Van-based robot hybrid pickup and delivery routing problem

Shaohua Yu*[a,b], Jakob Puchinger[b,c], Shudong Sun[a,d]

[a] *Department of Industrial Engineering, Northwestern Polytechnical University, Xi'an 710072, China*
[b] *Laboratoire Genie Industriel, CentraleSupelec, Universite Paris-Saclay, Gif-sur-Yvette, France*
[c] *Institut de Recherche Technologique SystemX, Palaiseau, France*
[d] *Key Laboratory of Industrial Engineering and Intelligent Manufacturing, Ministry of Industry and Information Technology, Xi'an 710072, PR China*

Abstract

We present a two-echelon van-based robot last-mile pickup and delivery system in urban settings. Robots can visit areas with van access restrictions, such as pedestrianized areas or university campuses. The van stops at parking nodes to drop off and/or pick up its robot, and to replenish its robot and/or swap its robot's battery if needed. Five van/robot pickup and delivery cases are considered, according to the roles the van/robot plays in the process of pickup/delivery and whether the van transports its robot.

To model the proposed problem, we introduce a mixed-integer program including time, freight, and energy. We further propose an adaptive large neighborhood search algorithm to solve larger instances and a capacity feasibility test approach for a single route. We then assess the influence of parking node density on model output. A case study based on a realistic city scenario is introduced. A sensitivity analysis is performed on the robot's travel cost and maximum travel distances, and the effect of van no-go areas is studied. Two classical models (two-echelon vehicle routing with hybrid pickup and delivery and parallel van and robot scheduling with hybrid pickup and delivery) are compared with ours, and results show our model is competitive in appropriate scenario settings. We therefore advocate using the two-echelon van-based robot last-mile pickup and delivery system in urban areas.

Keywords: Logistics, Innovative last mile distribution, Mothership based robot pickups and deliveries, Adaptive Large Neighbourhood Search

1. Introduction

Urban logistics and transportation companies have to conduct pickup and delivery operations. Pickup-and-delivery problems (PDPs) are an essential family of routing problems in which goods or passengers have to be transported from different origins to different destinations (Battarra et al. 2014).

Supposing an e-commerce company has surplus distribution capacity after completing its e-commerce logistics distribution task. In such a case, it can make gainful use of its surplus logistics capacity to generate additional earnings by helping other individuals deliver freight. For example, JD.com delivers and picks up JD-Mall's own products and helps individuals or individual stores (offline store or local hypermarket) deliver goods in real business operations. For the latter, a request needs to be made to pick up goods from individuals or individual stores and transport them to corresponding customers. A hybrid distribution model requires logistics companies to perform customer-to-customer (one-to-one) and depot-to-customer-to-depot (one-to-many-to-one) hybrid pickup and delivery services simultaneously, raising new challenges for the design and optimization of the distribution system.

Email addresses: shaohua.yu@centralesupelec.fr (Shaohua Yu*), jakob.puchinger@irt-systemx.fr (Jakob Puchinger), sdsun@nwpu.edu.cn (Shudong Sun)

Bergmann et al. (2020) analyzed the route efficiency trade-offs that emerge from combining first-mile pickup and last-mile delivery operations in an urban logistics network. They show that combining deliveries and pickups on a single route can significantly improve the distribution system efficiency. Integrating e-commerce proprietary logistics into one system instead of executing them separately may thus improve logistics distribution efficiency. Perboli and Rosano (2019) studied the mix of traditional and green vehicles in last-mile delivery and found hybrid fleets could reduce CO2 emissions.

As the e-commerce market has grown, researches began to study robot delivery, Boysen et al. (2018), Bakach et al. (2021) studied two models in which a van cooperated with several robots to provide service for customers based on a two-echelon vehicle routing model. Technology companies and traditional logistics providers have been experimenting with robot delivery (Kitjacharoenchai et al. 2020). For example, Starship Technologies (Andrew 2019) and JD.COM (Liping 2018) have implemented autonomous distribution robots on campuses and in some residential areas. Most robots are electrically powered in real-world applications and so will be very limited in range owing to limited battery capacity for the foreseeable future. These robots travel at walking speed for safety reasons, and so their application for long-distance delivery is not efficient (Dongmei 2017). Accordingly, logistic companies are considering van-based robot systems (mothership systems) to overcome the robot's limited ranges by transporting the robot in the van and by extending the number of customers served by the robot by replenishing it in the van.

However, current research on the van-based robot service system mainly focuses on delivery of goods, and very few studies involve goods pickup operations. To our knowledge, only Karak and Abdelghany (2019) have studied a one-to-many-to-one problem with van-based robot pickup and delivery. The mothership leaves a depot to deliver commodities to customers, pick up commodities from the customers, and then transport them back to the depot.

We extend the van-based robot pickup and delivery research by integrating multiple pickup and delivery modes in one trip to increase the efficiency of distribution systems, as envisaged by Bergmann et al. (2020).

This paper presents van-based robot hybrid pickup and deliveries (2E-VRHPD) as a prototypical problem. Larger vans carry small robots along *1st-level routes*. The robots travel along the *2nd-level routes*. Both vans and robots can serve customers directly, but some customers can be visited only by robots. The van stops at parking nodes to drop off and/or pick up its robot, replenish its robot and swap its robot's battery if needed. For hybrid pickup and delivery operations, vans and robots can load goods from a depot and deliver them to a customer, or they can pick up goods from a customer (supplier) and deliver them to another customer or to a depot.

Because of its involvement in pickup operations, the allocation and adjustment of goods during the route becomes a relevant scientific problem. This holds especially for allocating goods to the van and its robot in the parking nodes before they leave to do the distribution separately. The difficulties are twofold: (1) Both robots and vans can reach their maximum capacity when they serve customers independently. However, the combined load of the robot and van cannot exceed the van's maximum capacity when the robot is on board the van. Cargo capacity needs to be coordinated. Given a simple example, the maximum capacity of a van is 200, and the maximum capacity of a robot is 50. If at a parking node, a total load of a van and its robot is 130, and in the following parallel routes where the van and the robot travel independently starting from this parking node, the van should pick up 50 goods, and the robot should pick up 20 goods, we should let van load be less than or equal to 150 and robot load be less than or equal to 30 when starting their trip. Since the van load should be less than or equal to 200 and the robot load should be less than or equal to 50 during parallel routes. (2) Whether a couple of pickup-delivery-pair customers are in a van/robot route or not will influence the freight flow of the van/robot. The details of the differences are described in Section 3.3.4.

Our 2E-VRHPD model presents the following differences to the vehicle routing problem with drones (VRPD) (model of Agatz et al. (2018)) and the truck-and-trailer routing problem with pickup and delivery (TTRPPD) (model of Drexl

(2020)): (i) In our 2E-VRHPD model, we extend the VRPD model to cover hybrid pickup and delivery aspects, (ii) the 2E-VRHPD model extends TTRPPD with second-level open routes, and also introduces a new van and robot capacity trade-off problem, and (iii) small robots are generally safer than other transport modes in cities. For example, when a fast-flying drone stalls, it can cause serious accidents in a city, whereas for a slowly moving robot, accident risks from stalled vehicles are controllable (Yu et al. 2020).

This paper makes the following contributions: (1) We propose a new two-echelon van-based robot routing problem with hybrid pickup and delivery, including five new one-to-one pickup and delivery modes, and a new van-robot capacity trade-off problem. (2) We first present a mathematical programming model to address two special cases: Interdiction to serve a couple of pair-customers (a pickup-pair customer and its corresponding delivery-pair-customer) in parallel routes; Differential calculation of the freight flow when there is a couple of pair-customers and no couple of pair-customers in independent-van/robot route. (3) We propose an ALNS algorithm for the newly introduced problem and a capacity feasibility test approach. (4) We introduce a case study and perform a sensitivity analysis on travel cost, maximum travel distances of a robot, and the impact of van no-go zones on the distribution system. We then compare the 2E-VRHPD model with two classical models and give suggestions for using the new model.

In the following, Section 2 reviews related literature, Section 3 describes the problem and formulation, Sections 4 deals with the adaptive large neighborhood search approaches, a computational study and a case study are presented in Section 5 and Section 6, respectively, and Section 7 concludes.

2. Related literature

This section presents a literature review and a comparative analysis of problems related to the proposed 2E-VRHPD model. We first give the latest research status for van-based robot delivery problems. We then give an overview of recent work on pickup and delivery problems as well as vehicle routing problems with synchronization constraints.

2.1. Van-based robot delivery

Van-based robot (drone or autonomous robot) delivery problems have drawn much attention (Otto et al. 2018, Li et al. 2021). In van-based robot deliveries, a van can transport robots, and along the route it can drop off and pick up a robot at different locations. The advantage of van-based robot deliveries is that goods can be delivered in parallel by the van and its robot, thereby increasing the efficiency of distribution systems compared to classical vehicle routing problems. Murray and Chu (2015) first presented and studied the van-based robot delivery concept, and farther-reaching problems have since been studied.

Table 1 gives similarities and differences between van-based robot routing problems.

The comparison focuses on how many vans and robots are considered and the objective of the problem, the number of customers a drone can visit in the course of one trip (VMC), whether there are time-window constraints for customers (CTW), whether pickup and delivery operations are allowed in the model (PD), and the contribution of the work (Contribution).

Table 1 shows that our paper studies the multi-mothership distribution system and first simultaneously considers pickup and delivery operations and customer time window, compared with the previous van-based robot distribution system. Besides, we also allow a robot to visit multiple customers in one trip in the model. Note that Karak and Abdelghany (2019) studied a one-to-many-to-one pickup and delivery problem in a van-based robot model. However, we integrated the one-to-one and one-to-many-to-one pickup and/or delivery services in the van-based robot model. Hence our model is innovative.

Table 1: Similarities and differences between van-based robot routing problems

Reference	Vans	Robots	Objective	VMC	CTW	PD	Contribution
Murray and Chu (2015)	1	1	time	1	no	no	MIP, heuristic
Poikonen et al. (2017)	n	m	time	1	no	no	Theoretical insights
Wang et al. (2017)	n	m	time	1	no	no	Theoretical insights
Pugliese and Guerriero (2017)	n	m	cost	1	yes	no	MIP
Luo et al. (2017)	n	n	time	m	no	no	MIP, heuristic
Carlsson and Song (2018)	1	1	time	1	no	no	Heuristic
Agatz et al. (2018)	1	1	time	1	no	no	IP, heuristic
Bouman et al. (2018)	1	1	cost	1	no	no	DP
Schermer et al. (2019a)	n	m	time	1	no	no	MIP, VNS, TS
Schermer et al. (2019b)	n	m	time	1	no	no	MIP, matheuristic
Karak and Abdelghany (2019)	1	m	cost	m	no	yes	MIP, heuristic
Sacramento et al. (2019)	n	1	cost	1	no	no	MIP, ALNS
Wang and Sheu (2019)	n	m	cost	m	no	no	MIP, branch-and-price
Poikonen et al. (2019)	1	1	time	1	no	no	Branch-and-bound, heuristic
Murray and Raj (2020)	1	m	time	1	no	no	MIP, heuristic
Poikonen and Golden (2020a)	1	1	time	1	no	no	Branch-and-bound, heuristic
Poikonen and Golden (2020b)	1	m	time	m	no	no	ILP, heuristic
Kitjacharoenchai et al. (2020)	n	m	time	m	yes	no	MIP, heuristic, LNS
Moshref-Javadi et al. (2020a)	1	m	waiting time	1	no	no	MIP, heuristic
Moshref-Javadi et al. (2020b)	1	m	waiting time	1	no	no	MIP, hybrid TS-SA
Dayarian et al. (2020)	1	1	maximize orders	m	no	no	Heuristics
Gonzalez-R et al. (2020)	1	1	time	m	no	no	MIP, Iterated greedy heuristic
Salama and Srinivas (2020)	1	m	time,cost	1	no	no	MIP, Heuristic
Yu et al. (2020)	n	m	cost	m	yes	no	MIP, hybrid metaheuristic
Roberti and Ruthmair (2021)	1	1	time	1	no	no	MIP, DP, branch-and-price
This work	n	n	cost	m	yes	yes	MIP, ALNS

2.2. Pickup and delivery problem

The general pickup and delivery problem (GPDP) refers to when several transportation demands have to be satisfied by a fleet of vehicles executing a set of routes. Each vehicle route has a distinct start and end location and finite load capacity. Each transportation demand is characterized by a group of pickup locations (origins), delivery locations (destinations), and load size. Each order has to be fulfilled as a whole (Savelsbergh and Sol 1995). A classification of pickup and delivery problems can be found in Battarra et al. (2014), and Koç et al. (2020).

Time-window constraints are usually important and should be considered in urban pickup and delivery problems. Dumas et al. (1991) were the first to adopt column generation to solve the PDPTW problem. They proposed a branch-and-bound method that can handle up to 55 requests. Ropke and Pisinger (2006) presented an adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows.

In our van-based robot pickup and delivery model, van, and robot are seen as heterogeneous vehicles. Qu and Bard (2013) studied a heterogeneous pickup and delivery problem in which each vehicle's capacity could be modified by reconfiguring its interior to satisfy different types of customer demands. The number of participants and support equipment items that a van can accommodate depends on how it is configured. They developed a two-phase heuristic that uses ideas from greedy randomized adaptive search procedures with multiple starts to solve this problem. Avci and Topaloglu (2016) studied a heterogeneous vehicle routing problem with simultaneous pickup and delivery problems. They developed a hybrid local search algorithm in which a non-monotone threshold adjustment strategy was integrated with tabu search. They showed that the developed approach could produce efficient and effective solutions. Sun et al. (2019) presented the formulation and exact solution for heterogeneous vehicle pickup and delivery problems, intending to minimize carbon emissions of pickups and deliveries by a fleet of heterogeneous vehicles. Their exact algorithm is based on a set partitioning model and the key characteristics of its optimal solution, which can rapidly find the largest-scale instance's optimal solution. Drexl (2020) studied a one-to-one pickup-and-delivery problem with time windows and trailers. The discussed extension consists in considering a heterogeneous vehicle fleet comprising lorries with detachable trailers. Trailers are advantageous as they increase overall vehicle capacity. However, some locations may be accessible only to lorries.

To our knowledge, the work of Drexl (2020) is the one most similar to ours. But in our model, we integrate one-to-one

and one-to-many-to-one pickup and delivery operations, and the vans and robots can drive independently.

2.3. Vehicle routing problem with multiple synchronization

Vehicle routing problems with multiple synchronizations are vehicle routing problems that exhibit additional synchronization requirements regarding spatial, temporal, and load aspects (Drexl 2012). In a sense, the cooperative delivery of vans and robots can also be regarded as a kind of vehicle synchronization. Here we introduce some vehicle synchronization problems which are closely related to our 2E-VRHPD problem. They are truck and trailer routing problem (TTRP), two-echelon vehicle routing problem (2E-VRP), and two-echelon location routing problem (2E-LRP).

