An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots

Cheng Chen[a*], Emrah Demir[b,c], Yuan Huang[c]

[a] School of Transportation and Civil engineering, Fujian Agriculture and Forestry University, Fuzhou, China

[b] PARC Institute of Manufacturing, Logistics and Inventory, Cardiff Business School, Cardiff University, Cardiff, UK

[c] Logistics and Operations Management, Cardiff Business School, Cardiff University, Cardiff, UK

fjnlcc@fafu.edu.cn (* Corresponding Author, C. Chen), Demire@cardiff.ac.uk (E. Demir), huangy66@cardiff.ac.uk (Y. Huang)

Abstract

Considering autonomous delivery robots in urban logistics has attracted a great deal of attention in recent years. In the meantime, new technology has led to new operational challenges, such as the routing and scheduling of vehicles and delivery robots together that are currently outside the logistics service providers' capability. In this paper, we present a vehicle routing problem with time windows and delivery robots (VRPTWDR) as a variant of the classical VRP. The investigated problem is concerned with the routing of a set of delivery vans equipped with a number of self-driving parcel delivery robots. To tackle the VRPTWDR, an Adaptive Large Neighborhood Search heuristic algorithm is developed. Experiments show the performance and effectiveness of the algorithm for solving the VRPTWDR and provide insights on the use of self-driving parcel delivery robots as an alternative last mile service.

Keywords: Vehicle routing problem; Transportation; Delivery robot; Metaheuristic algorithm

1. Introduction

Parcel delivery service in last mile logistics has been subjected to a great deal of pressure caused by population growth and urbanization over the last decade. The development of e-commerce and the faster-growing lifestyle led to a dramatic change in last mile logistics for on-time parcel deliveries since the early 2000s. At the same time, the arrival of new technologies and environmental concerns have started shaping the logistics industry by increasing the competition amongst the logistics service providers (LSPs) to cope with the increasing number of requests for greener, faster, safer and cheaper deliveries.

The adaptions of new technologies such as artificial intelligence, digitalization and autonomous systems can achieve contactless delivery and may help improve the efficiency of logistics systems. However, there is a need for systematic assessment of the evidence before fully committing to these technologies. For example, Pelletier et al. (2016) pointed out the challenges and issues that need to be overcome for electric vehicles. This is also a valid point for new technologies, such as self-driving parcel delivery robots, drones and automated guided vehicles with compartments.

As alternative last mile service solutions, parcel delivery robots use sidewalk of a road whereas drones are

able to make deliveries on different heights. Both can achieve door-to-door delivery and offer on-time deliveries for customers even in a highly congested urban network while contributing to congestion alleviation. With distinct advantages including mobility, flexibility, low cost and energy-consumption, tentative applications of drones and parcel delivery robots have already been implemented in practice (see, e.g., BBC News, 2019; Diaz, 2019; Francis, 2019; Vincent & Gartenberg, 2019). What's more, in emergencies with high risks, e.g., deliveries in virus epidemic areas, those autonomous delivery systems play a more crucial role in protecting people from contact and ensuring the social distance for drivers.

Poikonen et al. (2017) investigated the practical advantages offered through using drones as delivery assistants and concluded that the highest benefit from using drones depends on the number of drones used and their flying speeds. Although drones travel faster than delivery robots, LSPs can benefit from simultaneous/parallel deliveries as a delivery van can be equipped with multiple robots (e.g., Mercedes-Benz futuristic electronic van can be loaded with up to 8 delivery robots (McFarland, 2016)).

On the other hand, the biggest issues related to drone deliveries are about the safety and security. Besides instructions and regulations laid by governments to restrict the use of drones for commercial activities, severe weather conditions may weaken the safety of the drone delivery or make it even unavailable (Kottasova, 2016). By contrast, as parcel delivery robots travel at a pedestrian speed and can work in challenging environments (Hutter et al., 2017), delivery robots can offer dintinctive advantages over drones.

Consequently, several commercial organizations, such as FedEx, have already started their trials on combined vehicle and robot services. The van gets close to the delivery address with a few hundred meters and the driver manually monitors and controls the robot to complete the delivery to the customer's door. In this circumstance, robots do not stay far from the driver, which strengthens the safety of parcel and the robot itself.

