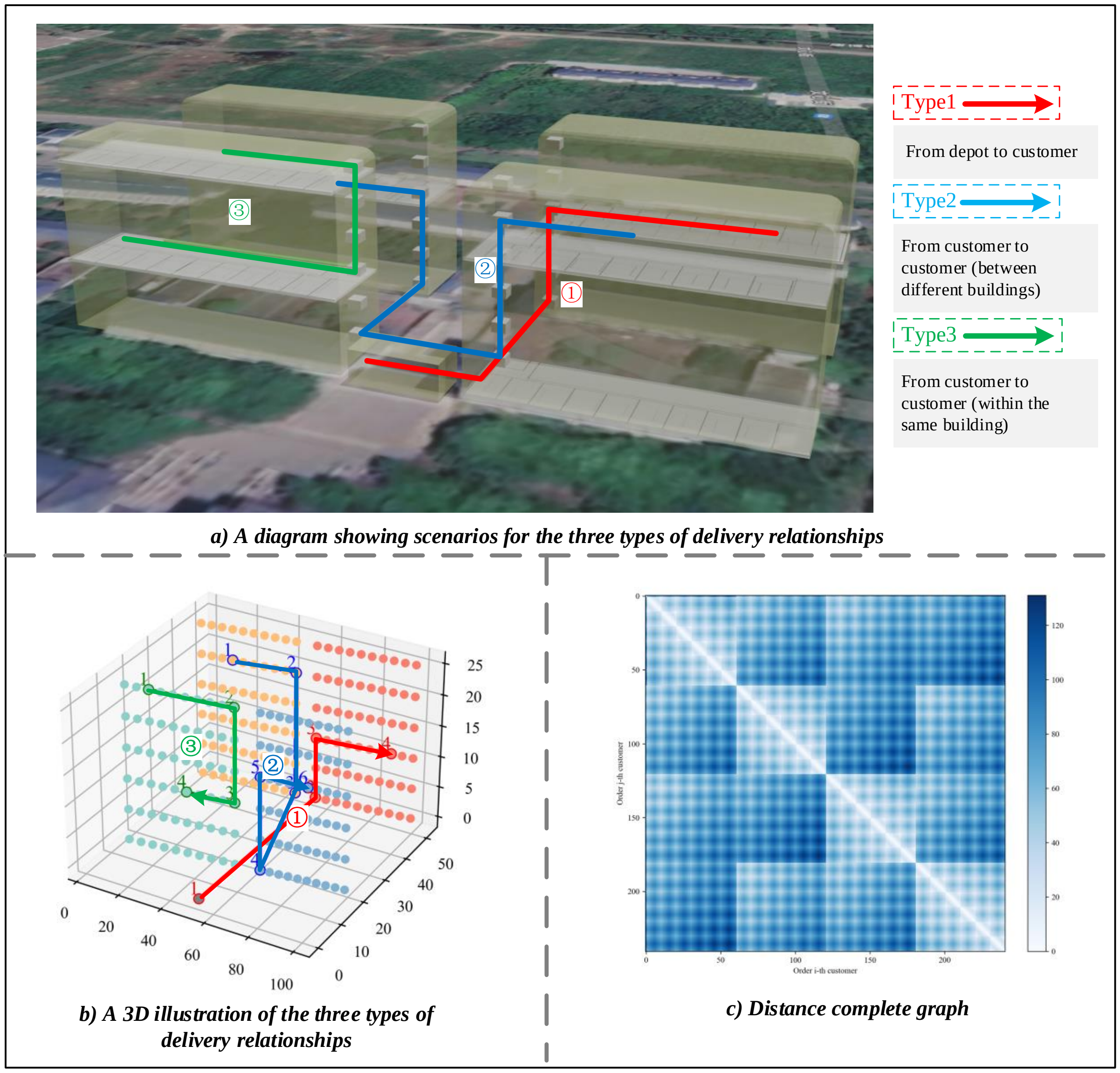
Return $G'=(V,E,W)$

Explanation:

τ is the distance coefficient up (down) to the first floor:

$$\tau = \frac{Time\ per\ unit\ distance\ travelled\ by\ the\ vehicle}{Time\ per\ unit\ distance\ travelled\ by\ the\ vehicle\ on\ level\ ground} * h$$

h is the height of a single-story building. $D()$ function indicates the distance from different room types to the passageway exit, a constant value function.



Type1
From depot to customer
Type2
From customer to customer (between different buildings)
Type3
From customer to customer (within the same building)
①
②
③
Order i-th customer
Order j-th customer

a) A diagram showing scenarios for the three types of delivery relationships

b) A 3D illustration of the three types of delivery relationships

c) Distance complete graph

Figure 9. The distance graph in a 3D environment.