TTRP describes a fleet of truck and trailer combinations with known capacity serving a set of customers with predetermined demands and locations. In TTRP, a vehicle may be a truck pulling a trailer, or a single truck. Some customers must be served by a truck while others can be served either by a truck or a truck pulling a trailer (Parragh and Cordeau 2017, Boysen et al. 2018, Rothenbächer et al. 2018, Drexl 2020, Accorsi and Vigo 2020). In the 2E-VRHPD model, both the van and the robot can move and serve customers by themselves. Whereas the trailer in TTPR model stays stationary while the truck is not pulling, also, the trailer can not serve customers.

2E-VRP is a well-known variant of the classic VRP. The 2E-VRP model plays a very important role in managing urban freight activities in last-mile delivery (Perboli et al. 2021). The 2E-VRP involves a two-layered distribution network: In the first level, vans perform deliveries from distribution centers to urban consolidation centers, also called satellites. At the second level, customers' orders are consolidated into small vehicles that can travel along any street in the city center area, such as minivans, electric vans, and cargo bikes, and delivered to the final customers (Liu et al. 2018, Dellaert et al. 2019, Li et al. 2020, Mühlbauer and Fontaine 2021, Perboli et al. 2021).

2E-LRP is similar to the 2E-VRP. The main difference between the two problems is that the 2E-LRP incorporates the location decision problem into the VRP while the 2E-VRP does not (Wang et al. 2018, Darvish et al. 2019, Mirhedayatian et al. 2021). In 2E-VRP and 2E-LRP, the vehicle synchronization position is satellite, used as a transfer station where vehicle flows between vehicles are synchronized. In the 2E-VRHPD model, the van/robot synchronization position is a parking node, without any storage capacity. Besides, the van serves customers in 2E-VRHPD, while in 2E-VRP and 2E-LRP, the van does not directly serve customers. Furthermore, the connection between the van and the robot is closer than that of the first level vehicle and the second level vehicle in the 2E-VRP and 2E-LRP problem: The van carries the robots and sends them out and picks them up at the same/different rendezvous nodes. Note that the 2E-VRHPD model also incorporates a location decision as the 2E-LRP does: it has to be decided which parking nodes are used by which van and which robot. However, the van is allowed to leave the pickup-parking node until its robot arrives at there. While in the 2E-LRP model, the vehicle can leave the satellites at any time.

3. Problem description and formulation

3.1. Problem statement

There are customers located in narrow streets, on campuses, or in other communities where the entry of vans is restricted or physically impossible. We therefore distinguish two kinds of customers. One kind can be visited by either the van or the robot: we call these customers van customers. The other kind can only be visited by the robot: we call these robot customers. This implies that the robot can serve all the customers, but the van can only serve a subset pf customers.

We consider a two-echelon routing problem with van/robot pickup and delivery. Vans or vans carrying robots move along a *1st-level route*, serve van customers, or drop off/pick up, and replenish or swap their robots' batteries at parking

nodes. Robots handle customer services along *2nd-level open routes*: in other words, the robots need not return to the parking node from where they set out.

In our hybrid pickup and delivery system, we define three kinds of pickup and delivery modes: (i) picking up goods from customers and delivering them to a depot, (ii) delivering goods from a depot to customers, and (iii) picking up goods from suppliers (or customers) and delivering them to other customers (or suppliers).

For the pickup-delivery-pair customers, we assume they have pairing (coupling) and precedence constraints. These constraints mean that the pick-up of goods from a customer and then delivery to other customers is one-to-one, and each customer's goods must be picked up before being delivered. We also let the pickup-delivery-pair customers be served by the same van-based robot mothership system.

In our 2E-VRHPD model, there are five pickup and delivery cases for the vehicles to serve the pickup-delivery-pair-customers. Figure 1 shows the five cases.

- Case 1: A robot picks up goods from a customer and delivers them to another customer.
- Case 2: A van picks up the goods from a customer and delivers them to another customer.
- Case 3: A van picks up goods from a customer and its robot then delivers them to another customer.
- Case 4: A robot picks up goods from a customer and returns to the van; the van then delivers the goods to another customer.
- Case 5: A robot picks up goods from a customer and returns to the van. The van carries the robot and then drops off the robot letting it deliver the goods to another customer.

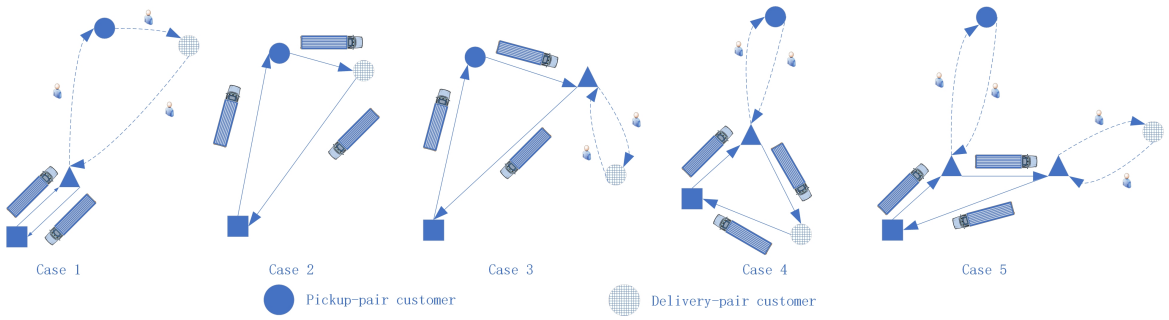

Figure 1: Pickup and delivery cases

These five cases stem mainly from the needs of real scenario. For example, some pickup-delivery-pair goods can only be picked up and/or delivered by a robot.

We make the following assumptions that are common in vehicle routing problems and van-based robot routing problems: (i) each van can only and must carry one robot, and they are one-to-one correspondence, (ii) the robot cannot leave the depot to serve customers directly, (iii) a robot dropped off by a van must be picked up by the same van, (iv) waiting at all locations is costless, (v) the operating time spent dropping off, picking up, replenishing, and swapping a battery is a fixed value, (vi) and the goods can be divided during transportation.

We assume there are specific parking nodes for the vehicles, used for the van to rendezvous, replenish and swap the battery for its robot, rather than performing these operations at customer nodes. This is reasonable because some customer nodes cannot be accessed by the van and some are unsuited to vehicles performing rendezvous operations in the real world. Parking nodes can be visited multiple times by vans/robots. Besides, we assume that rendezvous and other operations take a fixed amount of time.

The robot has a maximum battery capacity. We note that the battery capacity of the robot determines the maximum distance it can travel by itself. Here we choose battery-swap technologies, and the robot can swap its battery when it meets its corresponding van. We assume that every time the van picks up its robot, the robot's battery will be replaced. We also assume that the van has no maximum travel distance restrictions.

The robot can be replenished with goods from the van at parking nodes. We do not consider the case where a van replenishes other vans, or where robots replenish other robots. There are maximum capacities for both the van and the robot. We assume that when the robot is on board the van, the van and its robot's total load cannot exceed the van's maximum capacity. We also allow a robot to visit multiple customers during a dispatch instead of serving only one customer, since the robot's maximum capacity is usually greater than a drone's. In practical applications, there are several independent containers in a robot. After a customer enters the pickup code in the robot, the corresponding containers will open, and the customer gets access to the goods.

Here we assume that each customer node must be visited by exactly one van/robot once. Customer nodes and the depot have their time windows.

We define five types of routes: (i) a van route (*1st-level route*) is a route where a van travels, (ii) a robot route (*2nd-level route*) is a route where a robot travels with its own power, (iii) a van-robot route is a route along which a van carries its robot, (iv) an independent-van route is a route where a van travels independently, and (v) a whole-robot route is a route traveled by a robot.

An example illustrating the problem is given in Figure 2. Triangles represent parking nodes, the square represents the depot, and circles correspond to customer nodes (including pickup-customers, delivery-customers, and pickup-delivery-pair-customers). Green circles correspond to customers that can be served by vans and robots, and yellow circles correspond to customers that can be served only by robots. Solid lines correspond to van routes, dotted lines to robot routes. Circles with P/D/P(1)/D(1) correspond to pickup-customers, delivery-customers, pickup-pair-customer, and delivery-pair-customer, respectively.

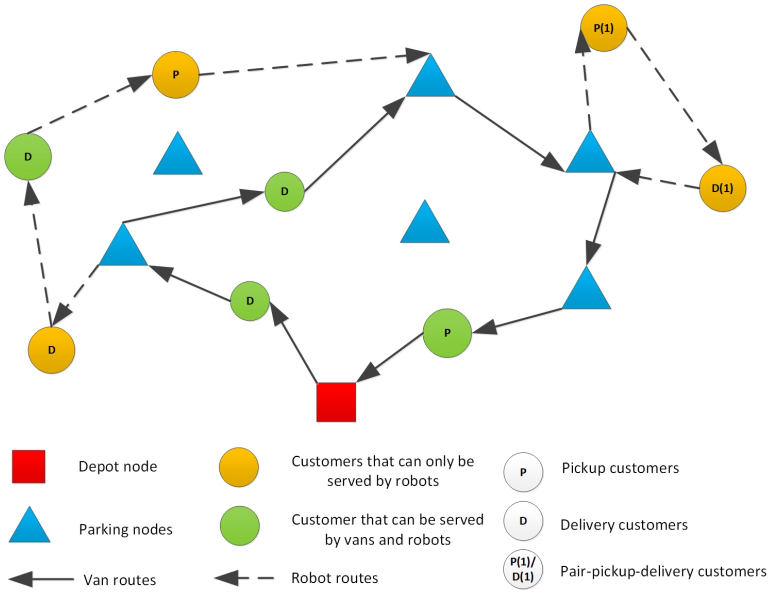

Figure 2: 2E-VRHPD model

3.2. Variable and Parameter Definitions

The problem is defined on a directed graph $G = (V, A)$, where the depot V_0 is represented by two nodes 0 and $0^{'}$. Every van route starts at node 0 and ends at node $0^{'}$. Let V_r be the set of parking nodes where vans can drop off, pick up, replenish and swap the battery for their robots. A parking node can be visited more than once, and we introduce dummy nodes of the parking node allowing to model these multiple visits. Num represents the number of parking nodes in the model.

Let V_{c1} denote the van customer node set and V_c the customer node set, where $V_{c1} \subseteq V_c$. The set $V_\alpha^0, V_\alpha^{'}, V_\alpha^{''}$ is the node set V_α union the depot 0, $0^{'}$, $0 \cup 0^{'}$ respectively, and α can be expressed in a variety of different combinations of customer nodes and parking nodes sets. For example we let $V_{rc1} = V_r \cup V_{c1}$, $V_{rc1}^0 = V_{rc1} \cup V_0$, $V_{rc} = V_r \cup V_c$. We define l couples of pair-customers, where a couple of pair-customers represents a pickup-pair-customer and its corresponding delivery-pair-customer. For pair-customers, let $V_{cp} = \{1, ..., l\}$ denote pickup-pair-customer set and let $V_{cd} = \{l+1, ..., 2*l\}$ denote delivery-pair-customer set. We use $vd(i)$ to be the corresponding delivery-pair-customer of the pickup-pair-customer i. For pickup-customers or delivery-customers, $V_{cpd} = \{2*l+1, ...n\}$. Let $A_1 = \{(i,j) \mid i \in \{0\}; j \in V_{rc1}\} \cup \{(i,j) \mid i, j \in V_{rc1}, i \neq j\} \cup \{(i,j) \mid i \in V_{rc1}; j \in \{0^{'}\}\}$ be the arc set corresponding to the van routes and let $A_2 = \{(i,j) \mid i \in V_r; j \in V_c\} \cup \{(i,j) \mid i, j \in V_c, i \neq j\} \cup \{(i,j) \mid i \in V_c; j \in V_r\}$ be the arc set corresponding to the robot routes. We also let $A_3 = A_1 \cup A_2$ be the arc set corresponding to complete possible robot routes and $A_4 = A_3 \setminus A_1$ be the arc set corresponding to routes that the van cannot reach.

We define c_1 and c_2 as the unit travel cost of vans and robots, and v_1 and v_2 the travel speed of vans and robots, respectively. For each edge, $d_{i,j}$ is the associated travel distance, $c_1 * d_{i,j}$ $(c_2 * d_{i,j})$ is the associated travel cost and $d_{i,j}/v_1$ $(d_{i,j}/v_2)$ is the associated travel time for the van (robot). The freight must be delivered from the depot $\{0\}$ to customer nodes i, or from pickup node V_{cp} to the corresponding delivery node V_{cd}, with the demand δ_i. Variables s_i refer to the service time at customer nodes, depot, or parking nodes i. Note the service time at the depot or parking nodes is a fixed value (as fixed operational time). We let $\delta_i < 0$ for pickup-customers and let $\delta_i > 0$ for the delivery-customers, and $\delta_i = -\delta_{i-l}$ for the pickup-delivery-pair customers. The time window of the customer nodes $i \in V_c$ is $[a_i, b_i]$, is the time interval where the service at node i is permitted to start. Let $[a_0, b_0] = [a_{0'}, b_{0'}]$, where a_0 and $a_{0'}$ represent the earliest possible departure time from the depot 0 and $0'$ respectively. b_0 and $b_{0'}$ are the latest possible arrival times at the depot 0 and $0'$ respectively. These time windows are hard. $F_T = \{1, 2, .., k, .., K\}$ is the set of vans, where K is the number of vans, and k represents the kth van. Because vans and robots are in one-to-one correspondence, it also corresponds to the set of robots. Let M be an arbitrary large constant number. Let C_1 be the maximum capacity for the van and C_2 the maximum capacity for the robot. Note that when the robot is on board the van, the total load of the van and its carried robot can not exceed C_1. Let T be the fixed operational time at depot or parking nodes. Let G be the maximum battery capacity for the robot. Let h be the charge consumption rate of the robot. In addition, we introduce the following decision variables.

Let $x_{i,j,k}$ be equal to 1 if arc (i,j) in A_1 is traveled by the $kth-van$, 0 otherwise.

Let $y_{i,j,k}$ be equal to 1 if arc (i,j) in A_3 is traveled by the $kth-robot$, 0 otherwise.

Let $Q_{i,j,k}$ be equal to 1 if arc (i,j) in A_1 is traveled by the $kth-van$ with its robot on board, 0 otherwise.

Let e_i^k be the remaining battery capacity of the robot at node i on arrival.

Let W_i^k be the last arrival time for the $kth-van$ (or/and $kth-robot$) at node i.

Let $p_{i,j,k}$ be the freight flow of the robot in arc (i,j) in A_2.

Let $P_{i,j,k}$ be the freight flow of the van in arc (i,j) in A_1. If there is robot on board the van, let $P_{i,j,k}$ be the freight flow of the van and its robot.

Let w_i^k be an auxiliary binary decision variable in parking nodes, which can be equal to 0 or 1. This variable has no practical significance.

Let $Setp_i^k$ be the binary variable for the subroute. If there is a couple of pair-customers in the independent-van route/robot route, $Setp_i^k = 1$, 0 otherwise, with i being a pickup-pair-customer.

Let $pp_{i,j}^k$ be the specific freight flow of the robot in arc (i, j) in A_2. The specific freight flow of the robot is defined as the robot flow being carried from the parking nodes. Only when the goods carried from the parking nodes are delivered, the value of the robot's specific freight flow will change.