In operations research literature, last mile parcel deliveries are generally modeled and studied as Vehicle Routing Problem (VRP) (Demir et al., 2019). In this problem, multiple delivery vans are located at a central depot and vehicle routes are constructed to serve all customer nodes by minimizing the sum of routes' completion times or operational costs. Furthermore, the VRP involving customers who are only available during specific time periods is named as VRP with time windows (VRPTW). After the emergence of delivery assistants (e.g., delivery robots), each van can be equipped with several delivery robots while operating last mile parcel delivery tasks in populated areas. Using swarm dispatching approach, several self-driving robots take care of deliveries simultaneously while the standard delivery by a driver is being carrying out. Investigating the addition of delivery robots as a delivery assistant introduces us the VRPTW and delivery robots (VRPTWDR) as the focus of this research.

The contributions of this research to the literature are as follows. We first studied the VRPTW where each delivery van (vehicle) is collaborating with several autonomous delivery robots. It is assumed that these self-driving delivery robots are dispatched and collected at the same location after the vehicle parks at a customer site. While they are serving their assigned customers, the driver of the vehicle visits the customer where the vehicle parks in the way of traditional delivery. Second, we systematically analyzed the synchronization issues caused by time windows constraints and the relation of two different delivery resources on time issues under the

dispatch-wait-collect system. This analysis is essential for checking the feasibility of a solution in search procedures of heuristics. Third, as it is more complicated than VRPTW which is a NP-hard problem, an advanced Adaptive Large Neighborhood Search (ALNS) heuristic algorithm is developed for the investigated VRPTWDR. The proposed algorithm can be used as an effective choice for the routing of delivery vans and robots in collaboration. Besides the standard ALNS framework (Ropke & Pisinger, 2006; Pisinger & Ropke, 2007), the suggested operators are deliberately tailored for the VRPTWDR. Results highlight the performance of the presented algorithm and significant savings in terms of operational times can be achieved. And finally, the characteristics of adopting this coordinated delivery system are explored through conducting sensitivity analyses and several managerial insights are derived.

The rest of this paper is organized as follows. Section 2 presents a literature review of the related routing problems. In Section 3, we formally and mathematically present the VRPTWDR. The ALNS metaheuristic algorithm is provided in detail in Section 4. The following section, Section 5, details the computational experiments carried out in this paper and draws managerial insights. Next to analyses on the effectiveness and efficiency of the algorithm, sensitivity analyses concerning the modification of characteristics of delivery robots and the application circumstances are conducted. Section 6 presents both conclusions and the future research directions emerging from this study.

2. Literature review

To improve the efficiency of last mile services and address the challenges arising in the constantly changing delivery environment, different means of services have been introduced into traditional single-type vehicle delivery systems. Besides routing problems exclusive for new delivery resources, such as Drone Delivery Problem (Dorling et al., 2017) and Drone Arc Routing Problems (Campbell et al., 2018), there are various ways in which those multiple resources can cooperate with each other. For example, the Parallel Drone Scheduling Travelling Salesman Problem routes a vehicle and a drone separately (Ham, 2018), while the Flying Sidekick TSP (FSTSP) or TSP with drones (TSP-D) integrates them into the same route (Murray & Chu, 2015; Agatz et al., 2018; Es Yurek & Ozmutlu, 2018). Given the complexity of the integrated routing problem with two resources, heuristic or metaheuristic algorithms are used in most related studies.

One of the closest problems is the Truck and Trailer Problem (TTRP) in which a cooperation of a truck and a trailer are explicitly considered. A subset of the customer needs to be served by a truck, and the rest should be visited by a truck or a truck with a trailer. Lin et al. (2011) introduced the TTRP with time windows and proposed a Simulated Annealing algorithm. However, in this variant, the trailer cannot move unless it is connected to a truck. This is different than the self-driving delivery robot case.