It can be observed that in distance calculations in a 3D environment, three primary distinct cases exist: ① From depot to customer, ② From customer to customer between different buildings, and ③ From customer to customer within the same building. Figure 9a) illustrates the movement of the delivery vehicle between buildings for these three cases. Figure 9b) provides a more intuitive representation by placing this scenario within a 3D environment. By applying Algorithm 4, the corresponding complete distance graph can be obtained in Figure 9c).

5.2.2. Adding dummy node

Due to the discrete capacity window feature, dummy nodes are introduced. This paper generates a distance complete graph in 3D environment $G = (V, E, W)$, where vertex set V is a combination of the customer set V = $\{v_1, v_2, ..., v_n\}$ and depot v_0. For each customer node v_i, if v_i owns m_i demand orders, the customer node v_i is then augmented to $\{v_i^1, v_i^2, ..., v_i^{m_i}\}$. $\{v_i^1, v_i^2, ..., v_i^{m_i}\}$ corresponding to a 3D location information remaining consistent with the 3D spatial information of the node v_i in G. After that, the corresponding generated graph $G' = (V', E')$ is input into the weight calculation algorithm, resulting in a complete graph $G' = (V', E', W')$, where the vertex set V' = $\{v_1^1, v_1^2, ..., v_n^{m_n}\}$. The number of vertexes V' is $\sum_{i=1}^{n} m_i$. Thus, after generating the augmented complete graph of distances, the distance between any two nodes can be directly indexed during the algorithm implementation.

5.3. *Optimization results*

In this paper, we set $|S| = 2$, as the discrete containers are divided into two types: large and small, corresponding to large goods and small goods; the large container can accommodate $n = 4$ small containers. We generate customer demands for different models with different customer numbers under ratios of large to small goods (Ratio = sum of large goods number/ sum of small goods number). For example, if the total number of customer demands is set to 100 with a large-to-small ratio of 1:1, we randomly assign 50 large goods and 50 small goods across the customers. The corresponding augmented complete distance graph's cumulative order numbers range from 200 to 1500. We also give the baseline. The characteristic of baseline route planning is that vehicles always first serve customers near the depot such as in the order of T5A101, T5A102, T5A103, etc., resulting in a lower distance cost. In conducting the experiments, this paper uses the following parameter settings as Table 4.

Table 4. Parameter settings.

Notation	Explanation	Value
Q	The total amount of pheromone released by an ant	5000
α	Distance factor in Eq (43)	2
β	Pheromone factor in Eq (43)	1
γ	Time factor in Eq (43)	1
ρ	Pheromone volatilization rate	0.8
$epochs$	Number of iterations of the algorithm	20
f_1	The max cell number of large containers	20
f_2	The max cell number of small containers	10
p	The number of population size of ants	50
c_s / c_t	Ratio of distance penalty factor to time penalty factor	36
τ	Distance coefficient in Algorithm 2	1

The complete 45 optimization results can be viewed in Table 6 in the appendix. Results show that the algorithm achieves large reductions in total costs across all scenarios compared to the baseline, with improvement ratios ranging from 0.23 to 0.94. The algorithm adeptly balances distance and time costs to optimize the delivery process. And the number of activated vehicles can often reach the minimum. Regarding P3, it adjusts constraints to integrate delivery and pick-up operations, resulting in a significant reduction in activated vehicles. It indicates the effectiveness of the search mechanism.

To highlight the heterogeneity among models, Figure 10 compares their performance under a scenario with 200 customers and a goods size ratio of 1:2. P1 and P2 show similar results across all metrics, while P3 stands out with significantly higher total cost and computation time, but fewer required vehicles—reflecting its use of a lookahead search mechanism. Cost breakdown shows that distance costs are relatively consistent across models, whereas time costs dominate, especially in large-scale instances. Notably, P3 achieves the lowest distance cost due to more efficient container utilization—each container handles both delivery and pickup, reducing routing demands. The higher computation time for P3 further confirms its greater algorithmic complexity.