Let $PP_{i,j}^k$ be the specific freight flow of the van in arc (i, j) in A_1. The specific freight flow of the van is defined as the van flow being carried from the parking nodes. Only when the goods carried from the parking nodes are delivered, the value of the van's specific freight flow will change.

Figure 3 is a simple example to distinguish specific freight flow and freight flow. For specific freight flow, the robot departs from the parking node with specific freight flow 10 to deliver delivery-pair-customer 1. After serving the delivery-pair-customer 1, the specific freight flow becomes 0. For freight flow, the robot departs from the parking node with freight flow 10 to deliver delivery-pair-customer 1. After visiting pickup-pair-customer 2 and 3, the freight flow become 20 and 30, and after visiting delivery-pair-customer 3 and 1, the freight flow become 20 and 10.

We want to jointly use freight flow and specific freight flow to forbid some cases in our model. Detailed instructions are given in Section 3.3.4.

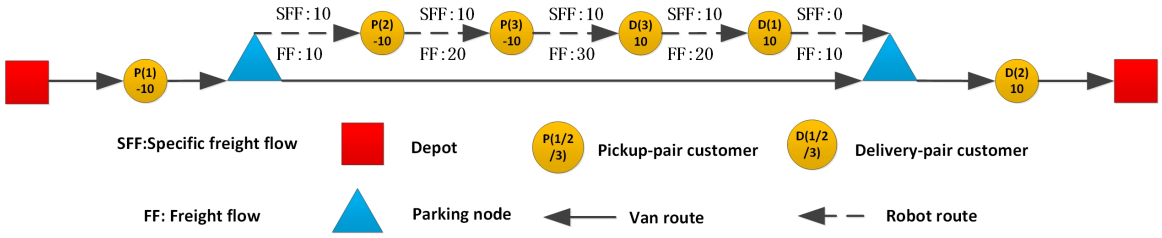

Figure 3: A simple example of specific freight flow and freight flow

3.3. Mixed Linear Integer Programming Model

3.3.1. Objective

$$\min\Big(\sum_{k\in F_T}\sum_{(i,j)\in A_1} c_1 * d_{i,j} * x_{i,j,k} + \sum_{k\in F_T}\sum_{(i,j)\in A_3} c_2 * d_{i,j} * y_{i,j,k} - \sum_{k\in F_T}\sum_{(i,j)\in A_1} c_2 * d_{i,j} * Q_{i,j,k}\Big) \tag{1}$$

The objective function (1) minimizes the total travel cost. It corresponds to the van route cost plus the whole-robot route cost, minus the van-robot route cost.

3.3.2. Arc constraints

Pure van/robot arc constraints guarantee which route type is allowed and which route type is not allowed. The arc constraints are described in Figure A.10.

$$\sum_{(i,j)\in A_1} x_{i,j,k} \leq 1, \forall j \in V_{rc1}, k \in F_T \tag{2}$$

$$\sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(j,i)\in A_1} x_{j,i,k} = 0, \forall j \in V_{rc1}, k \in F_T \tag{3}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} \leq 1, \forall j \in V_{rc}, k \in F_T \tag{4}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} - \sum_{(j,i)\in A_3} y_{j,i,k} = 0, \forall j \in V_{rc}, k \in F_T \tag{5}$$

$$\sum_{i \in V_{rc1}} x_{i,0',k} = \sum_{j \in V_{rc1}} x_{0,j,k} = \sum_{i \in V_{rc1}} y_{i,0',k} = \sum_{j \in V_{rc1}} y_{0,j,k} \leq 1, \forall k \in F_T \tag{6}$$

$$\sum_{k \in F_T} (\sum_{(i,j) \in A_3} y_{i,j,k} + \sum_{(i,j) \in A_3} x_{i,j,k} - \sum_{(i,j) \in A_3} Q_{i,j,k}) = 1, \forall j \in V_c \tag{7}$$

$$\sum_{(i,j) \in A_3} y_{i,j,k} + \sum_{(i,j) \in A_1} x_{i,j,k} - \sum_{(i,j) \in A_1} Q_{i,j,k} \leq 1, \forall i \in V_{c1}^0, k \in F_T \tag{8}$$

$$\sum_{(i,j) \in A_3} y_{i,j,k} + \sum_{(i,j) \in A_1} x_{i,j,k} - \sum_{(i,j) \in A_1} Q_{i,j,k} \leq 1, \forall j \in V_{c1}^{'}, k \in F_T \tag{9}$$

$$\sum_{(i,j) \in A_3} y_{i,j,k} \leq \sum_{(i,j) \in A_1} x_{i,j,k}, \forall i \in V_r, k \in F_T \tag{10}$$

$$2 * Q_{i,j,k} \leq x_{i,j,k} + y_{i,j,k} \leq 2 * Q_{i,j,k} + 1, \forall (i,j) \in A_1, k \in F_T \tag{11}$$

$$x_{i,j,k}, y_{i,j,k}, Q_{i,j,k} \in \{0,1\}, \forall (i,j) \in A_3, k \in F_T \tag{12}$$

$$Q_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{13}$$

$$x_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{14}$$

$$\sum_{(i,j) \in A_3} x_{i,j,k} + \sum_{(i,j) \in A_3} y_{i,j,k} - \sum_{(i,j) \in A_3} Q_{i,j,k} = \sum_{(j,vd(i)) \in A_3} x_{j,vd(i),k} + \sum_{(j,vd(i)) \in A_3} y_{j,vd(i),k} - \sum_{(j,vd(i)) \in A_3} Q_{j,vd(i),k}, \quad \forall i \in V_{cp}, k \in F_T \tag{15}$$

Constraints (2)-(14) are pure van/robot arc constraints. Constraints (2)-(6) ensure that every node is visited by a van/robot at most once, and the departure times are equal to the arrival times. Constraints (6) force the number of a van leaving the depot to be equal to that of its robot leaving the depot, and equal to the number of the van/robot coming back to the depot. Constraints (7) ensure that every customer node is visited by vans or robots exactly once. Constraints (8)-(9) specify for customer nodes and depot, that a van and its robot cannot visit the same node unless the robot is on board the van. Constraints (10) ensure a robot cannot visit a parking node unless its van visited this node, and vice versa. Constraints (11) let $Q_{i,j,k}$ be equal to 1 only if $x_{i,j,k}$ and $y_{i,j,k}$ are all equal to 1. Constraints (12) are the binary variable constraints. Constraints (13)-(14) force the arc variable to be equal to 0 where they are not allowed to visit.

Constraints (15) are new constraints ensuring that the origin and destination nodes of a request are visited by the same mothership vehicles (van and/or robot).

3.3.3. Time constraints

Van and robot can serve different customers in parallel. However, the case where a van picks up/delivers a pair-customer and its robot delivers/picks up the corresponding pair-customer in a parallel routes cannot occur, even if the mothership visits the pickup-pair-customer earlier than the corresponding delivery-pair-customer. Figure 4 is a simple example of such a forbidden case. Here we use time constraints to forbid cases where a van and its robot serve a couple of pair-customers in a parallel routes.

$$W_i^k + s_i + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall \{i \in V_{rc1}^0 | (i,j) \in A_1\}, k \in F_T \tag{16}$$

$$W_i^k + s_i + d_{i,j}/v_2 - W_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall \{i \in V_{rc} | (i,j) \in A_2\}, k \in F_T \tag{17}$$

$$W_i^k + s_i - W_j^k \leq M(1 - y_{i,j,k}), \forall \{(i,j) \in A_3\}, k \in F_T \tag{18}$$

$$W_i^k \leq W_{i+l}^k - s_i - d_{i,i+l}/v_1, \forall \{i \in V_p\}, k \in F_T \tag{19}$$

$$a_i \leq W_i^k \leq b_i, \forall i \in V_c^{''}, k \in F_T \tag{20}$$

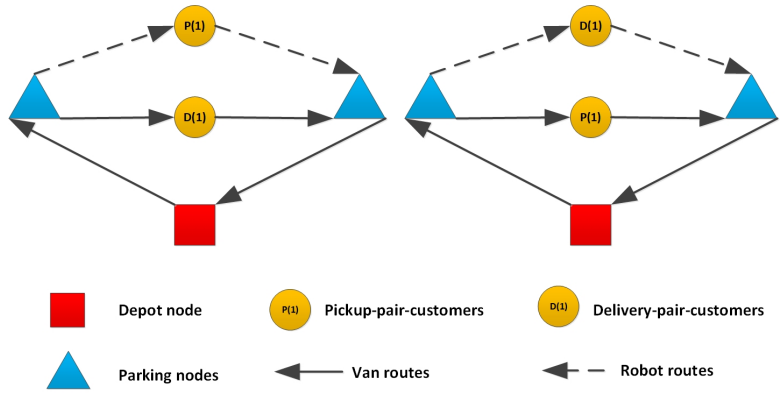

Figure 4: Forbidden cases for serving a couple of pair-customers in parallel routes

$$\sum_{(i,o)\in A_2}(y_{i,o,k}-Q_{i,o,k})-\sum_{(i+l,o)\in A_1}(x_{i+l,o,k}-Q_{i+l,o,k}))=0\Rightarrow W_i^k\le W_j^k\le W_{i+l}^k,\exists j\in V_r,\forall i\in V_p,k\in F_T \tag{21}$$

$$\sum_{(i,o)\in A_1}(x_{i,o,k}-Q_{i,o,k})-\sum_{(i+l,o)\in A_2}(y_{i+l,o,k}-Q_{i+l,o,k}))=0\Rightarrow W_i^k\le W_j^k\le W_{i+l}^k,\exists j\in V_r,\forall i\in V_p,k\in F_T \tag{22}$$

Constraints (16-22) are time-related constraints. Constraints (16) model the time flow of the van route. Constraints (17) represent the time flow constraints of the robot route. Constraints (18) model the time flow of the whole-robot route leaving the parking node. Constraints (19) ensure precedence constraints for the visiting of pickup-customers and delivery-customers. Constraints (20) enforce time window constraints for all nodes.

Constraint (21-22) model time and arc constraints to forbid the cases in Figure 4. Constraints (21) ensure that when a pickup-pair-customer is in a robot route, and its corresponding delivery-pair-customer is in an independent-van route, there exists at least one parking node visited by van after the pickup-pair-customer but before the delivery-pair-customer. Constraints (22) ensure that when a delivery-pair-customer is in a robot route, and its corresponding pickup-pair-customer is in an independent-van route, there exists at least one parking node visited by van after the pickup-pair-customer but before the delivery-pair-customer.

3.3.4. Freight and energy constraints

Drexl (2020) described capacity considerations on subroutes for a one to one pickup and delivery truck and trailer routing problem. Drexl (2020) pointed out that the following two quantities are relevant: the minimal truck load at decoupling and the subroute load balance at each position. Here we present a freight flow mathematical model for our 2E-VRHPD problem, which can also be used in truck and trailer routing problems.

If there are pair-customers in the subtour (van/robot route between two parking nodes), the freight flow may differ from the case where there are no pair-customers present. Figure 5 is a simple example of the freight flow in a robot route, with a pickup-pair-customer and a delivery-pair-customer in the robot subroute. Case 1 is a forbidden situation, and case (ii) is a permitted situation. We assume the capacity of the robot is 50. In case (i), there is no couple of pair-customers in the subroute, and the robot should load 40 when leaving the parking node. Before serving delivery-pair-customer 1, the robot should first serve pickup-pair-customer 2, so the robot route's freight flow exceeds the robot's maximum capacity. Hence case (i) is forbidden. In case (ii), there are a couple of pair-customers in the subroute, and the robot could load 0 capacities when leaving the parking node. The robot could serve a pickup-pair-customer 1 and then serve the corresponding delivery-pair-customer 1, so the freight flow in the robot route will not exceed the robot's maximum capacity of 50. Hence case (ii) is permitted.

We define 'special freight flow' and jointly use freight flow and special freight flow to forbid case (i) in Figure 5.

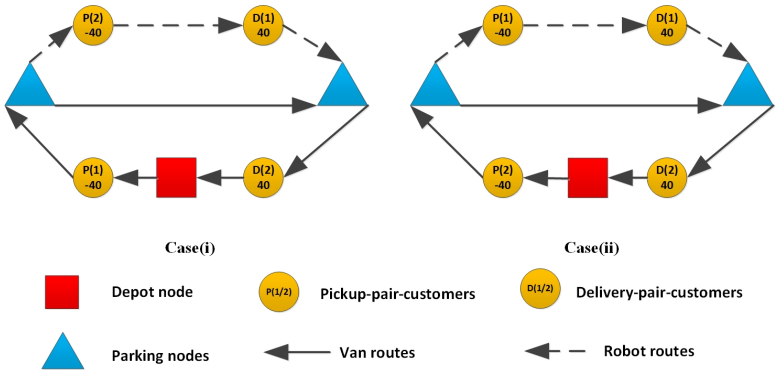

Figure 5: An example of freight flow in robot route

Freight constraints

$$\sum_{(i,j)\in A_2} p_{i,j,k} - \sum_{(j,i)\in A_2} p_{j,i,k} = \delta_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in V_c, k \in F_T \tag{23}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} - \sum_{(j,i)\in A_1} P_{j,i,k} = \delta_j * \sum_{(i,j)\in A_1} x_{i,j,k}, \forall j \in V_{c1}, k \in F_T \tag{24}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} - \sum_{(j,i)\in A_1} P_{j,i,k} = \sum_{(i,j)\in A_2} p_{i,j,k} - \sum_{(j,i)\in A_2} p_{j,i,k}, \forall j \in V_r, k \in F_T \tag{25}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} + \sum_{(i,j)\in A_1} p_{i,j,k} \leq C_1, \forall j \in V_r, k \in F_T \tag{26}$$

$$0 \leq p_{i,j,k} \leq C_2 * (1 - Q_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{27}$$

$$0 \leq p_{i,j,k} \leq C_2 * (y_{i,j,k} - Q_{i,j,k}), \forall (i,j) \in A_2, k \in F_T \tag{28}$$

$$0 \leq P_{i,j,k} \leq C_1 * (x_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{29}$$

$$Setp^k_{i+l} = 0 \Rightarrow W^k_i \leq W^k_j \leq W^k_{i+l}, \forall i \in V_p, k \in F_T, \exists j \in V_r \tag{30}$$

$$Setp^k_{i+l} = 1 \Rightarrow W^k_i \leq W^k_j \leq W^k_{i+l}, \forall i \in V_p, k \in F_T, \nexists j \in V_r \tag{31}$$

$$PP_{i,j,k} = P_{i,j,k}, \forall i \in \{V_r | (i,j) \in A_1\}, k \in F_T \tag{32}$$

$$pp_{i,j,k} = p_{i,j,k}, \forall i \in \{V_r | (i,j) \in A_2\}, k \in F_T \tag{33}$$

$$0 \leq PP_{i,j,k} \leq C1 * x_{i,j,k}, \forall (i,j) \in A_1, k \in F_T \tag{34}$$

$$0 \leq pp_{i,j,k} \leq C_2 * (1 - Q_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{35}$$

$$0 \leq pp_{i,j,k} \leq C_2 * (y_{i,j,k} - Q_{i,j,k}), \forall (i,j) \in A_2, k \in F_T \tag{36}$$

$$\sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = 0, \forall j \in V_p \cup \{V_{dp} | \delta_j \leq 0\}, k \in F_T \tag{37}$$

$$\sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = \delta_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in \{V_{dp} | \delta_j > 0\}, k \in F_T \tag{38}$$

$$Setp^k_j = 1 \Rightarrow \sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = 0, \forall j \in V_d, k \in F_T \tag{39}$$

$$Setp^k_j = 0 \Rightarrow \sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = \delta_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in V_d, k \in F_T \tag{40}$$

$$\sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = 0, \forall j \in V_p \cup \{V_{dp} | \delta_j \leq 0\} \cap V_{c1}, k \in F_T \tag{41}$$

$$\sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = \delta_j * \sum_{(i,j)\in A_1} (x_{i,j,k}), \forall j \in \{V_{dp} | \delta_j > 0\} \cap V_{c1}, k \in F_T \tag{42}$$

$$Setp_j^k = 1 \Rightarrow \sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = 0, \forall j \in V_d \cap V_{c1}, k \in F_T \tag{43}$$

$$Setp_j^k = 0 \Rightarrow \sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = \delta_j * \sum_{(i,j)\in A_1} (x_{i,j,k}), \forall j \in V_d \cap V_{c1}, k \in F_T \tag{44}$$

Energy constraints

$$e_j^k + d_{i,j} * h - e_i^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall \{i \in V_{rc} | (i,j) \in A_2\}, k \in F_T \tag{45}$$

$$0 \leq e_i^k \leq G, \forall i \in V_{rc}, k \in F_T \tag{46}$$

Constraints (23-29) model freight flow constraints. Constraints (23) represent the freight flow in the robot route. Constraints (24) represent the freight flow in the van route. Constraints (25) ensure the conservation of cargo flow at parking nodes. Constraints (26) ensure that at the parking node the capacity of the van and the robot is less than the maximum capacity of the van. Constraints (27-29) are the variable constraints.