Vehicle(s) plus courier(s) can also be considered as a dual-mode delivery system. Nguyêñ et al. (2018) investigated the routing problem arising in this kind of delivery system that considers both a courier's walking and a van's driving in a single route. With predefined clusters, the study investigated a variant of clustered TSP with time windows (CTSPTW). A new mathematical model involving the routing of vehicles, parking locations, and drivers' walking sequences as decisions is developed. The model was solved with a mathematical solver and

the results revealed the impact of time windows on the feasibility of routes. Alvarez & Munari (2017) studied the VRPTW and multiple deliverymen (VRPTWMD) in which customer service times are dependent of the number of deliverymen allocated to a route as the customers are grouped. The authors developed a branch-price-and-cut algorithm along with two different metaheuristic algorithms. The VRPTWMD shares many similarities with the VRPTWDR as they both try to increase the delivery efficiency through parallel deliveries. However, in the VRPTWMD, customer clusters are defined in advance while the VRPTWDR focuses on the clustering of customers.

In other multi-mode delivery systems, the resources separate and rejoin at different locations, e.g., the truck-drone tandem delivery system, and those locations are generally restricted to be customer locations. In a recent study, Chiang et al. (2019) presented a mixed-integer linear programming (MILP) formulation for the VRP with drones (VRPD), in which each vehicle is outfitted with a single drone. A Genetic Algorithm (GA) is proposed for large-sized benchmark instances. Given the complexity of the VRPD, Sacramento et al. (2019) proposed an ALNS algorithm with two removal and four insertion operators to solve large instances. In addition, a variant of VRPD considering one truck and multiple drones was investigated in Wang and Sheu (2019) because using multiple drones enhances the productivity of the delivery system. The authors presented an exact algorithm for instances with up to 13 customers and two docking nodes. Another variant of the VRPD is studied by Schermer et al. (2019) who proposed a new VRP with drones and en-route operations, expanding the drone take-off locations with discrete locations other than customer locations. An integrated Variable Neighborhood and Tabu Search algorithm was presented in their study. In another study, a delivery system with multiple drones serving each customer is studied (Chang & Lee, 2018). The authors considered vans carrying drones and parcels depart from a central depot and travels to different locations where drones are dispatched for their final delivery/pickup tasks. Those locations served as centers of customer clusters are determined by their proposed approach in a Euclidean plane. Karak and Abdelghany (2019) combined the pickup and the delivery requests within a kind of mothership system with stations and introduced the Hybrid Vehicle-Drone Routing Problem (or HVDRP). Vehicles visit stations to transport delivery items and drones, while drones are launched and collected only at stations. The authors modified the well-known Clarke and Wright algorithm (Clarke & Wright, 1964) to tackle the investigated problem. For related works on drone delivery systems, the reader is referred to two special issues of Networks (Agatz & Campbell, 2018) and COR (Murray & Smith, 2020) and a review paper by Otto et al., (2018).

The integration of traditional vehicles with robots is another innovative city logistics delivery concept. Simoni et al. (2020) investigated an integrated truck and robot delivery model and proposed a routing problem named as TSP-R, which share similarities with TSP-D. The TSP-R allows multiple customers to be visited in a robot sortie while the drone's capacity is limited to be one parcel. Boysen et al. (2018) studied a new vehicle-based robot delivery (TBRD) system. A truck loaded with demands of customers and robots travels to drop-off points. Then, the robots travel to their targeted destinations. There are a set of decentralized robot depots in the delivery system, where the truck can always replenish robots to capacity. After delivery, robots return to those depots. Yu et al. (2020) introduced a two-echelon delivery problem where multiple large vehicles cooperate with

their associated small autonomous vehicles. They proposed construction heuristics and a LNS-based hybrid metaheuristic algorithm. Recently, Cheng et al. (2020) investigated delivery robots for the use of urban logistics. The authors provided a new MILP model and MILP-based matheuristic algorithm to conduct various sensitivity analyses on small- to medium-sized instances.

In another related problem, Baldacci et al. (2017) investigated an optimization problem named VRP with Transshipment Facilities (VRPTF) and a branch-and-cut-and-price (BCP) algorithm is presented. Although the structure of its solutions resembles that of the VRPTWDR, the latter has more constraints which greatly strengthen the difficulties to solve it, including time windows, and the limited number of robots installed into a van.