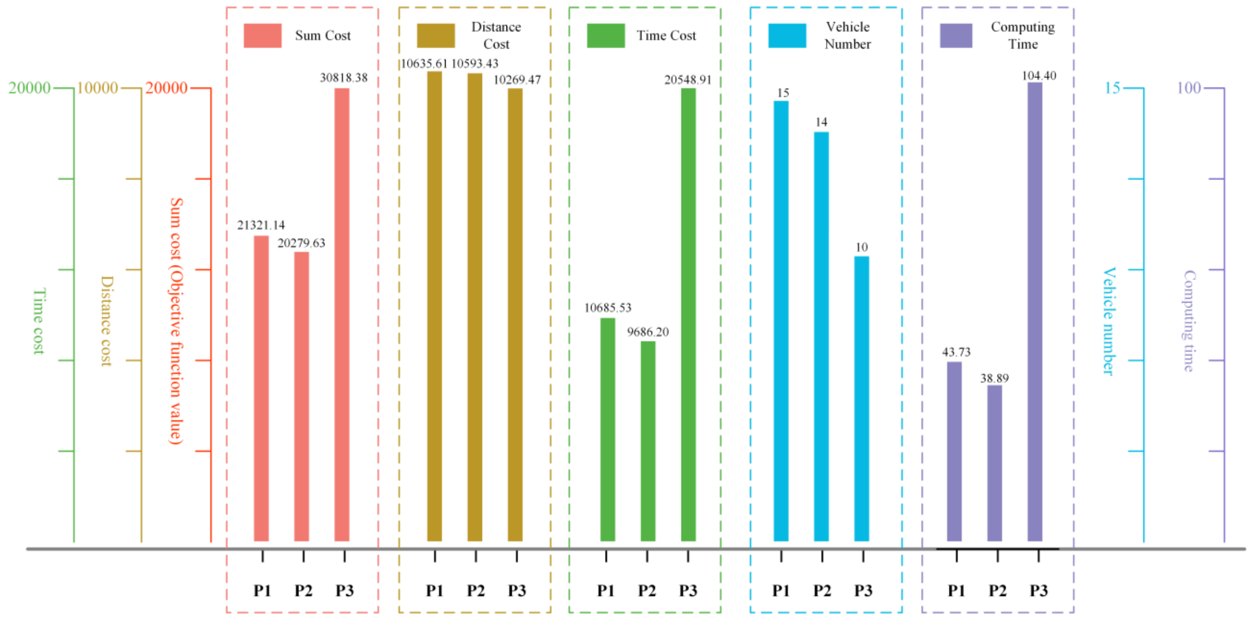


Figure 10. Comparative analysis of optimization results for three models.

5.4. *Ablation experiment*

We conduct the comprehensive analysis of several solution algorithms for the problem P3. The baseline is an unoptimized algorithm. The algorithms analyzed include standard ACO and several optimized variants, which are parts of the SMC-TACSATP ablation studies. All algorithms, except the baseline, use the SMC method to enhance computational speed, hence they are not specifically labeled.

- TACSATP: Fully optimized variant that uses all methods proposed in this paper
- ACSATP: A variant that only removes the tabu table
- ACSA: A variant that removes the temporal pheromones and the tabu list

- TACOTP: A variant that removes the searching mechanism proposed in this paper
- ACOTP: A variant that further removes both the searching mechanism and the tabu list
- ACO: Standard implementation of the ACO algorithm
- Baseline: Unoptimized routing generated based on the proximity principle

Table 5 shows the advantages of SMC-TACSAP by varying the size-to-goods ratio for a fixed number of 300 customers. The metrics compared include the objective function value (OV), the number of used vehicles (VN), the ratio of large to small goods (Ratio)—which represents the proportion of large goods relative to small goods—and the overall capacity utilization (AC). Its integration of time pheromones and a custom search mechanism significantly improves performance, outperforming all other variants in 11 out of 16 cases. Compared to the baseline and standard ACO, TACSATP reduces costs by over 80% in some settings, proving its strong efficiency and optimization capability in complex delivery environments.

Table 5. Ablation experiment.

Ratio	1:1	1:2	2:1	1:3	3:1	1:1	2:3	3:2
TACSATP (OV)	36,737	27,482	51,004	44,611	69,801	20,648	39,838	43,593
TACSATP (VN)	34	14	51	17	55	4	23	47
ACSATP (OV)	43,112	39,825	55,379	37,732	77,613	26,424	41,527	46,482
ACSATP (VN)	34	14	54	17	55	4	22	43
ACSA (OV)	62,738	46,129	86,670	64,599	98,507	27,763	56,050	73,552
ACSA (VN)	41	16	67	20	70	5	28	55
TACOTP (OV)	36,865	30,420	43,668	42,991	53,286	27,960	33,963	37,003
TACOTP (VN)	35	14	53	17	55	4	24	46
ACOTP (OV)	41,002	31,073	44,416	41,240	50,193	20,821	33,442	37,814
ACOTP (VN)	34	13	51	17	54	4	23	48
ACO (OV)	64,963	34,149	69,536	55,769	70,142	30,779	54,832	56,605
ACO (VN)	34	24	54	17	55	4	23	45
Baseline (OV)	184,328	91,632	189,969	142,013	191,875	49,771	136,796	186,124
Baseline (VN)	47	21	72	24	75	6	34	63
AC	2310	1016	3169	1193	3286	299	1680	2911