Constraints (30-44) are new freight flow constraints for collaborative distribution of vehicles. Constraints (30-31) are used to mark whether there is a couple of pair-customers in a subroute (independent-van/robot route). Constraints (30) ensure that if binary variable $Setp_{i+l}^k = 0$ for a delivery-pair-customers in the subroute, then there are no couples of pair-customers in the subroute. In other words, there exists at least one parking node visited by a van after the pickup-pair-customer but before the corresponding delivery-pair-customer. Constraints (31) force that if binary variable $Setp_{i+l}^k = 1$ for a delivery-pair-customers in the subroute, then there is a couple of pair-customers in the subroute. In other words, there is no parking node visited by van after the pickup-pair-customer but before the corresponding delivery-pair-customer.

Constraints (32-36) represent the continuous variable constraints for the specific freight flow of the robot route. Constraints (32-33) ensure that when the van/robot leaves the parking node, the specific freight flow is equal to the freight flow. Constraints (34) ensure the specific freight flow of a van does not exceed it's maximum capacity. Constraints (35-36) ensure the specific freight flow of a robot does not exceed its maximum capacity. Constraints (37) ensure that when the robot visits a pickup-customer (or pickup-pair-customer), the specific freight flow keeps unchanged. Constraints (38) ensure that when the robot visits a delivery-customer, the specific freight flow change is equal to the weight of goods. Constraints (39) ensure that when the robot visits a delivery-pair-customer, and its corresponding pickup-pair-customer is also in the subroute, the specific freight flow remains unchanged. Constraints (40) ensure that when the robot visits a delivery-pair-customer, and its corresponding pickup-pair-customer is not in this subroute, the change in the specific freight flow is equal to the weight of goods.

Constraints (41-44) model the continuous variable constraints for the specific freight flow of the van route. Constraints (41) ensure that when a van visits a pickup-customer (or pickup-pair-customer), the specific freight flow keeps unchanged. Constraints (42) ensure that when a van visits a delivery-customer, the specific freight flow change is equal to the weight of goods. Constraints (43) ensure that when a van visits a delivery-pair-customer, and its corresponding pickup-pair-customer is also in the subroute (independent-van route), the specific freight flow remains unchanged. Constraints (44) ensure that when a van visits a delivery-pair-customer, and its corresponding pickup-pair-customer is not in this subroute (independent-van route), the change in the specific freight flow is equal to the weight of goods.

Constraints (45-46) represent energy constraints for the robot. Constraints (45) are energy flow constraints in the robot route, and constraints (46) ensure the robot's battery level is always larger than or equal to 0, smaller than or equal to the robot's maximum battery capacity at the parking node and customer node.

3.3.5. Constraint linearization

Constraints (21-22) and constraints (30-31) can be linearized by using the binary variable w_i^k in parking nodes as follows. For example, constraints (47-49) can express the constraints (21).

$$W_i^k - W_j^k - M * w_j^k \leq 2 * M * Setp_{i+l}^k, \forall i \in V_p, j \in V_r, k \in F_T \tag{47}$$

$$W_j^k + M * w_j^k - W_{i+l}^k \leq 2 * M * Setp_{i+l}^k, \forall i \in V_p, j \in V_r, k \in F_T \tag{48}$$

$$\sum_{i \in V_r} w_i^k \leq Num - 1, \forall k \in F_D \tag{49}$$

Constraints (22), (30), and (31) can be expressed using the same linearization methods.

4. Adaptive Large Neighborhood Search

We propose an adaptive large neighborhood search (ALNS) algorithm to solve the 2E-VRHPD problem. The main idea of ALNS is to iteratively apply a set of removal and insertion operators on an initial solution until the best solution is found (Ropke and Pisinger 2006, Mourad et al. 2020).

Since our problem is composed of pickup and delivery, two-echelon, van no-go zones, and VRPD components, we draw on some operations used in (Ropke and Pisinger 2006, Mühlbauer and Fontaine 2021, Anderluh et al. 2021, Sacramento et al. 2019). In our problem, we have parking stations for the vehicle, battery, and cargo transshipment, and the parking station is an independent station that can be visited multiple times. We also need to consider the pickup and delivery in the vehicle collaborative distribution scenario during destroy-repair operations. We focus mainly on forbidding the cases in Figure 4 and addressing the capacity constraints described in Figure 5. The detail of our proposed ALNS algorithm is described in Appendix B.

The feasibility of a given route is also important in the ALNS algorithm. We focus mainly on the capacity feasibility in this section, since the time and energy feasibility of a given route is straightforward.

4.1. Initial solution

We start with a simple heuristic to generate initial feasible solutions. A pseudocode for the construction of the initial solution is shown in Appendix C.

At the beginning of the heuristic, all customers are in an un-assigned customer list. We then repeat the following two main steps until all the customer nodes are assigned.

1. We randomly choose a customer node in the un-assigned customer list. If the chosen node is a pickup/delivery-customer node and can only be served by robots, we construct a route with van routes and robot routes. If the chosen node is a pickup/delivery-customer node and can both be served by vans and robots, we decide whether to construct a route with van routes and robot routes or to construct a route with only van routes, according to a roulette-wheel mechanism. If the chosen customer is a pickup/delivery-pair-customer, we first find its corresponding pickup/delivery-pair-customer. We then construct a route based on the pickup-pair-customer. Next, we insert the corresponding delivery-pair-customer into the random-feasible position of the constructed route. If the delivery-pair-customer can not be inserted into the constructed route, we try to build a new robot route from the constructed route.
2. The second step is to randomly choose a customer node in the un-assigned customer list to insert into the constructed route's random-feasible position until no customer node can be inserted. We note that for the pair-customer, we need to insert customers according to the precedence constraint in order. For example, in the process of a van launching and recycling its robot, the van serving a pickup/delivery pair-customer and its robot serving a corresponding customers is not allowed.

To this end, the feasibility of the returned solution, in terms of request time windows, capacity, energy, reachability synchronization, couple and precedence constraint, is assured. This initial feasible solution can then be improved by the ALNS operators. We describe the removal and insertion operators used by the ALNS algorithm in the following subsections.

4.2. Destroy operators

The destroy operation destroys a chosen 2E-VREC route by removing nodes or routes. According to the structure of the 2E-VRHPD route, we start with customer nodes, pair-customer nodes, parking nodes, and routes to define six destroy operators:

1. **Random customer removal (D1)** operator randomly removes customer nodes from a 2E-VRHPD route. If the removed customer is a pair-customer, we remove its corresponding pair-customer node.
2. **Greedy customer removal (D2)** operator removes the customer that can yield the largest cost reduction for a given route. If the removed customer is a pair-customer, we first remove the selected pickup/delivery-pair-customer node and then remove its corresponding pair-customer node.
3. **Pair-customer removal (D3)** operator removes a couple of pair-customers (pickup-pair-customer and its corresponding delivery-pair-customer) that can yield the largest cost reduction for a given route.
4. **Station-route removal (D4)** operator randomly removes a parking station, and the robot routes depart from and arrive at this station.
5. **Random route destruction (D5)** operator randomly selects a 2E-VRHPD route in the solution to destroy.
6. **Greedy route destruction (D6)** operator selects a 2E-VRHPD route,with a minimum number of customer nodes to destroy.

Note that the largest cost reduction for a given route is computed as follows. We first calculate the cost difference between the original route and the route after removing nodes. And then we chose the largest cost difference one.

4.3. Repair operators

The route repair operator repairs the existing routes or constructs a new 2E-VRHPD route if needed. The repair operations do not allow infeasible solutions but allow worse solutions through simulated annealing evaluation and calibration. According to the structure of the 2E-VRHPD route, we start with customer nodes, parking nodes, and routes to define five repair operators:

1. **Route reconstruction (R1)** operator ensures that we always get a viable solution. We adopt the same method as used to get the initial solution in Section 4.1.
2. **Random customer insertion (R2)** operator randomly chooses a customer to insert into a random-feasible position in a given route until all customers have been tried. If the inserted customer is a pickup/delivery-pair-customer, we randomly insert its corresponding delivery/pickup-pair-customer node after/before this customer.
3. **Greedy customer insertion (R3)** operator randomly chooses a customer to insert into a position in the route with the least cost increases. If the inserted customer is a pickup/delivery-pair-customer, we greedy insert its corresponding delivery/pickup-pair-customer node after/before this customer.
4. **Random station-route insertion (R4)** operator randomly chooses a station to insert into a given route and then generates a robot route from this station with one customer node. If the customer is a pickup/delivery-pair-customer, we randomly insert its corresponding delivery/pickup-pair-customer node after/before this customer.

5. **Greedy station-route insertion (R5)** operator chooses a station to insert into a given route and then generates a robot route from this station with one customer node, with least total travel cost increases. If the customer is a pickup/delivery-pair-customer, we greedy insert its corresponding delivery/pickup-pair-customer node after/before this customer.

4.4. Capacity feasibility tests

We draw on the capacity feasibility checking approach of (Drexl 2020), who studied the one-to-one pickup-and-delivery problem with time windows and trailers. A 2E-VRHPD route usually includes single van-robot routes and parallel routes (contains the independent-van route and robot route). For a single van-robot route, the capacity feasibility test is straightforward, so we focus on testing the capacity feasibility of the parallel routes.

A parallel path is capacity-feasible if and only if the capacity in the parking nodes, together with the independent-van route and robot route, are all feasible. At each parking node for dropping off or picking up the robot, the van and robot's total capacity should be less than the van's total capacity. In a parallel path, capacity considerations on the independent-van route and robot route, the following four amounts usually need to be considered:

(i) Amount of goods a robot must carry before leaving the parking nodes. This load includes the delivery-customers and the delivery-pair-customers whose corresponding pickup-pair-customer is not in this robot route.

(ii) Amount of goods a robot can carry at most before leaving the parking nodes. This load should be as large as possible provided the goods do not exceed the robot's capacity on each customer node during a robot route.

(iii) Amount of goods a van must carry before leaving the parking nodes. This load includes the delivery-customers and the delivery-pair-customers whose corresponding pickup-pair-customer is not in this independent-van route.

(iv) Amount of goods a van can carry at most before leaving the parking nodes. This load should be as large as possible provided the goods do not exceed the van's capacity on each customer node during an independent-van route.

One easy way to check the capacity feasibility of a parallel route is to ensure the robot capacity in the robot route and let the robot carry as many goods as possible subject to ensuring the van capacity feasibility for the independent-van-route. We then check the capacity feasibility in the parking node used for picking up the robot. The proposed parallel route capacity testing algorithm is described in Appendix D Algorithm 3.

5. Computational study

We performed extensive computational experiments to assess our solution method's performance and evaluated the influence of parking node density on solutions. First, we explain how we generate test instances, and we describe the different parameters used in Section 5.1. Second, we compare the performances of the operators in Section 5.2. Third, we compare our ALNS results with CPLEX results to assess our algorithm's performance and assessed the influence of parking node density on the solution in Section 5.3.

The mathematical programming model was coded in DOCPLEX by calling CPLEX 12.8 to solve the MIP model. The ALNS was coded in python version 3.6.8. Both CPLEX and python were executed on an Intel(R) Core(TM) 2.8 GHz processor with 16 GB of memory running under Windows 10. Python was run with single-threading.

5.1. Instance generation and parameters

For testing the proposed solution approach, we used the instances of Dellaert et al. (2019) and Yu et al. (2020) as a basis. The specific instance generation approach is described in those two articles.

We set the speed of the van at 2, and the speed of the robot at 1. The maximum capacity of a robot was 50, and the capacity of a van was 200. When the robot was on board the van, the van and robot's total maximum capacity was 200. The maximum battery level of a robot was 100. The robot's energy consumption rate was 1 unit per distance. The depot time window was [0,500]. Vans could access two-thirds of the total customer nodes in each instance. If the calculated number of van customer nodes was fractional, we rounded up. The first two-thirds of customers in the instance were the van customers. We chose $2 * \lceil (n/8) \rceil$ customers as the pair-customers, in which n was the total number of customers. When choosing a couple of pair-customers, we ensured that a van and its robot could feasibly visit a pair of pickup-delivery-pair-customers.

The set of parameters used in the computational study along with their descriptions and values are presented in Appendix E table E.12.

5.2. Sensitivity analysis on operators

Although the ALNS algorithm can automatically adjust the operator's weight, if we remove the operators that do not contribute much to the optimal solution but are time-consuming, the ALNS algorithm's efficiency can be significantly improved.

We performed a sensitivity analysis on the destroy and repair operators. Table 2 reports the results of our sensitivity analysis on operators. Row 1 and Row 3 indicate the different combinations of operators. The base-operators contain all destroy and repair operators, and the next columns represent the combinations of some operators, respectively. They are NO D1: without random customer removal in base-operators; No D2: without greedy customer removal in base-operators; No D3: without pair-customer removal in base-operators; No D4: without station-route removal in base-operators; No D5: without random route destruction in base-operators; No D6: without greedy route destruction in base-operators; No R1: without route reconstruction in base-operators; No R2: without random customer insertion in base-operators; No R3: without greedy customer insertion in base-operators; No R4: without random station-route insertion in base-operators; No R5: without greedy station-route insertion in base-operators; No R4 and R5: without random station-route insertion and greedy station-route insertion in base-operators. E1 is the objective function gap between ALNS with all operators enabled and the various settings leaving out a single operator or two operators. E2 is the solving time gap between those scenarios.