The related literature is presented in Table 1 with the selected references. The second column of the table provides the problem acronyms used in corresponding references. 'n' and 'm' are used in the columns to report the number of vehicles and assistants. The fifth column indicates whether the assistant can move by itself and the sixth column shows whether the assistant is installed in a vehicle. Finally, the inclusion of time windows is reported in the seventh column and the solving approach is presented in the eighth column.

Table 1: A summary of previous works

Reference	Problem	Number of vehicles (s)	Number of assistant (s)	Mobility of assistant	Vehicle and assistant rendezvous	Time windows	Proposed algorithm
Ham (2018)	PDSTSP	n	m	√	×	×	Constraint programming
Agatz et al. (2018)	TSP-D	1	1	√	√	×	Route-first, cluster-second heuristics
Es Yurek & Ozmutlu (2018)	TSP-D	1	1	√	√	×	Iterative two-stage algorithm
Murray & Chu (2015)	FSTSP	1	1	√	√	×	Heuristic approach
Lin et al. (2011)	TTRPTW	n	m	×	√	√	SA
Nguyêˆn et al. (2018)	CTSPTW	1	1	√	√	√	MILP formulation
Chiang et al. (2019)	VRPD	n	1	√	√	×	GA
Sacramento et al. (2019)	VRPD	n	1	√	√	×	ALNS
Wang & Sheu (2019)	VRPD	n	m	√	√	×	Branch-and-price algorithm
Schermer et al. (2019)	VRPDERO	n	1	√	√	×	Hybrid VNS/TS
Chang & Lee (2018)	TSP-D	1	m	√	√	×	K-means, nonlinear programming
Karak & Abdelghany (2019)	HVDRP	1	m	√	√	×	Heuristic approach
Baldacci et al. (2017)	VRPTF	n	0	-	-	×	BCP
Alvarez & Munari (2017)	VRPTWMD	n	m	√	√	√	Exact hybrid method
Boysen et al. (2018)	TBRD	1	m	√	×	×	Multi-start local search procedure
Simoni et al. (2020)	TSP-R	1	1	√	√	×	Heuristics
Chen et al. (2020)	VRPTWDR	n	m	√	√	√	Matheuristic
Yu et al. (2020)	LV-SAV	n	m	√	√	√	Construction heuristics, metaheuristic
This work	VRPTWDR	n	m	√	√	√	ALNS

This paper considered a VRPTW variant arising in coordination of trucks and delivery robots in the last mile delivery in populated areas. There are two types of delivery resources in the considered delivery system: delivery vans and self-driving robots. Each van is installed with several robots. While the van is serving a certain customer, multiple robots could be dispatched to serve multiple customers at the customer's parking site. The van cannot leave before the end of all services of customers served by the van itself and its robots, and the return

of all its robots. Furthermore, time window constraints are included as they are common and necessary as well as they have an apparent impact on solutions. Finally, we improved the ALNS algorithm to handle instances with large number of customers. Therefore, the current research differs from the literature in time window constraints on customers, mapping relationship between the delivery van and robots, tactics on resource separation and reunion, number of routes, and solution approach. Because of the above-mentioned reasons, the existing methods proposed in the literature cannot directly be applied for large-sized instances.

3. Problem definition

We present the description of the VRPTWDR along with a MILP model.

3.1 Mathematical formulation

Given a fleet of homogeneous delivery vans, expressed as a set $K=\{1, 2,.., k\}$, each of them equipped with several self-driving robots, denoted by $R=\{1, 2, \ldots, m\}$, the operational task is to deliver small to medium sized parcels to a given set of customers, denoted by $C=\{1, 2, \ldots, n\}$. Customer's demand is presented with q_i and all customers have to be visited within a given time window $[l_i, u_i]$, where l_i and u_i are used to define the lower and upper bounds, respectively. A single depot, denoted by 0, also has a time window, denoted as $[l_0, u_0]$, and $V=\{0\} \cup C$ is defined. The time window in this study is considered to be hard. This means that the delivery can only happen within the time window even if the delivery resource (a van or a robot) arrives before than the lower bound. To ensure a feasible solution, it is assumed that the number of delivery vans is sufficient in each instance.