In Figure 11, we fix the large-to-small goods ratio at 1:2. P3 experiments range from 50 to 500 customers, showcasing SMC-TACSATP's advantages. For comparative experiments, Baseline, ACSA, TACOTP, and TACSATP are selected. It can be observed that the indicators increase with the number of customers. To eliminate this increasing trend, Figure 12 calculates the mean values for each metric, including the average cost and the average number of activated vehicles per customer. By analyzing these two figures, it can be observed that TACSATP consistently achieves a lower value.

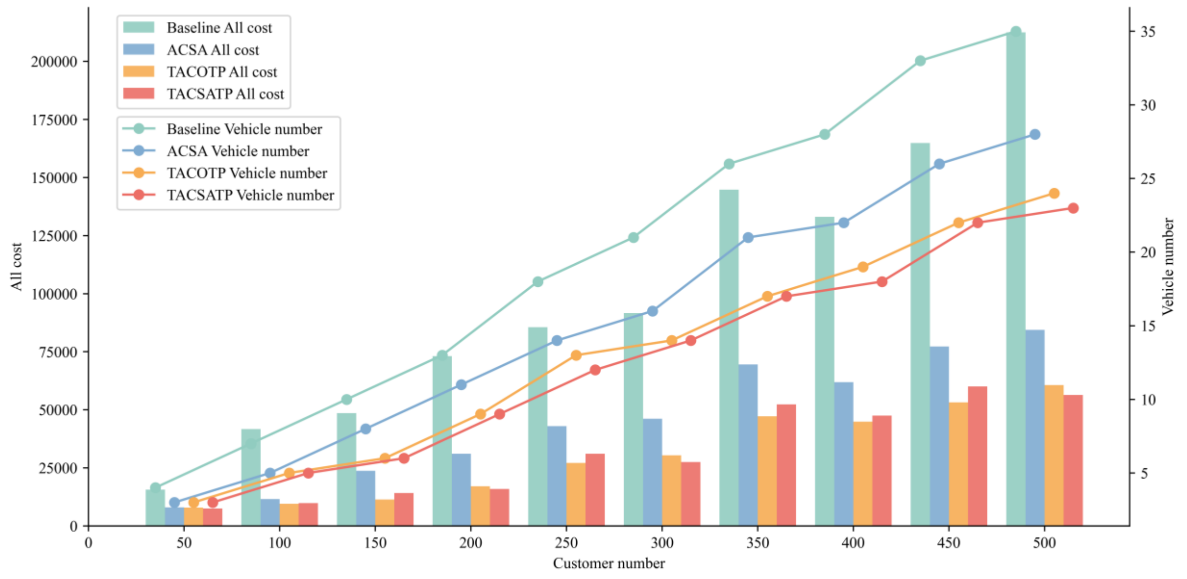


Figure 11. Illustration of the comparative results 1.