Table 2: Sensitivity analysis on the operators

	Base-operators	Gap	No D1	No D2	No D3	No D4	No D5	No D6
Average cost	539.3	E1	0.52%	0.63%	-0.08%	-0.03%	0.26%	0.05%
Average time	115.5	E2	0.06%	-6.38%	-1.18%	9.75%	-0.90%	-2.91%
	Base-operators	Gap	No R1	No R2	No R3	No R4	No R5	No R4 and R5
Average cost	539.3	E1	0.08%	-0.24%	9.04%	0.62%	1.41%	7.36%
Average time	115.5	E2	9.07%	7.66%	64.01%	-46.57%	-46.60%	-55.05%

From table 2 we can observe that removing the operator can hardly improve the solution's quality. Removing destroy operators does not have a significant effect on the solving time. Operator R3 is very important to the ALNS algorithm since removing R3 will increase the solving time and make the solution worse. Removing the operator R4 or R5 can reduce the solution time, but the solution's quality will slightly deteriorate. Removing the operator R4 and R5 can reduce the solution time, but the solution's quality will deteriorate.

Next, we studied all operators' computational performance to see how often an operator improves a solution. We drew on the analysis method Chen et al. (2021) did. We defined a metric %IBest/%Usage to measure operators' performance, where %IBest is the percentage of iterations in which a new feasible solution is better than the current best solution,

and%Usage is the rate of total iterations in which a destroy/repair operator is utilized. This metric helps to find out how often an operator improves a solution.

Table 3 reports the metric %IBest/%Usage analysis on operators. Row 1 represent the operators and Row 2 are the metric %IBest/%Usage.

Table 3: Sensitivity analysis on the usage of operators

Operators	D1	D2	D3	D4	D5	D6	R1	R2	R3	R4	R5
%IBest/%Usage	2.19	3.21	1.62	1.57	2.93	2.68	0.65	1.14	4.08	0.55	1.39

From table 3 we can see that for destroy operators, the greedy customer removal (D2) is the one that contributes most to the new best solutions while pair-customer removal (D3) and station-route removal (D4) contribute more to the solution diversity. For repair operators, the greedy customer insertion (R3) has a strong ability to find the new best solutions, while the route reconstruction (R1) and random station-route insertion (R4) are responsible for adding diversity.

Synthesizing table 2 and table 3, we remove operator R4 in the following experiments because this can save computing time, and R4 does not contribute much to finding the optimal solution and does not have much impact on the quality of the solution.

5.3. ALNS experiments

We first compared the ALNS results with the CPLEX results in small-scale instances to see the proposed ALNS algorithm's overall performance. Next, we analyzed the influence of parking node density on the solution and then obtained a suitable number of parking nodes for the model-comparison below.

5.3.1. Comparison of CPLEX and ALNS

The VRPD model is hard to solve for general-purpose MIP solvers (Wang and Sheu 2019). In our problem, we replicated the parking nodes for multiple visiting. Dummy nodes increase the number of nodes in the model, making it more difficult to solve and reduce the problem size we can address.

We tested 20 instances with five customer nodes and three parking nodes using CPLEX 12.8, and used the dummy node iterates-growth steps as follows: We iteratively solved our model by CPLEX, gradually increasing the number of dummy nodes. The increase in the number of dummy nodes stops when no improvement in the solution cost is found. However, if the solution is still worse than the ALNS solution, we continue to increase the number of dummy nodes until the CPLEX solution is better than or as good as the ALNS solution. The iterative process will also stop if the procedure runs out of solving time or out of memory.

For each instance, CPLEX 12.8 runs with default settings until it finds an optimal solution, until it reaches the predetermined maximum computation time (10800 s), or until the program runs out of memory. The computational results for all instances are presented in table 4, in which $CplexK$ represents the number of times a vehicle can visit a parking node, and UB is the upper bounds of CPLEX solutions. E3 is the CPLEX gap between CPLEX upper bound (baseline) and CPLEX lower bound, and TIME is the CPLEX solving time. BC and AC are the best and average solutions for ALNS in 10 tests. SDC and AT are the standard deviation and solving time of ALNS in 10 tests.

Table 4 shows that our ALNS can always reach the best solutions in very small instances. For case CB2, CC4, CD4, ALNS performed better than the CPLEX solver, which ran out of time in K = 3 or K = 4. We note in ALNS solution of case CB2, CC4, and CD4, K = 3, K = 4, and K = 4 dummy parking nodes were replicated, respectively.

Table 4: CPLEX and ALNS results comparison

	Cplex K=2			Cplex K=3			Cplex K=4			ALNS			
	UB	E3(%)	TIME(s)	UB	E3(%)	TIME(s)	UB	E3(%)	TIME(s)	BC	AC	SDC	AT(s)
CA1	262.0	0.0	179.6	262.0	0.0	9801.0	253.3	13.1	10800.0	253.3	253.6	0.0	19.6
CA2	365.8	0.0	784.3	365.8	47.8	10800.0	/	/	/	365.8	365.8	0.0	40.8
CA3	333.8	0.0	5933.6	333.8	53.1	10800.0	/	/	/	333.8	333.8	0.0	34.0
CA4	261.2	0.0	23.2	261.2	0.0	3379.2	261.2	43.4	10800.0	261.2	261.2	0.0	27.6
CA5	171.6	0.0	5.1	167.7	0.0	25.5	167.7	0.0	25.4	167.7	167.7	0.0	19.0
CB1	249.6	0.0	93.6	249.6	0.0	8196.2	249.6	21.3	10800.0	249.6	249.6	0.0	24.4
CB2	393.0	0.0	226.1	403.4	59.0	10800.0	/	/	/	**366.9**	366.9	0.0	47.8
CB3	346.2	0.0	6454.5	346.2	49.7	10800.0	/	/	/	346.2	346.2	0.0	31.2
CB4	261.2	0.0	13.3	261.2	0.0	57.0	261.2	0.0	446.7	261.2	261.2	0.0	29.0
CB5	151.0	0.0	6.9	151.0	0.0	136.7	151.0	45.8	10800.0	151.0	151.0	0.0	16.2
CC1	183.8	0.0	11.2	183.8	0.0	39.4	183.8	0.0	852.8	183.8	183.8	0.0	14.4
CC2	289.5	0.0	241.3	289.5	31.3	10800.0	/	/	/	289.5	289.5	0.0	38.0
CC3	297.7	0.0	1012.3	297.7	41.2	10800.0	/	/	/	297.7	298.7	2.2	52.8
CC4	333.0	0.0	839.6	337.9	41.1	10800.0	/	/	/	**277.6**	277.6	0.0	24.0
CC5	140.7	0.0	4.6	138.3	0.0	35.1	138.3	0.0	80.7	138.3	138.3	0.0	22.2
CD1	276.4	0.0	30.2	276.4	0.0	270.4	276.4	0.0	464.6	276.4	276.4	0.0	20.2
CD2	353.4	0.0	32.5	353.4	0.0	2044.2	353.5	22.1	10800.0	353.5	353.5	0.0	25.8
CD3	271.3	0.0	19.5	271.3	0.0	391.6	271.3	0.0	1623.3	271.3	271.3	0.0	36.8
CD4	253.9	0.0	19.7	253.9	0.0	10356.0	253.9	38.8	10800.0	**231.9**	231.9	0.0	21.4
CD5	151.0	0.0	6.6	151.0	0.0	104.4	/	/	/	151.0	151.0	0.0	15.2
AVER	267.3	0.0	796.9	/	/	/	/	/	/	261.4	261.5	0.1	28.0

5.3.2. Assessment of the influence of parking node density on the solution

As the number of parking points in the city is likely to affect the delivery efficiency of the 2E-VRHPD system, we analyzed the impact of parking node density (the number of parking nodes) changes on the results.

We chose 20 instances with one depot and 15 customers to conduct the sensitivity analysis experiments. The random distribution scenario was adopted for the parking node distribution. To quantify the impact of parking node density, we gradually increased the number of parking nodes from 10, 20, up to 50.

Table 5 reports the influence of parking node density on the solutions. Row 1 shows the number of parking nodes. Row 2 shows the average cost for the instances. Row 3 reports an average percentage gap between the baseline and experimental group solution.

Table 5: Influence of parking node density on the solution

Number of parking nodes	10	20	30	40	50
Average cost	531.5	520.7	507.6	504.5	504.3
Gap with regards to baseline	baseline	-2.0%	-4.5%	-5.1%	-5.1%

Table 5 shows that the cost function of the system presents a rising trend with the increase in the number of parking nodes. Increasing the number of parking nodes from 10 to 30 had a strong effect on reducing the objective function. Increasing the number of parking nodes from 30 to 50 had little impact on reducing the objective function. Therefore, planning an appropriate number of stops can bring the 2E-VRHPD distribution system's efficiency to a high level for city managers.

6. Case study

In this section, we first built realistic scenarios and instances for the case study. We then conducted a sensitivity analysis experiment on the robot's travel cost rate and maximum travel distances and assessed the effect of the restricted area (that are van can not access) on the 2E-VRHPD model. Finally, we compared the 2E-VRHPD model with the other two classical models.

6.1. Scenario construction

We examined our distribution concept, its model, and the proposed solution algorithm on realistic scenarios.

Figure 6 is an example of a realistic scenario (depot and distribution range) for Xi'an city used in our case study. The distribution range (colored in blue) is the southwest corner of Xi'an (China). The depot we chose is marked in red. It is the Xi'an Intelligent Logistics depot of JD.com.

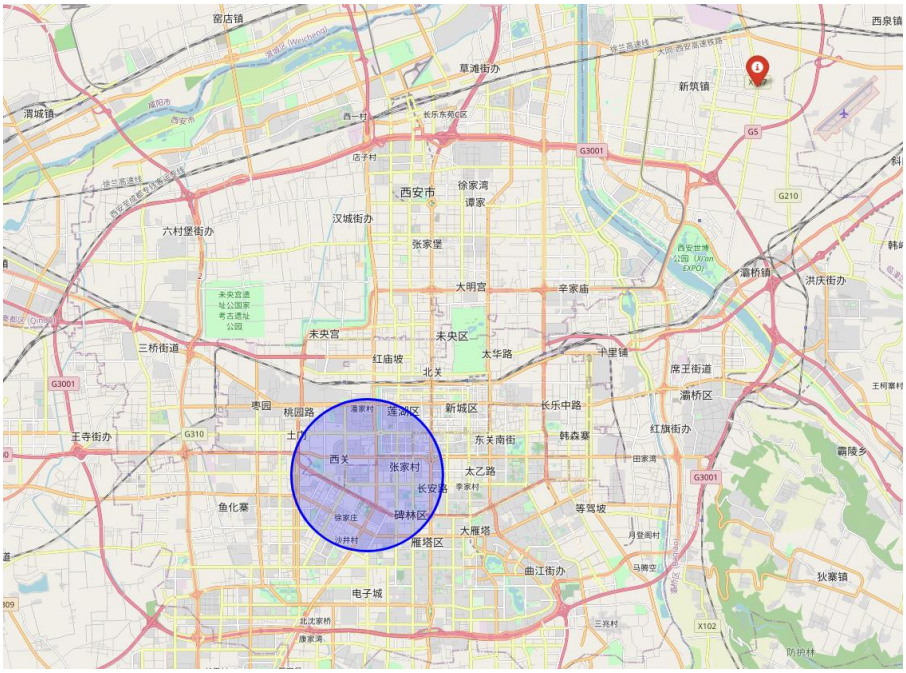

Figure 6: Example of the distribution range and depot in Xi'an city

The van no-go areas, parking nodes, customer nodes, and satellites were set as follows.

Van restricted areas: Van no-go areas were selected within the city walls of Xi 'an and seven universities in the southwest corner of Xi 'an.

Supplier locations: For suppliers, here we only considered four supermarkets, namely Wal-mart, RT-Mart, Yonghui Superstores, and Renrenle Supermarket. The Wal-mart is in the van restricted areas.

Parking node: Some parking nodes were selected near the campuses and city walls, where the van could park. We also randomly selected some parking nodes in the distribution area.

Customer nodes: We randomly generated some customer nodes on the distribution area and ensured that some customers were within van no-go areas, and some customers are pair-customers.

Satellites: The satellites were chosen as the JD.com Xi'an GaoXin satellite, YouYi satellite, and ZaoYuan satellite, responsible for serving our selected distribution range. The satellite was only used in model comparisons part.

Figure 7 is a detailed example of the scenarios of Xi'an city used in our case study. The van no-go zone is filled in the red area on the map. The blue nodes represent the parking nodes we chose, the black dots represent the suppliers, the gray-green dots represent customer nodes, and the orange dots represent satellites.

The basic parameters used in the case study were set as follows.

According to Yu et al. (2020), simply improving the robot's speed can improve the solution, but improvements remain limited in the complex constraints model. We set the robot's speed equal to 5km/h, and the van's speed was 25km/h, suggested by (Yu et al. 2020).

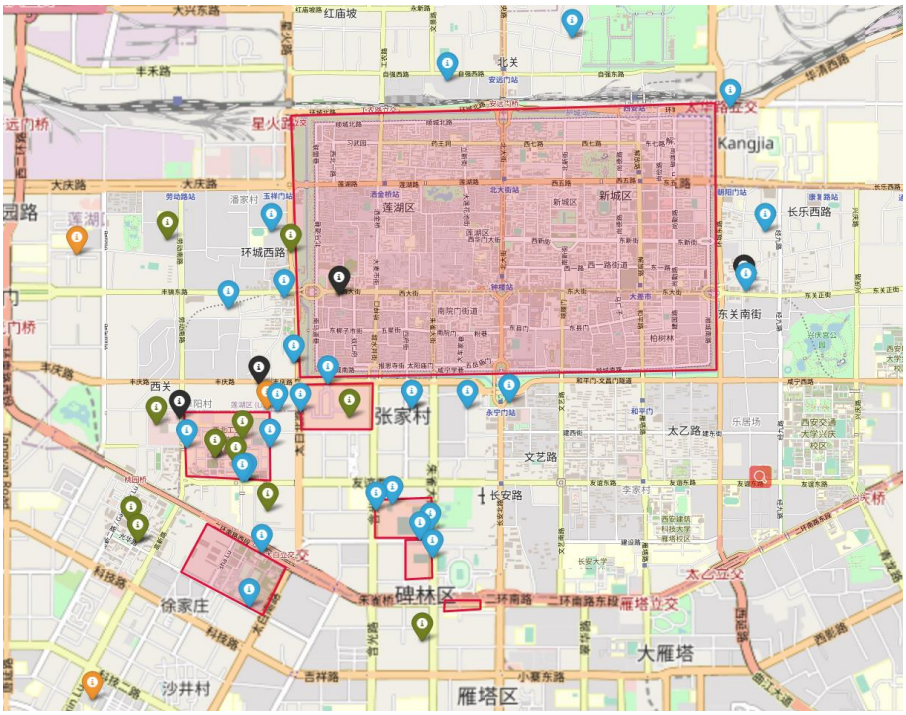

Figure 7: Example of van no-go zone, parking nodes, suppliers, customer nodes, and satellites in Xi'an city

We reset the robot's maximum travel distances equal to 10km. We used artificial customer demands selected in [-30.-20,-10,10,20,30], and vehicle capacities of 50 for robot and 200 for the van. We used a planning horizon of 8 hours. Accordingly, the depot time windows were set to [0,8 h]. We randomly selected typical values for the customer time windows from [30 min, 8 h], and the service times were 10 min. The associated travel cost rate for a van was 3/km, and the robot's associated travel cost rate was 0.3/km since the robot is smaller than the van and electrically powered.