In our problem setting, customer can be served by a van or an autonomous delivery robot operating in coordination with the delivery van. All vans originate and end at the single depot. They can dispatch robots after they are parked at their target customers' parking sites. All robots are retrieved at the same locations where they are dispatched as the dispatch-wait-collect tactic is adopted. The delivery robot serves a single customer within a sortie due to the limitations on the capacity and each robot is allowed to be dispatched once at most at a site. The maximum radius of the robot, denoted by r^d, is defined due to the safety concern and the capacity of battery and d_{ij} is defined to be the distance between customer i and customer j. Each van has a limited capacity for robots and payload, denoted by $|R|$ and Q, respectively.

A parameter f_i^d is introduced to consider the customer preferences for delivery robots. This binary parameter is set to one if the customer i is served by a robot, and zero if not visited by a delivery robot. We note that, in some cases, customers may not be served by a delivery robot. For example, physical obstacles at the customer's site, customers' preferences, or the robot's capacity/energy limitation can limit the use of robots. As the two types of vehicles have different speeds, we also denoted s_{ij}^v and s_{ij}^d to represent traveling times between customers i and j for vehicles and robots, respectively. Finally, we differentiated service times for delivery vans and robots, denoted by t_i^v and t_i^d, respectively. The VRPTWDR makes a set of integrated decisions, including the number of vans used and their routes, the customers served by robots, and times and locations to dispatch robots to them and the studied objective is to minimize the sum of all routes' duration. Fig.1 illustrates an example solution of the VRPTWDR.

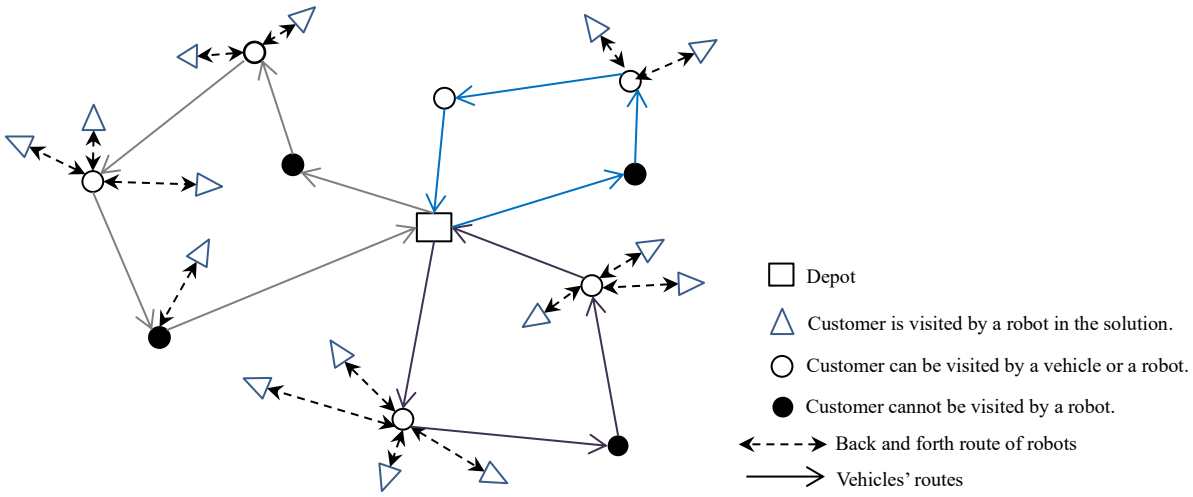


Fig.1. An example solution to the VRPTWDR.

A MILP formulation of the VRPTWDR is presented as follows based on the decision variables listed in Table 2.