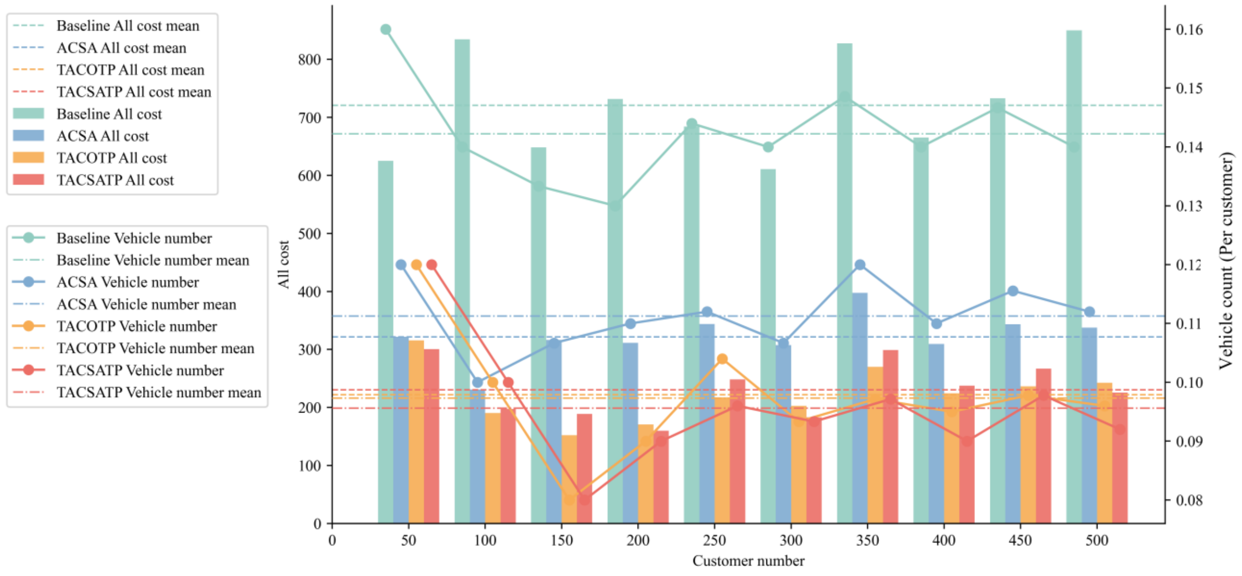


Figure 12. Illustration of the comparative results 2.

The proposed model and algorithms demonstrate substantial advantages in complex 3D delivery environments by accurately modeling multi-floor, multi-order scenarios through the use of virtual nodes and a customized distance calculation method. Specifically, our model effectively addresses the simultaneous pickup and delivery problem within a 3D environment, reducing overall operational costs while reliably meeting customer demands.

Key mechanisms such as the SMC and spatiotemporal pheromones informed by prior knowledge contribute directly to performance improvements. By effectively decomposing complex routing problems into manageable temporal and spatial sub-problems, the SMC approach optimizes computational efficiency and route effectiveness. Additionally, the proposed TACSATP algorithm leverages ACO integrated with a tabu table and improved search mechanisms, significantly enhancing the exploration of feasible solutions and preventing premature convergence.

Extensive experimentation demonstrates notable cost reductions across various customer scales and ratios of goods, achieving savings ranging from 23% to 94%. Particularly, the integrated P3 model utilizes a forward-looking search mechanism that effectively coordinates delivery and pickup tasks, thereby optimizing container utilization, minimizing vehicle deployment, and reducing travel distances.

6. Conclusions

This paper presents a novel approach to solve the last-mile delivery problem in 3D environment by addressing the DCVRP-SDP-STW for electric ADRs. The improved ant colony search algorithm significantly enhanced solution quality. The case study indicates that the algorithm achieved optimization improvements ranging from 23% to 94% compared to the pre-optimization state. These results highlight the effectiveness of integrating advanced algorithms with 3D modeling in complex delivery systems.

This paper explores the large-scale route optimization within 3D environment. Future research might focus on refining operational models, particularly by integrating dynamic consumer behavior data and real-time adjustments in delivery strategies. Additionally, investigating the interaction between smart delivery robots and consumer preferences, as well as the potential for differentiated pricing mechanisms, could lead to more efficient and cost-effective logistics operations. By incorporating machine learning and advanced heuristic methods[38,39], future studies can further enhance the quality of solutions in complex delivery systems, contributing to the development of more intelligent and autonomous logistics networks.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this paper.