We built ten instances for the following 2E-VRHPD model analysis experiments and model comparison experiments. The instances were generated as follows: Satellite, supplier, and parking node' locations were the same for the ten instances. We set one depot, three satellites, two suppliers, and 15 parking nodes for each instance. For customer nodes, we randomly generated 130 customer nodes in the distribution range and ensured that 3/7 customer nodes were in the van no-go zone. These 130 customer nodes constituted the customer node pool. We then randomly chose 28 customer nodes from the customer node pool and two supplier nodes from suppliers (as customer nodes) for each instance. We ensured there were 13 customers in the van no-go zone for each instance. We also ensure there are 7 couple of pair-customers, 16 pickup/delivery-customers. The instances used in this paper can be download from https://www.researchgate.net/publication/350605064.

6.2. 2E-VRHPD model analysis experiments

In 2E-VRHPD model analysis experiments, we analyzed the impact of the travel cost rate (TCR) for robots, the maximum travel distance (MTD) of the robot, and the travel cost rate and the maximum travel distance combination (TCR-MTD) of the robot on the mode-computed outputs. We also compared the effects of no-go zones on the solution: we assumed the van could also access the no-go areas from the original model and called it 2E-VRHPD-NR, and compared it with the 2E-VRHPD model, to see how the no-go areas affect the performance of the system.

The experiments were conducted as follows.

Experiment 1: We gradually increased the travel cost rate for the robot from 0.3/km, 0.6/km,.., 3.0/km in 2E-VRHPD and 2E-VRHPD-NR models.

Experiment 2: We gradually increased the maximum travel distance of the robot from 10km, 15km, 20km, ..., to 55km in the 2E-VRHPD and 2E-VRHPD-NR models.

Experiment 3: We gradually increased the travel cost rate for the robot and the maximum travel distance of the robot simultaneously from (0.3/k, 10km), (0.6/km, 15km),...,(3.0/km,55km) in the 2E-VRHPD and 2E-VRHPD-NR models.

Table 6a-6c report the results of a scenario based sensitivity analysis on different TCR, MTD, and TCR-MTD combinations of robots. NR represents the 2E-VRHPD-NR model as shorthand in the table. e1, e2, and e3 represent the gap between the cost of the 2E-VRHPD model (baseline) and the 2E-VRHPD-NR model in TCR, MTD, and TCR-MTD experiments, respectively.

Table 6: Impact of TCR, MTD, TCR-MTD and no-go zone constraint

TCR			
TCR	2E-VRHPD	NR	e1(%)
0.3	325.78	312.67	4.02
0.6	333.26	317.94	4.60
0.9	340.08	323.25	4.95
1.2	346.20	323.69	6.50
1.5	349.56	324.01	7.31
1.8	357.78	324.62	9.27
2.1	363.55	325.16	10.56
2.4	367.35	325.18	11.48
2.7	372.59	325.25	12.71
3.0	375.09	325.48	13.23

(a) Impact of TCR and no-go zone

MTD			
MTD	2E-VRHPD	NR	e2(%)
10	325.78	312.67	4.02
20	322.88	305.44	5.40
30	319.42	294.86	7.69
40	318.07	288.80	9.20
50	317.87	288.51	9.24
60	317.73	288.61	9.17
70	317.37	288.78	9.01
80	317.38	288.42	9.13
90	317.17	288.41	9.07
100	317.18	288.43	9.06

(b) Impact of MTD and no-go zone

TCR-MTD			
TCR-MTD	2E-VRHPD	NR	e3(%)
0.3,10	325.78	312.67	4.02
0.6,15	329.56	314.98	4.42
0.9,20	336.25	317.15	5.68
1.2,25	339.31	319.39	5.87
1.5,30	346.72	324.23	6.49
1.8,35	356.11	324.30	8.93
2.1,40	365.31	324.36	11.21
2.4,45	367.19	324.47	11.63
2.7,50	380.18	324.50	14.64
3.0,55	380.26	325.00	14.53

(c) Impact of TCR-MTD and no-go zone

For the sensitivity analysis on TCR (table 6a), we have found that the no-go area constraint of the van can reduce the efficiency of the 2E-VRHPD distribution system by 4-14%. With the increase in TCR, e1 tends to increase. For the sensitivity analysis on MTD (table 6b), increasing MTD could increase the system's efficiency, but the increase was limited in the 2E-VRHPD and 2E-VRHPD-NR model. For the sensitivity analysis on TCR-MTD (table 6c), we found that no-go areas of the van would probably reduce the efficiency of the distribution system by 4-15%.

Overall, table 6 shows that the van's no-go restriction had no more than a 15% effect on the system from the above experiments, and the unit mileage of a robot has a strong influence on the system cost. Besides, increasing the driving distance of robots can improve system efficiency, but the overall improvement is small. We therefore conclude that logistics companies do not need to increase the robot's battery capacity excessively in the 2E-VRHPD model. Because the increasing battery capacity has a limited impact on distribution efficiency and will increase the robot's weight and acquisition price.

6.3. Comparison of 2E-VRHPD with classical models

We compared the 2E-VRHPD model with a parallel van and robot scheduling problem with hybrid pickup and delivery operations (PVRSP-HPD) and a model with a two-echelon vehicle routing problem with hybrid pickup and delivery operations (2E-VRP-HPD). We also allowed the van to visit all the customers in the PVRSP-HPD model, used different types of robots, and added the fixed cost of using van, robot, and depot in the 2E-VRP-HPD models to see the influence.

Comparison with PVRSP-HPD

The PVRSP-HPD model, see Figure 8, is similar to the parallel drone scheduling TSP (PDSTSP) introduced by (Murray and Chu 2015), but has hybrid pickup and delivery operations. In the PVRSP-HPD model setting, vans and robots go alone from a single depot to serve customers and then returned to the depot with a TSP route. There is no synchronization between a van and a robot in the PVRSP-HPD. The other constraints of van, robot, depot, and

customers were the same as those in the 2E-VRHPD model: For customers, there are van customers and robot customers, and can be divided into pickup-customers, delivery-customers, and couples of pair-customers. A couple of pair-customers should be served in one van/robot trip. Customers have service time and time window constraints; For the depot, a van/robot, multiple visits to the depot were not allowed, and the depot has time window constraints; For van and robot, the van/robot has its capacity constraints. The robot has energy constraints.

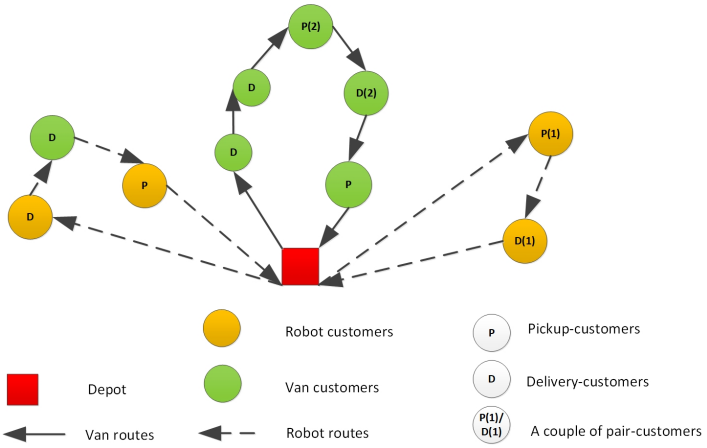

Figure 8: Example of PVRSP-HPD model

Through the pre-analysis of instances we found all ten instances were insolvable in the PVRSP-HPD model, because almost no robot customers or pair customers with robot customers could be successfully served by robots (beyond the maximum travel distance of robots).

We tried to gradually increase the robot's maximum travel distance from 10 km to 100 km (far enough); however, instances were also insolvable because robot delivery violated some customer time windows. It is obviously not feasible to unilaterally increase the robot's speed to a value high enough to fit the customer's time window.

In the next experiments, we relaxed the PVRSP-HPD model's constraints to allow vans to access the no-go areas (PVRSP-HPD-NR model). Hence all instances would become feasible.

However, the distribution system efficiency may be affected due to the high cost per unit van mileage. The experimental results show the average best cost for all the instances was 326.61.

We found that the cost of 2E-VRHPD model was slightly smaller than that for the PVRSP-HPD-NR model (325.78 compared to 326.61). However, if we increased the travel cost rate for the robot in the 2E-VRHPD model to 0.6/km, the cost of 2E-VRHPD model was larger than that for the PVRSP-HPD-NR model (332.26 compared to 326.61). Hence, in our proposed experimental setting the 2E-VRHPD model performs better than the PVRSP-HPD model and to some extent better than the PVRSP-HPD-NR model.

Note that our 2E-VRHPD-NR model is always better than PVRSP-HPD-NR model, even when the travel cost rate for the robot in the 2E-VRHPD-NR is increased to 3.0/km (325.48 compared to 326.61).

Comparison with 2E-VRP-HPD

The 2E-VRP-HPD model, see Figure 9, is similar to the two-echelon vehicle routing problem (2E-VRP) introduced by (Perboli et al. 2011), but has hybrid pickup and delivery operations. In the 2E-VRP-HPD model setting, vans transport goods from a single depot to the satellites. Robots travel from satellites to serve customers and then back to satellites. Then vans transport goods from satellites to the depot. We assumed that there were enough robots in the satellites and satellites can store enough goods. Multiple visits to the satellites by vans and robots are not allowed. The other constraints for vans, robots, depot, and customers are the same as those in the 2E-VRHPD model: For customers, there

are van customers and robot customers, and they can be divided into pickup-customers, delivery-customers, and couples of pair-customers. A couple of pair-customers should be served in one robot trip. Customers have service time and time window constraints; For the depot, a van multiple visits the depot was not allowed, and the depot has time window constraints; For van and robot, the van/robot has its capacity constraints. The robot has energy constraints.

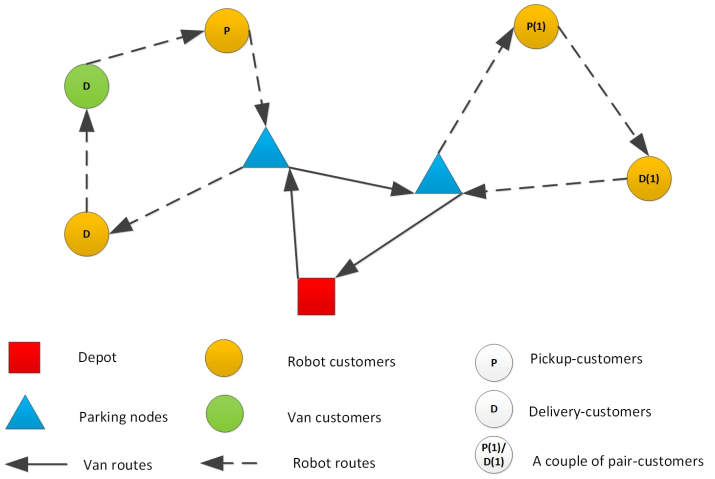

Figure 9: Example of 2E-VRP-HPD model

Through the pre-analysis of instances, we found that there might be instances that are infeasible in the 2E-VRP-HPD model setting. Robots could not complete the delivery task because some customers or pair-customers lay beyond the robot's maximum travel ranges (10 km).

We then provided different types of the robot to see how this would work in the 2E-VRP-HPD model. The types of robots are shown in table 7. Row 1 shows five types of robots. Row 2 represents the maximum travel distances of each kind of robot, and Row 3 gives the travel cost rate of each type of robot.

Table 7: Types of robot

	Robot L1	Robot L2	Robot L3	Robot L4	Robot L5
Maximum travel distances	10km	15km	20km	25km	30km
Travel cost rate	0.3/km	0.6/km	0.9/km	1.2/km	1.5/km

In our computational study, we used the type of robot that would just do the hybrid pickup and delivery task, and then looked at the system's efficiency. In other words, we assumed there were different types of robots in the satellites, and multiple types of robots could be used in an experiment.

Experimental results show the average best cost of the 2E-VRP-HPD model with different types of robots was 277.78, better than the average best cost of the 2E-VRHPD model, which cost was 325.78 (see table 6).

We suspected that the travel cost rate might affect the comparison results of the model. We therefore conducted a sensitivity analysis of the robot's different TCR from 0.3, 0.6,..., 3.0 in the 2E-VRP-HPD model like in Section 6.2.

Different types of robot for TCR experiments are shown in table 8. Row 1 shows five types of robots. Row 2 represents the maximum travel distances of each kind of robot, and Row 3-12 give the travel cost rate of each type of robot in TCR experiments.

Table 9 reports the results of TCR experiments for 2E-VRP-HPD model. Row 1 shows the robot's different TCR. Row 2 shows the average cost for the instances.

We found that when TCR was equal to (or larger than) 2.1, the average best cost of the 2E-VRP-HPD model was higher than that of the 2E-VRHPD model: 373.97 for the 2E-VRP-HPD model compared to 363.55 (see table 6) for the

Table 8: Types of robot for TCR experiments

	Robot L1	Robot L2	Robot L3	Robot L4	Robot L5
Maximum travel distances	10km	15km	20km	25km	30km
TCR experiment	0.3/km	0.6/km	0.9/km	1.2/km	1.5/km
	0.6/km	0.9/km	1.2/km	1.5/km	1.8/km
	0.9/km	1.2/km	1.5/km	1.8/km	2.1/km
	1.2/km	1.5/km	1.8/km	2.1/km	2.4/km
	1.5/km	1.8/km	2.1/km	2.4/km	2.7/km
	1.8/km	2.1/km	2.4/km	2.7/km	3.0/km
	2.1/km	2.4/km	2.7/km	3.0/km	3.3/km
	2.4/km	2.7/km	3.0/km	3.3/km	3.6/km
	2.7/km	3.0/km	3.3/km	3.6/km	3.9/km
	3.0/km	3.3/km	3.6/km	3.9/km	4.2/km

Table 9: TCR experiments for 2E-VRP-HPD model

TCR	0.3	0.6	0.9	1.2	1.5	1.8	2.1	2.4	2.7	3
Average cost	277.78	293.97	310.16	326.41	341.30	357.09	373.97	388.50	404.64	419.02

2E-VRHPD model.