Table 2: Decision variables

Notation	*Type*	*Description*
x_{ij}^{k}	Binary (0,1)	(=1) If a delivery van k goes to customer j from customer i, (=0) o/w
y_{ij}^{dk}	Binary (0,1)	(=1) If a delivery robot d loaded in van k arrives at i to visit node j, (=0) o/w
p_{ij}^{k}	Continuous	Total carried load of van k when it moves between customers i and j
a_i	Continuous	Arrival time of a van or delivery robot at customer i
b_i	Continuous	Delivery time at customer i's location
w_i	Continuous	Time span between b_i and the van's departure time from customer i

$$\text{minimize} \quad \text{OBJ} = \sum_{i\in V}\sum_{j\in V}\sum_{k\in K} s_{ij}^{v} x_{ij}^{k} + \sum_{i\in C}(b_i - a_i + w_i) \tag{1}$$

subject to:

$$\sum_{i\in V}\sum_{k\in K} x_{ij}^{k} + \sum_{i\in C}\sum_{d\in R}\sum_{k\in K} y_{ij}^{kd} = 1, \forall j \in C \tag{2}$$

$$\sum_{i\in C}\sum_{k\in K}\sum_{d\in R} y_{ij}^{kd} <= f_j^{d}, \forall j \in C \tag{3}$$

$$\sum_{j\in C}\sum_{d\in R} y_{ij}^{kd} \le |R| \sum_{j\in V} x_{ji}^{k}, \forall i \in C, k \in K \tag{4}$$

$$\sum_{j\in C}\sum_{k\in K} y_{ij}^{kd} \le 1, \forall i \in C, d \in R \tag{5}$$

$$\sum_{j\in C} x_{0j}^{k} \le 1, \forall k \in K \tag{6}$$

$$\sum_{i\in V} x_{ij}^{k} = \sum_{i\in V} x_{ji}^{k}, \forall j \in V, k \in K \tag{7}$$

$$\sum_{i\in V}\sum_{k\in K} p_{ij}^{k} - \sum_{i\in V}\sum_{k\in K} p_{ji}^{k} = q_j * \sum_{i\in V}\sum_{k\in K} x_{ij}^{k} + \sum_{i\in C}\sum_{k\in K}\sum_{d\in R} q_i y_{ji}^{kd}, \forall j \in C \tag{8}$$

$$p_{ij}^{k} \le (Q - q_i - \sum_{\theta\in C}\sum_{d\in R} q_\theta y_{i\theta}^{kd}) x_{ij}^{k}, \forall i \in C, j \in V, k \in K \tag{9}$$

$$b_i - a_j + w_i + s_{ij}^v \le M(1 - \sum_{k \in K} x_{ij}^k), \forall i \in C, j \in C \quad (10)$$

$$a_j - b_i - w_i - s_{ij}^v \le M(1 - \sum_{k \in K} x_{ij}^k), \forall i \in C, j \in C \quad (11)$$

$$a_i - a_j + s_{ij}^d \le M(1 - \sum_{k \in K} \sum_{d \in R} y_{ij}^{kd}), \forall i, j \in C \quad (12)$$

$$w_i \ge t_i^v * \sum_{j \in V} \sum_{k \in K} x_{ji}^k, \forall i \in C \quad (13)$$

$$(b_j - b_i + t_j^d + s_{ji}^d) - w_i <= M(1 - \sum_{k \in K} \sum_{d \in R} y_{ij}^{kd}), \forall i \in C, j \in C \quad (14)$$

$$a_i \le b_i, \forall i \in C \quad (15)$$

$$u_i \le b_i \le l_i, \forall i \in C \quad (16)$$

$$\sum_{i \in C} \sum_{j \in C} d_{ij} y_{ij}^{kd} \le r^d, \forall k \in K, d \in R \quad (17)$$

$$x_{ij}^k \in \{0,1\}, \forall i \in V, j \in V, k \in K \quad (18)$$

$$y_{ij}^{kd} \in \{0,1\}, \forall i \in C, j \in C, k \in K, d \in R \quad (19)$$