Acknowledgments

This paper is supported by the Youth Program of National Natural Science Foundation of China (No. 72301066), Fundamental Research Funds for the Central Universities, China (No. 2242024RCB0048), Natural Science Foundation of Jiangsu Province (No. SBK2024021394), Wuxi Science and Technology Development Fund Project (No. K20231014), National Undergraduate Training Programs for Innovation and Entrepreneurship (NO. 202410286135Z).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. J. Juhász, T. Bányai, Last mile logistics: an integrated view, *IOP Conf. Ser.: Mater. Sci. Eng.*, **448** (2018), 012026. https://doi.org/10.1088/1757-899X/448/1/012026
2. C. Liu, Z. Wang, Z. Liu, K. Huang, Multi-Agent reinforcement learning framework for addressing Demand-Supply imbalance of Shared Autonomous Electric Vehicle, *Transp. Res. Part E Logist. Transp. Rev.*, **197** (2025), 104062. https://doi.org/10.1016/j.tre.2025.104062
3. M. R. Ibarra, J. D. M. Saphores, 1,000 HP electric drayage trucks as a substitute for new freeway lanes construction, *Transp. Res. Part Policy Pract.*, **171** (2023), 103646. https://doi.org/10.1016/j.tra.2023.103646
4. D. Huang, J. Zhang, Z. Liu, A robust coordinated charging scheduling approach for hybrid electric bus charging systems, *Transp. Res. Part Transp. Environ.*, **125** (2023), 103955. https://doi.org/10.1016/j.trd.2023.103955
5. Z. Liu, X. Chen, Q. Meng, I. Kim, Remote park-and-ride network equilibrium model and its applications, *Transp. Res. Part B Methodol.*, **117** (2018), 37–62. https://doi.org/10.1016/j.trb.2018.08.004
6. J. R. Montoya-Torres, J. L. Franco, S. N. Isaza, H. F. Jiménez, N. Herazo-Padilla, A literature review on the vehicle routing problem with multiple depots, *Comput. Ind. Eng.*, **79** (2015), 115–129.
7. C. C. Murray, A. G. Chu, The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery, *Transp. Res. Part C Emerg. Technol.*, **54** (2015), 86–109. https://doi.org/10.1016/j.trc.2015.03.005
8. D. Reyes, M. Savelsbergh, A. Toriello, Vehicle routing with roaming delivery locations, *Transp. Res. Part C Emerg. Technol.*, **80** (2017), 71–91. https://doi.org/10.1016/j.trc.2017.04.003
9. Y. Wang, Y. Wei, X. Wang, Z. Wang, H. Wang, A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints, *Appl. Soft Comput.*, **133** (2023), 109922. https://doi.org/10.1016/j.asoc.2022.109922
10. D. Huang, Z. Liu, P. Liu, J. Chen, Optimal transit fare and service frequency of a nonlinear origin-destination based fare structure, *Transp. Res. Part E Logist. Transp. Rev.*, **96** (2016), 1–19. https://doi.org/10.1016/j.tre.2016.10.004
11. Z. Jia, D. Huang, Z. Liu, Z. Hu, R. Liu, W. Yu, Multi-objective optimization for the sightseeing bus problem: Trade-off between tourists and operator, *Expert Syst. Appl.*, **269** (2025), 126341. https://doi.org/10.1016/j.eswa.2024.126341
12. Y. Liu, Z. Liu, R. Jia, DeepPF: A deep learning based architecture for metro passenger flow prediction, *Transp. Res. Part C Emerg. Technol.*, **101** (2019), 18–34. https://doi.org/10.1016/j.trc.2019.01.027
13. X. Chen, Z. Liu, K. Zhang, Z. Wang, A parallel computing approach to solve traffic assignment using path-based gradient projection algorithm, *Transp. Res. Part C Emerg. Technol.*, **120** (2020), 102809. https://doi.org/10.1016/j.trc.2020.102809
14. Q. Cheng, Z. Wang, Y. Lin, Z. Liu, Capturing traffic state variation process: An analytical modeling approach, *Transp. Res. Part E Logist. Transp. Rev.*, **198** (2025), 104119. https://doi.org/10.1016/j.tre.2025.104119