Hence, when the objective function is total energy consumption, the model output is closely related to the robot's travel cost rate.

In the above 2E-VRHPD and 2E-VRP-HPD model comparison, we did not consider the fixed cost of robots and vans, or satellites. We therefore added the fixed cost of using van, robot, and satellite into the objective of the two models to see how the fixed cost influence the model-output.

We set the cost of using one robot once from 1, 2,...,10. Then we set the cost of using one van once is twice that of using a robot and set the cost of using one satellite once is three times that of using a robot. Table 10 reports the effect of fixed cost changes on the two models' output. Row 1 represents the fixed cost combinations of (robot, van, satellite). Row 2 is the output of the 2E-VRP-HPD model. Row 3 is the output of the 2E-VRHPD model.

Table 10: The effect of fixed cost changes on the output of the 2E-VRP-HPD and 2E-VRHPD models

Fixcost	(0,0,0)	(1,2,3)	(2,4,6)	(3,6,9)	(4,8,12)	(5,10,15)	(6,12,18)	(7,14,21)	(8,16,24)	(9,18,27)	(10,20,30)
2E-VRP-HPD	277.88	295.91	311.29	328.53	342.99	359.06	373.99	389.48	405.00	420.49	435.43
2E-VRHPD	325.78	328.10	334.04	344.60	348.75	352.89	361.41	364.91	371.17	380.08	385.59

Table 10 shows that when the fixed cost of (robot,van,satellite) is smaller than (5,10,15), the output of the 2E-VRP-HPD model was better than that of the 2E-VRP-HPD model. Besides, when the fixed cost of (robot,van,satellite) is larger than (4,8,12), the 2E-VRHPD model is more competitive. The price of corporate logistics infrastructure will also strongly affect the output of 2E-VRHPD and 2E-VRP-HPD models.

Comparing the results of the PVRSP-HPD model and 2E-VRP-HPD model, we advocate logistic companies using the 2E-VRHPD model will most probably improve the distribution system's efficiency. Besides, the 2E-VRP-HPD model has advantages in some cases, especially when the fixed cost and the robot travel cost rate are low. We therefore suggest that logistic companies might consider hybrid distribution systems automatically selecting the optimal distribution mode according to the situation.

7. Conclusion and future research directions

Recent demonstrations by JD.com and Starship Technologies (among others) have shown the potential of robots for small parcel pickup and delivery services. In addition, the service demand of logistics companies has begun to diversify,

requiring the combination of multiple modes of pickup and delivery operations in one trip. Current research also shows that combining pickup and delivery operations in one trip can improve the distribution system's overall efficiency.

In this paper, a van-based robot logistic distribution system that combines hybrid pickup and delivery operations was studied. Larger vans carry small robots along the van route. The robots travel along with the robot open route. The van and robot can serve customers directly, but some constrained customers can be visited only by robots. The van stops at parking nodes to drop off and/or pick up its robot, replenish its robot, and swap its robot's battery if needed. Five detailed pickup and delivery cases in the 2E-VRHPD model were introduced. A van and/or robot can load goods from a depot and deliver them to a customer, or a van and/or robot can pick up goods from a customer and take them to another customer, or a van and/or robot can pick up goods from a customer and take them to a depot.

The associated optimization problem was formulated as a two-echelon heterogeneous vehicle routing problem with hybrid pickup and delivery, time windows, and vehicle collaborations. An MIP formulation including arc, time, freight, and energy parts, along with a simple ALNS-based heuristic approach solving large instances that have been introduced. The capacity feasibility test approaches are proposed for a given 2E-VRHPD route. An extensive computational study to evaluate the performance of the proposed ALNS approaches is presented. How the parking node density influences the output solution was studied. We constructed a case study based on realistic scenarios. We conducted a sensitivity analysis on the robot's travel cost rate and maximum travel distances and compared the van no-go area's effect on the 2E-VRHPD model. The results show that no-go restrictions for a van may significantly impact 2E-VRHPD system efficiency, and that the maximum travel distance of robots has a limited impact on 2E-VRHPD system efficiency if the robot's maximum travel distances exceed 10 km. We also compared the 2E-VRHPD model with two classical models. Results show the 2E-VRHPD model is competitive in appropriate scenarios. We therefore advocate considering the 2E-VRHPD model for distribution in appropriate scenarios to improve the system's efficiency.

Future research could aim to design a fast, exact algorithm for the 2E-VRHPD model and consider further aspects such as waiting time penalties. Incorporating the PHVRPD model into the 2E-VRHPD model is also an exciting research direction because it could improve the efficiency of distribution companies.

Acknowledgements The research is supported by the National Natural Science Foundation of China [grant no. 51975482] and Shaanxi Provincial Key R&D Program of China [grant no. 2019ZDLGY14-10]. This work is supported by the China Scholarship Council and by public funding within the scope of the French Program "Investissements d'Avenir".

References

Accorsi, L. and Vigo, D. (2020). A hybrid metaheuristic for single truck and trailer routing problems. *Transportation Science*, 54(5):1351–1371.

Agatz, N., Bouman, P., and Schmidt, M. (2018). Optimization approaches for the traveling salesman problem with drone. *Transportation Science*, 52(4):965–981.

Anderluh, A., Nolz, P. C., Hemmelmayr, V. C., and Crainic, T. G. (2021). Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and 'grey zone'customers arising in urban logistics. *European Journal of Operational Research*, 289(3):940–958.

Andrew, J. H. (2019). Thousands of autonomous delivery robots are about to descend on us college campuses. `https://www.theverge.com/2019/8/20/20812184/starship-delivery-robot-expansion-college-campus`. Accessed August 20, 2019.

Avci, M. and Topaloglu, S. (2016). A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*, 53(Jul.):160–171.

Bakach, I., Campbell, A. M., and Ehmke, J. F. (2021). A two-tier urban delivery network with robot-based deliveries. *Networks*, pages 1–23.

Battarra, M., Cordeau, J.-F., and Iori, M. (2014). Pickup and delivery problems for goods transportation. *Vehicle Routing: Problems, Methods, and Applications*, pages 161–191.

Bergmann, F. M., Wagner, S. M., and Winkenbach, M. (2020). Integrating first-mile pickup and last-mile delivery on shared vehicle routes for efficient urban e-commerce distribution. *Transportation Research Part B: Methodological*, 131:26–62.

Bouman, P., Agatz, N., and Schmidt, M. (2018). Dynamic programming approaches for the traveling salesman problem with drone. *Networks*, 72(4):528–542.

Boysen, N., Schwerdfeger, S., and Weidinger, F. (2018). Scheduling last-mile deliveries with truck-based autonomous robots. *European Journal of Operational Research*, 271(3):1085–1099.

Breunig, U., Baldacci, R., Hartl, R. F., and Vidal, T. (2019). The electric two-echelon vehicle routing problem. *Computers & Operations Research*, 103:198–210.

Carlsson, J. G. and Song, S. (2018). Coordinated logistics with a truck and a drone. *Management Science*, 64(9):4052–4069.

Chen, C., Demir, E., and Huang, Y. (2021). An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots. *European Journal of Operational Research*, doi.org/10.1016/j.ejor.2021.02.027.

Darvish, M., Archetti, C., Coelho, L. C., and Speranza, M. G. (2019). Flexible two-echelon location routing problem. *European Journal of Operational Research*, 277(3):1124–1136.

Dayarian, I., Savelsbergh, M. W. P., and Clarke, J.-P. (2020). Same-day delivery with drone resupply. *Transportation Science*, 54:229–249.

Dellaert, N., Dashty Saridarq, F., Van Woensel, T., and Crainic, T. G. (2019). Branch-and-price-based algorithms for the two-echelon vehicle routing problem with time windows. *Transportation Science*, 53(2):463–479.

Dongmei, L. (2017). Jd.com launches robot delivery services in chinese universities. `https://www.chinamoneynetwork.com/2017/06/19/jd-com-launches-robot-delivery-services-in-chinese-universities`. Accessed June 19, 2017.

Drexl, M. (2012). Synchronization in vehicle routing—a survey of vrps with multiple synchronization constraints. *Transportation Science*, 46(3):297–316.

Drexl, M. (2020). On the one-to-one pickup-and-delivery problem with time windows and trailers. *Central European Journal of Operations Research*, pages 1–48.

Dumas, Y., Desrosiers, J., and Soumis, F. (1991). The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7–22.

Gonzalez-R, P. L., Canca, D., Andrade-Pineda, J. L., Calle, M., and Leon-Blanco, J. M. (2020). Truck-drone team logistics: A heuristic approach to multi-drop route planning. *Transportation Research Part C-emerging Technologies*, 114:657–680.

Jie, W., Yang, J., Zhang, M., and Huang, Y. (2019). The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology. *European Journal of Operational Research*, 272(3):879–904.

Karak, A. and Abdelghany, K. (2019). The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transportation Research Part C: Emerging Technologies*, 102:427–449.

Kitjacharoenchai, P., Min, B.-C., and Lee, S. (2020). Two echelon vehicle routing problem with drones in last mile delivery. *International Journal of Production Economics*, 225:107598.

Koç, Ç., Laporte, G., and Tükenmez, İ. (2020). A review on vehicle routing with simultaneous pickup and delivery. *Computers & Operations Research*, page 104987.

Li, H., Chen, J., Wang, F., and Bai, M. (2021). Ground-vehicle and unmanned-aerial-vehicle routing problems from two-echelon scheme perspective: a review. *European Journal of Operational Research*, doi.org/10.1016/j.ejor.2021.02.022.

Li, H., Wang, H., Chen, J., and Bai, M. (2020). Two-echelon vehicle routing problem with satellite bi-synchronization. *European Journal of Operational Research*, 288(3):775–793.

Liping, G. (2018). Delivery robots hit the road in beijing. `http://www.ecns.cn/news/sci-tech/2018-06-20/detail-ifyvmiee7350792.shtml`. Accessed June 20, 2018.

Liu, T., Luo, Z., Qin, H., and Lim, A. (2018). A branch-and-cut algorithm for the two-echelon capacitated vehicle routing problem with grouping constraints. *European Journal of Operational Research*, 266(2):487–497.

Luo, Z., Liu, Z., and Shi, J. (2017). A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle. *Sensors*, 17(5):1144.

Mirhedayatian, S. M., Crainic, T. G., Guajardo, M., and Wallace, S. W. (2021). A two-echelon location-routing problem with synchronisation. *Journal of the Operational Research Society*, 72(1):145–160.

Moshref-Javadi, M., Hemmati, A., and Winkenbach, M. (2020a). A truck and drones model for last-mile delivery: A mathematical model and heuristic approach. *Applied Mathematical Modelling*, 80:290–318.

Moshref-Javadi, M., Lee, S., and Winkenbach, M. (2020b). Design and evaluation of a multi-trip delivery model with truck and drones. *Transportation Research Part E-logistics and Transportation Review*, 136:101887.

Mourad, A., Puchinger, J., and van Woensel, T. (2020). Integrating autonomous delivery service into a passenger transportation system. *International Journal of Production Research*, pages 1–24.

Mühlbauer, F. and Fontaine, P. (2021). A parallelised large neighbourhood search heuristic for the asymmetric two-echelon vehicle routing problem with swap containers for cargo-bicycles. *European Journal of Operational Research*, 289(2):742–757.

Murray, C. C. and Chu, A. G. (2015). The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54:86–109.

Murray, C. C. and Raj, R. (2020). The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transportation Research Part C: Emerging Technologies*, 110:368–398.

Otto, A., Agatz, N., Campbell, J., Golden, B., and Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks*, 72(4):411–458.

Parragh, S. N. and Cordeau, J.-F. (2017). Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows. *Computers & Operations Research*, 83:28–44.

Perboli, G., Brotcorne, L., Bruni, M. E., and Rosano, M. (2021). A new model for last-mile delivery and satellite depots management: The impact of the on-demand economy. *Transportation Research Part E: Logistics and Transportation Review*, 145:102184.

Perboli, G. and Rosano, M. (2019). Parcel delivery in urban areas: Opportunities and threats for the mix of traditional and green business models. *Transportation Research Part C: Emerging Technologies*, 99:19–36.

Perboli, G., Tadei, R., and Vigo, D. (2011). The two-echelon capacitated vehicle routing problem: Models and math-based heuristics. *Transportation Science*, 45(3):364–380.

Pisinger, D. and Ropke, S. (2019). Large neighborhood search. In *Handbook of Metaheuristics*, pages 99–127. Springer.

Poikonen, S. and Golden, B. (2020a). The mothership and drone routing problem. *INFORMS Journal on Computing*, 32(2):249–262.

Poikonen, S. and Golden, B. (2020b). Multi-visit drone routing problem. *Computers & Operations Research*, 113:104802.

Poikonen, S., Golden, B., and Wasil, E. A. (2019). A branch-and-bound approach to the traveling salesman problem with a drone. *INFORMS Journal on Computing*, 31(2):335–346.

Poikonen, S., Wang, X., and Golden, B. (2017). The vehicle routing problem with drones: Extended models and connections. *Networks*, 70(1):34–43.

Pugliese, L. D. P. and Guerriero, F. (2017). Last-mile deliveries by using drones and classical vehicles. In *International Conference on Optimization and Decision Science*, pages 557–565. Springer.

Qu, Y. and Bard, J. F. (2013). The heterogeneous pickup and delivery problem with configurable vehicle capacity. *Transportation Research Part C-emerging Technologies*, 32:1–20.

Roberti, R. and Ruthmair, M. (2021). Exact methods for the traveling salesman problem with drone. *Transportation Science*, 55(2):315–335.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):p.455–472.

Rothenbächer, A.-K., Drexl, M., and Irnich, S. (2018). Branch-and-price-and-cut for the truck-and-trailer routing problem with time windows. *Transportation Science*, 52(5):1174–1190.

Sacramento, D., Pisinger, D., and Ropke, S. (2019). An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C: Emerging Technologies*, 102:289–315.

Salama, M. and Srinivas, S. (2020). Joint optimization of customer location clustering and drone-based routing for last-mile deliveries. *Transportation Research Part C-emerging Technologies*, 114:620–642.

Savelsbergh, M. W. P. and Sol, M. (1995). The general pickup and delivery problem. *Transportation Science*, 29(1):17–29.

Schermer, D., Moeini, M., and Wendt, O. (2019a). A hybrid vns/tabu search algorithm for solving the vehicle routing problem with drones and en route operations. *Computers & Operations Research*, 109:134–158.

Schermer, D., Moeini, M., and Wendt, O. (2019b). A matheuristic for the vehicle routing problem with drones and its variants. *Transportation Research Part C: Emerging Technologies*, 106:166–204.

Sun, W., Yu, Y., and Wang, J. (2019). Heterogeneous vehicle pickup and delivery problems: Formulation and exact solution. *Transportation Research Part E: Logs and Transportation Review*, 125:181–202.

Wang, X., Poikonen, S., and Golden, B. (2017). The vehicle routing problem with drones: several worst-case results. *Optimization Letters*, 11(4):679–697.

Wang, Y., Assogba, K., Liu, Y., Ma, X., Xu, M., and Wang, Y. (2018). Two-echelon location-routing optimization with time windows based on customer clustering. *Expert Systems with Applications*, 104:244–260.