15. J. C. Thill, T. H. D. Dao, Y. Zhou, Traveling in the three-dimensional city: applications in route planning, accessibility assessment, location analysis and beyond, *J. Transp. Geogr.*, **19** (2011), 405–421. https://doi.org/10.1016/j.jtrangeo.2010.11.007
16. T. Bektas, The multiple traveling salesman problem: an overview of formulations and solution procedures, *omega*, **34** (2006), 209–219.
17. B. Eksioglu, A. V. Vural, A. Reisman, The vehicle routing problem: A taxonomic review, *Comput. Ind. Eng.*, **57** (2009), 1472–1483.
18. J. Mańdziuk, C. Nejman, Uct-based approach to capacitated vehicle routing problem, in *Artificial Intelligence and Soft Computing: 14th International Conference*, *Proceedings, Part II 14*. Springer International Publishing, (2015), 679–690. https://doi.org/10.1007/978-3-319-19369-4_60
19. G. Kim, Y. S. Ong, C. K. Heng, P. S. Tan, N. A. Zhang, City Vehicle Routing Problem (City VRP): A review, *IEEE Trans. Intell. Transp. Syst.*, **16** (2015), 1654–1666. https://doi.org/10.1109/TITS.2015.2395536
20. B. L. Golden, *Vehicle Routing Problems: Formulations and Heuristic Solution Techniques*. Massachusetts Institute of Technology, Operations Research Center, 1975.
21. H. Quak, M. B. de Koster, Delivering goods in urban areas: how to deal with urban policy restrictions and the environment, *Transp. Sci.*, **43** (2009), 211–227.
22. S. W. Lin, V. F. Yu, S. Y. Chou, A note on the truck and trailer routing problem, *Expert Syst. Appl.*, **37** (2010), 899–903. https://doi.org/10.1016/j.eswa.2009.06.077
23. M. M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, *Oper. Res.*, **35** (1987), 254–265.
24. N. Balakrishnan, Simple heuristics for the vehicle routeing problem with soft time windows, *J. Oper. Res. Soc.*, **44** (1993), 279–287.
25. M. Gendreau, F. Guertin, J. Y. Potvin, É. Taillard, Parallel tabu search for real-time vehicle routing and dispatching, *Transp. Sci.*, 1999. https://doi.org/10.1287/trsc.33.4.381
26. H. Hashimoto, T. Ibaraki, S. Imahori, M. Yagiura, The vehicle routing problem with flexible time windows and traveling times, *Discrete Appl. Math.*, **154** (2006), 2271–2290. https://doi.org/10.1016/j.dam.2006.04.009
27. E. Angelelli, R. Mansini, The vehicle routing problem with time windows and simultaneous pick-up and delivery, in *Quantitative Approaches to Distribution Logistics and Supply Chain Management*, Springer, (2002), 249–267.
28. M. Goetschalckx, C. Jacobs-Blecha, The vehicle routing problem with backhauls, *Eur. J. Oper. Res.*, **42** (1989), 39–51.
29. A. M. Altabeeb, A. M. Mohsen, A. Ghallab, An improved hybrid firefly algorithm for capacitated vehicle routing problem, *Appl. Soft Comput.*, **84** (2019), 105728.
30. L. M. Gambardella, É. Taillard, G. Agazzi, *MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows*, Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 1999.
31. Y. Wang, L. Wang, Z. Peng, G. Chen, Z. Cai, L. Xing, A multi ant system based hybrid heuristic algorithm for vehicle routing problem with service time customization, *Swarm Evol. Comput.*, **50** (2019), 100563. https://doi.org/10.1016/j.swevo.2019.100563
32. H. Zhang, Q. Zhang, L. Ma, Z. Zhang, Y. Liu, A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows, *Inf. Sci.*, **490** (2019), 166–190. https://doi.org/10.1016/j.ins.2019.03.070

33. X. Zheng, F. Gao, X. Tong, Research on green vehicle path planning of AGVs with simultaneous pickup and delivery in intelligent workshop, *Symmetry*, **15** (2023), https://doi.org/10.3390/sym15081505
34. A. Goel, V. Gruhn, A general vehicle routing problem, *Eur. J. Oper. Res.*, **191** (2008), 650–660.
35. Z. Gu, X. Yang, Q. Zhang, W. Yu, Z. Liu, TERL: Two-stage ensemble reinforcement learning paradigm for large-scale decentralized decision making in transportation simulation, *IEEE Trans. Knowl. Data Eng.*, **35** (2023), 13043–13054. https://doi.org/10.1109/TKDE.2023.3272688
36. A. H. Golsefidi, F. B. Hüttel, I. Peled, S. Samaranayake, F. C. Pereira, A joint machine learning and optimization approach for incremental expansion of electric vehicle charging infrastructure, *Transp. Res. Part Policy Pract.*, **178** (2023), 103863. https://doi.org/10.1016/j.tra.2023.103863
37. Y. Wang, S. Luo, J. Fan, L. Zhen, The multidepot vehicle routing problem with intelligent recycling prices and transportation resource sharing, *Transp. Res. Part E Logist. Transp. Rev.*, **185** (2024), 103503. https://doi.org/10.1016/j.tre.2024.103503
38. S. Erdoğan, E. Miller-Hooks, A green vehicle routing problem, *Transp. Res. Part E Logist. Transp. Rev.*, **48** (2012), 100–114. https://doi.org/10.1016/j.tre.2011.08.001
39. X. Yang, Z. Liu, Q. Cheng, P. Liu, Geometry-aware car-following model construction: Theoretical modeling and empirical analysis on horizontal curves, *Transp. Res. Part C Emerg. Technol.*, **166** (2024), 104772. https://doi.org/10.1016/j.trc.2024.104772
40. Z. Wang, Y. Lin, Z. Liu, Y. Zheng, P. Liu, Q. Cheng, *Traffic Dynamics Modeling with an Extended S3 Car Following Model*, Social Science Research Network, Rochester, https://doi.org/10.2139/ssrn.4882338

Appendix

Table A. The complete optimization results.