Wang, Z. and Sheu, J.-B. (2019). Vehicle routing problem with drones. *Transportation research part B: methodological*, 122:350–364.

Yu, S., Puchinger, J., and Sun, S. (2020). Two-echelon urban deliveries using autonomous vehicles. *Transportation Research Part E: Logistics and Transportation Review*, 124:102018.

Appendix A. Route cases of permission and prohibition

In order to well illustrate which route type is allowed and which route type is not allowed, we display the van-robot route cases in Figure A.10.

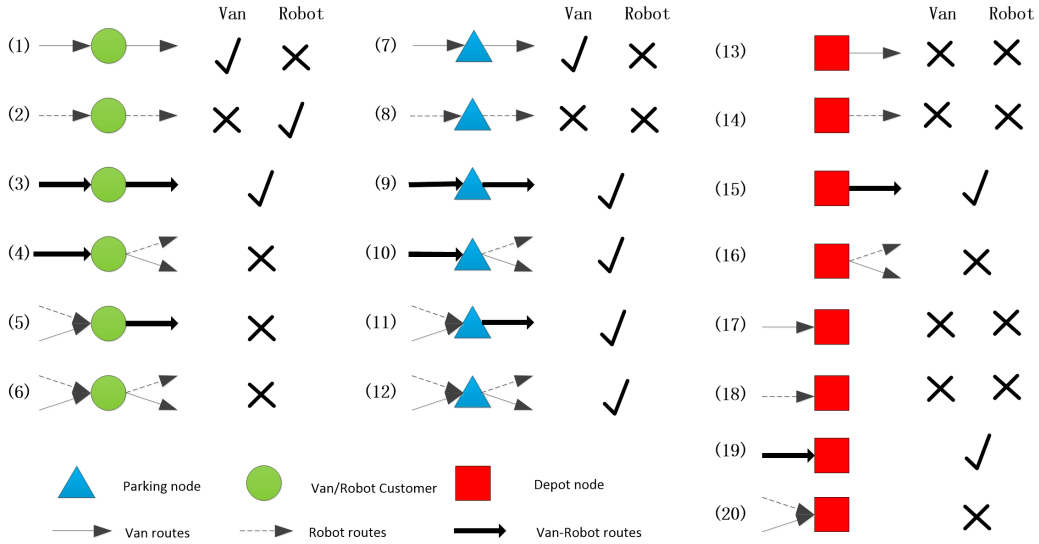

Figure A.10: 2E-VRHPD route cases of permission and prohibition

Appendix B. ALNS algorithm

After the initial solution has been constructed, the proposed ALNS algorithm seeks to improve it iteratively until a stopping condition is satisfied. We adopt a maximum iteration number (MaxIteNum) limit to terminate the algorithm. At each iteration, the existing feasible solution is destroyed by removing some nodes or routes by a destroy algorithm. The resulting partial solution is then repaired using a repair algorithm which heuristically repairs the existing route or constructs a new route if needed, for the purpose of obtaining a better solution than the previous one. We also set a number of non-improving iterations (MaxNonImp) before restarting the ALNS algorithm from a new initial solution as in Breunig et al. (2019).

During repair operations, the route evaluation method is used to check the feasibility of the routes. We use a simulated annealing-like acceptance criterion, which allows the algorithm to accept poor solutions during iterations. In addition, each of our destroy-and-repair operations is assigned a weight that controls how often the operation is selected. The weights are adjusted dynamically as the search progresses by a weight-adjusting method. The algorithm thus adapts to the instance at hand and to the state of the search. We draw on simulated annealing and weight-adjusting proposed by Pisinger and Ropke (2019).

The details of simulated annealing and weight-adjusting used here as follows.

Simulated annealing

The simulated annealing uses a temperature parameter T to control the acceptance probability. The temporary solution x^t is always accepted if x^t has a better objective value than the current best solution x^b ($c(x^t) <= c(x^b)$), and it is accepted with probability $\exp(c(x^b) - c(x^t)/T)$ if this is not the case. Here $T > 0$ is the current temperature. The start temperature is initialized at $T_{st} > 0$ and is decreased gradually by performing an update $T = \alpha * T$ at each iteration, where $\alpha \in [0, 1]$ is the cooling rate of simulated annealing.

Weight-adjusted

For the destroy and repair operations, a score $\psi = max(w_1, w_2, w_3, w_4)$ is used to adjust their respective weights and is computed according the following formula:

$$\psi = \begin{cases} w_1 & \text{if the new solution is a new global best solution,} \\ w_2 & \text{if the new solution is better than the current solution,} \\ w_3 & \text{if the new solution is accepted,} \\ w_4 & \text{if the new solution is rejected.} \end{cases}$$

Normally we have $w_1 \geq w_2 \geq w_3 \geq w_4 \geq 0$.

The components corresponding to the selected destroy-and-repair operations in the ρ^- and ρ^+ vectors are updated using the following equations: $\rho_a^- = \lambda\rho_a^- + (1-\lambda)\psi$, $\rho_b^+ = \lambda\rho_b^+ + (1-\lambda)\psi$, in which a and b are the indices of the destroy-and-repair methods that were used in the last iteration of the algorithm, respectively. The components corresponding to the selected destroy, and repair methods in the ρ^- and ρ^+ vectors are updated using the equations. $\lambda \in [0, 1]$ is the decay parameter that controls how sensitive the weightings are to changes in the performance of the destroy-and-repair methods.

Our proposed ALNS algorithm is described in Algorithm 1. The best solution is recorded in *recordset* (Line 1). The algorithm starts with an initial solution x and it is also the current best solution x^b (Line 2). The value of $NumNunImp$ is then initialized to 0 (Line 3), where $NumNunImp$ records the number of iterations until $NumNunImp$ becomes larger than $MaxNunImp$. Next, a destroy-and-repair operation is performed until the stopping criterion ($MaxIteNum$) is met (Line 4-19). The $restart()$ function is used to restart the ALNS from Line 2 but keeps the iteration number unchanged.

Appendix C. Pseudocode for the construction of the initial solution

Algorithm 2 is the pseudocode for the initial solution generation algorithm. The input data are the un-assigned customer list S_1 and the 2E-VRHPD route set S_2 (Line 1). We first construct a basic route contains a pickup-customer or a delivery-customer, or a couple of pair-customers (Line 3-22). We then insert nodes to the basic route (Line 23-52). We repeat the above two steps until the $S_1 = \varnothing$ (Line 2-58).

The procedures used in Algorithm 2 are introduced as follows.

The $RandomFeasibleInsert1(node, route)$ procedure randomly find a position in the route to insert node until the node is inserted or all positions have been tried: If insertion is feasible, the procedure ends. Otherwise, randomly find a position (except the positions that have been tried before) to insert the node.

The $RandomFeasibleInsert2(node, route)$ procedure is similar to the RandomFeasibleInsert1(node,route) procedure, except that all the test positions are after the pickup-pair-customer node.

The $FindCorrespondingNode(node)$ procedure gets corresponding pickup/delivery-pair-customer. If the input node

Algorithm 1 *Adapt_Large_Neighborhood_Search*()

```
recordset = {}
Initialization: use RouteReconstruction operator to get an initial solution x, let x^b = x.
NumNunImp = 0
while stopping criterion is not met do
    select destroy and repair methods from set.
    x^t = r(d(x))
    if simulated annealing criterion accepts the solution then
        x = x^t
    end if
    if c(x) < c(x^b) then
        x^b = x, NumNunImp = 0
    else
        NumNunImp+ = 1
        if restart conditions is met then
            recordset.append(x^b) and then restart()
        end if
    end if
    update weights and selection parameters
end while
return: output best x^b from recordset
```

is pickup-pair-customer, FindCorrespondingNode(node) output the delivery-pair-customer, if the input node is delivery-pair-customer, FindCorrespondingNode(node) output the pickup-pair-customer.

The $GenerateNewRobotRoute(node, route)$ procedure builds a new robot route from the constructed route. The procedure works as follows. Rely on the parking node used for recycling robot (which serves the pickup-pair-customer) or the parking node after the pickup-parking node to establish a new robot route and insert the corresponding delivery-pair-customer.

The $ConstructRoute(node)$ procedure is used to construct a route. If the node is a robot customer, constructs a route with van routes and robot routes. Otherwise, construct a route with van routes and robot routes or construct a route with only van routes, according to a roulette-wheel mechanism.

Appendix D. parallel routes capacity testing algorithm

Algorithm 3 is the algorithm to check the capacity of the parallel routes. The input is the van and robot's total capacity (FreightAmount) at the parking node, the parallel routes (RouteRoute and IndependentVanRoute), and the BasicData (Line 1). The BasicData includes the van customer nodes, the demand of customer node ($Load[node]$), the maximum capacity of van (C_1) and robot (C_2), and a large value ($LargeValue$). We note that $Dict[node]$ represents the corresponding pair-customer of the node. We first calculate how many goods a robot must carry before leaving the parking nodes (Line 2 - Line 13). We then test the feasibility of the RobotRoute and calculate the 'gap' between the number of goods the robot must take and the number of goods the robot can take at most. (Line 14 - Line 29). We further calculate how many goods a van must carry before leaving the parking nodes (Line 30 - Line 41). Next, we calculate how many FreightAmount the van must carry according to the gap (Line 42 - Line 46), and we test whether the independent-van route's capacity is feasible (Line 47 - Line 54). Finally, we test whether the van and robot's total capacity in the parking node for picking up exceeds the van's capacity (Line 55 - Line 59).

Algorithm 2 *Initial_Solution_Generation_Algorithm*()

```
1: Initialization: S_1, S_2.
2: while S_1 ≠ ∅ do
3:    node ← RandomChooseANode(S_1).
4:    if node is pair-customer then
5:       if node is pickup-pair-customer then
6:          node1 ← node
7:          node2 ← FindCorrespondingNode(node1)
8:       else
9:          node2 ← node
10:         node1 ← FindCorrespondingNode(node2)
11:      end if
12:      route1 ← ConstructRoute(node1)
13:      if !(RandomFeasibleInsert2(node2,route1)) then
14:         route2 ←RandomFeasibleInsert2(node2,route1).
15:      else
16:         route2 ← GenerateNewRobotRoute(node2,route1)
17:      end if
18:      S_1.remove(node1,node2)
19:   else
20:      route2 ← ConstructRoute(node)
21:      S_1.remove(node)
22:   end if
23:   while S_1 ≠ ∅ do
24:      node ← RandomChooseANode(S_1).
25:      if node is pair-customer then
26:         if node is pickup-pair-customer then
27:            node1 ← node
28:            node2 ← FindCorrespondingNode(node1)
29:         else
30:            node2 ← node
31:            node1 ← FindCorrespondingNode(node2)
32:         end if
33:         if !(Random-FeasibleInsert1(node1,route2)) then
34:            route1 ←RandomFeasibleInsert1(node1,route2)
35:         else
36:            break
37:         end if
38:         if !(Random-FeasibleInsert2(node2,route1)) then
39:            route2 ←RandomFeasibleInsert2(node2,route1)
40:            S_1.remove(node1,node2)
41:         else
42:            break
43:         end if
44:      else
45:         if !(RandomFeasibleInsert1(node,route2)) then
46:            route2 ←RandomFeasibleInsert1(node,route2)
47:            S_1.remove(node)
48:         else
49:            break
50:         end if
51:      end if
52:   end while
53:   S_2.add(route2)
54: end while
55: return: output S_2
56: Note: S_1: un-assigned customer list; S_2: 2E-VRHPD route set.
```

Algorithm 3 *Test_Parallel_Route_Capacity*()

Input: FreightAmount, RobotRoute, IndenpedentVanRoute, BasicData
for node in RobotRoute **do**
  RobotFreight = 0
  if node is DeliveryCustomerNode **then**
    RobotFreight += $Load[node]$
  end if
  if node is DeliveryPairCustomer and $Dict[node]$ is not in RobotRoute **then**
    RobotFreight += $Load[node]$
  end if
end for
if RobotFreight > C_2 **then**
  return False
end if
FreightAmount -= RobotFreight
TempGap = $LargeValue$
for node in RobotRoute **do**
  if node is CustomerNode **then**
    RobotFreight -= $Load[node]$
    if RobotFreight > C_2 **then**
      return False
    else
      Gap = C_2 - RobotFreight
      if Gap < TempGap **then**
        TempGap = Gap
      end if
    end if
  end if
end for
Gap = TempGap
for node in IndependentVanRoute **do**
  VanFreight = 0
  if node is DeliveryCustomerNode **then**
    VanFreight += $Load[node]$
  end if
  if node is DeliveryPairCustomer and $Dict[node]$ is not in IndependentVanRoute **then**
    VanFreight += $Load[node]$
  end if
end for
if FreightAmount < VanFreight **then**
  return False
end if
if FreightAmount - Gap < VanFreight **then**
  FreightAmount = VanFreight
else
  FreightAmount -= Gap
end if
for node in IndependentVanRoute **do**
  if node is CustomerNode **then**
    FreightAmount = FreightAmount - $Load[node]$
    if FreightAmount > C_1 **then**
      return False
    end if
  end if
end for
FreightAmount = FreightAmount + RobotFreight + Gap
if FreightAmount > C_1 **then**
  return False
end if
return True, FreightAmount

Appendix E. ALNS parameters

We drew on the parameter tuning concept (Jie et al. 2019) to determine the parameters. The parameter tuning approach started with a set of initial parameter values (table E.11). Parameter tuning conducted the ALNS algorithm with the parameter values in the search interval. The parameter value that had the best solution was chosen to perform the following calculation. The process was repeated until all the parameters in the parameter set had been tuned. We chose eight instances with 3-15 satellite-customers to conduct the parameter tuning procedure: CA1-3-15, CA2-3-15, CB1-3-15, CB2-3-15, CC1-3-15, CC2-3-15, CD1-3-15, and CD2-3-15. The parameter notations, descriptions, and values used here can be found in E.11.

Table E.12 lists the parameter, initial value, search interval, and parameter value we used in ALNS algorithm.

Table E.11: Parameters values

Parameter	Notation and description	Value
MaxIteNum	Maximum iteration number of ALNS	20000
MaxNunImp	Maximum iteration number with no improved solution	1800
β	Destruction rate	0.15
λ	Decay parameter	0.95
T_{st}	Start temperature per customer of SA	1000
α	Cooling rate of SA	0.92
w_1	new solution is a new global best solution	3
w_2	new solution is better than the current one	2
w_3	new solution is accepted	1
w_4	new solution is rejected	0

Table E.12: Parameters values

Parameter	Initial value	Search interval	Final value
MaxIteNum	20000	5000-50000 (5000)	10000
MaxNunImp	1800	100-2000 (100)	400
β	0.15	0.05-0.5 (0.05)	0.40
λ	0.95	0.90-0.99 (0.01)	0.90
T_{st}	1000	1000-10000 (1000)	1000
α	0.92	0.90-0.99 (0.01)	0.98
w_1	3	3-30	22
w_2	2	2-21	17
w_3	1	1-16	1
w_4	0	0	0