NO	All cost	Distance Cost	Time Cost	Vehicle Number	Baseline	Ratio Improved	Customer Number	Ratio	Sum Good	Type
1	6071.4	1940.9	4130.5	3	11,172.8	0.46	30	1： 1	120	1
2	6315.5	1879.5	4436.0	3	13,497.1	0.53	30	1： 2	115	1
3	5181.3	2242.4	2938.9	5	26,101.7	0.80	30	2： 1	227	1
4	4745.3	1994.2	2751.1	4	18,808.8	0.75	30	1： 1	164	2
5	7326.2	1956.2	5369.9	2	9458.8	0.23	30	1： 2	90	2
6	6381.3	1968.9	4412.5	5	18,631.4	0.66	30	2： 1	235	2
7	9918.8	3825.0	6093.8	6	29,382.3	0.66	50	1： 1	257	1
8	7190.0	2982.2	4207.8	3	16,342.6	0.56	50	1： 2	140	1
9	9592.7	4090.2	5502.5	9	45,553.1	0.79	50	2： 1	416	1
10	9737.7	3458.8	6279.0	6	29,513.2	0.67	50	1： 1	294	2
11	6915.7	3089.1	3826.7	4	22,048.9	0.69	50	1： 2	165	2
12	9186.5	4735.9	4450.6	9	34,942.5	0.74	50	2： 1	379	2
13	13,084.4	7110.1	5974.4	10	49,403.5	0.74	100	1： 1	462	1
14	9256.8	5807.7	3449.1	7	47,634.2	0.81	100	1： 2	330	1
15	20,066.7	7673.9	12,392.9	16	68,338.1	0.71	100	2： 1	743	1
16	14,113.7	6627.3	7486.4	11	49,689.2	0.72	100	1： 1	542	2
17	14,131.5	5635.4	8496.1	7	44,126.7	0.68	100	1： 2	340	2
18	13,575.6	7228.8	6346.8	15	59,247.0	0.77	100	2： 1	676	2
19	14,352.5	6864.8	7487.7	10	52,370.0	0.87	100	1： 1	706	3
20	12,101.8	6250.6	5851.2	5	38,641.7	0.84	100	1： 2	330	3
21	20,378.4	9662.6	10,715.8	20	66,210.3	0.85	100	2： 1	1138	3
22	17,601.7	9209.5	8392.2	17	92,819.4	0.81	150	1： 1	812	1
23	20,890.5	8714.0	12,176.5	11	76,726.7	0.73	150	1： 2	510	1
24	21,026.4	11,049.8	9976.6	22	106,124.4	0.80	150	2： 1	1041	1
25	16,134.4	9191.5	6942.9	15	79,299.7	0.80	150	1： 1	717	2
26	19,210.4	8822.2	10,388.1	11	80,862.7	0.76	150	1： 2	525	2
27	20,044.3	11957.9	8086.4	23	109,801.2	0.82	150	2： 1	1071	2
28	22,595.1	10,933.4	11,661.7	21	119,158.0	0.81	200	1： 1	1001	1
29	21,207.2	10,565.1	10,642.1	13	94,226.2	0.78	200	1： 2	635	1
30	26,282.7	13,967.4	12,315.3	30	125,250.8	0.79	200	2： 1	1410	1
31	23,169.0	11,492.5	11,676.5	21	98,015.0	0.76	200	1： 1	1036	2
32	23,106.6	9604.9	13,501.7	14	107,608.4	0.79	200	1： 2	660	2
33	27,223.2	14,759.7	12,463.5	31	138,247.0	0.80	200	2： 1	1435	2
34	36,208.0	11,979.9	24,228.1	21	103,563.1	0.88	200	1： 1	1396	3
35	28,057.0	10,140.3	17,916.7	9	71,281.3	0.86	200	1： 2	635	3
36	33,433.7	15,555.7	17,878.0	39	130,443.0	0.88	200	2： 1	2255	3
37	36,737.0	15,121.1	21,615.9	34	184,327.9	0.92	300	1： 1	2311	3
38	27,482.0	14,209.1	13,272.9	14	91,631.7	0.84	300	1： 2	1020	3
39	5,1003.7	20,951.6	30,052.0	52	189,968.8	0.89	300	2： 1	3170	3

40	60,351.2	19,172.7	41,178.5	44	244,832.1	0.92	400	1：1	3069	3
41	45,441.6	14,571.2	30,870.4	20	172,108.7	0.92	400	1：2	1465	3
42	87,590.1	26,886.2	60,703.9	76	252,777.2	0.89	400	2：1	4502	3
43	63,709.3	21,013.3	42,696.0	56	330,393.7	0.94	500	1：1	3914	3
44	63,582.0	19,625.5	43,956.6	24	200,449.5	0.90	500	1：2	1735	3
45	114,400.5	31,468.5	82,932.0	90	339,705.1	0.91	500	2：1	5524	3