$$p_{ij}^k \ge 0, \forall i \in V, j \in V, k \in K \quad (20)$$

$$a_i \ge 0, \forall i \in C \quad (21)$$

$$b_i \ge 0, \forall i \in C \quad (22)$$

$$w_i \ge 0, \forall i \in C. \quad (23)$$

The studied objective (OBJ) (1) aims to minimize the total time duration in all routes for both vehicle(s) and delivery robot(s). More specifically, it is the sum of vans' traveling time and additional times that they are staying idle at customers' locations for the delivery and waiting delivery robots. Constraints (2) is used to ensure that each customer is served by a van or a delivery robot. Some customers may not be visited by a delivery robot as ensured in Constraints (3). Constraints (4) and (5) assume that each robot is used no more than once at any customer location. Also, each van can only be used at most once in a schedule, which is asserted in constraints (6). Constraints (7) restrict the vehicle flow conservation at every node in the network. Constraints (8) guarantee that the payload balance before and after a vehicle visits a customer. Constraints (9) ensure that vehicle capacity is not violated. Constraints (10) and (11) ensure the arrival time at customer *j* and the leave time at customer *i* if the van travels from customer *i* to customer *j*. Constraints (12)-(14) ensure the vehicle leaves the customer after its service is done and all the robots dispatched here have returned. Constraints (15) restrict that each service only begins after the arrival of a van or a robot. Inequalities (16) are the time window related constraints. The maximum radius limitation on robots is imposed by constraints (17). The domains of decision variables are finally listed in constraints (18)-(23).

3.2 Feasibility discussion on the VRPTWDR

When dealing with the well-known VRPTW, the consideration of total routes' duration is more prominent than the total traveled distance by all vehicles. However, given an objective of minimizing the sum of all routes' duration, departure times of vehicles are determined not only to ensure time windows requirements, but also to minimize the waiting times before and after visiting a customer. Obviously, the worst-case scenario for the VRPTWDR is that all customers are served by delivery vans as in the general delivery system, which means that no robot is dispatched to serve a customer. Therefore, it is necessary to promote deliveries by robots which result in the reduction of the total durations.

There are three conditions for the feasibility of dispatching a robot from customer i to serve customer j. First, the capacity constraint of delivery van must be satisfied after including node j to route where node i is located. Second, the walking distance by a delivery robot between customers i and j needs to be within its maximum walking radius. Finally, departure and arrival times of the vehicle and robots within the route must be synchronized. The first two conditions can easily be checked, while the last condition is hard to tackle as the time-related variables are essentially continuous. There might be infinite possibilities of those decision variables. Therefore, the configuration that results in the minimum route duration needs to be identified. To simplify the process of time feasibility check for a robot service operation during the solution search, we create a dummy time window ($[u_i^{'},\ l_i^{'}]$) and service time ($t_i^{v'}$) for customer i who is visited by a van.

In terms of dispatching a robot at customer i to visit customer j, the robot needs to arrive at j before the upper bound of customer's (j) time window, which implies the latest time on the van's arrival at customer i. Hence, the upper bound of dummy time window of customer i is determined by equation (24) after allocating a robot service operation for customer j.

$$u_i^{'} = \min(u_i, u_j - s_{ij}^{d}) \tag{24}$$

By contrast, the lower bound of dummy time window is affected by several factors, including time windows of customers i and j, vehicle service time (of customer i), robot service time (of customer j), and robot walking time between customers i and j. All possible conditions related to the synchronization of a van and robot are listed in Table 3 along with equations of the upper bound of the dummy time window, $l_i^{'}$. The principle in deriving of these equations is to minimize extra waiting (idle) time (solid rectangle with slash in Table 3) of the van at customer i. There are seven conditions in which idle time is unavoidable. For example, the van needs to arrive at customer i earlier than the lower of customer i's time window to dispatch robot for customer j due to the restriction imposed by the upper of customer j's time window (conditions seven and nine in Table 3). In another example, the delivery van needs to wait idle for the delivery robot's return due to the lower of time window of customer j (conditions one and four in Table 3) or the longer total service time (robot traveling time plus serving time) for customer j (conditions five, six and eight in Table 3).

Given $l_i^{'}$, dummy service time $t_i^{v'}$ can be easily calculated as the van leaves customer i after both customer i and customer j have been served and the robot has returned to the van. It equals to the difference of the earliest

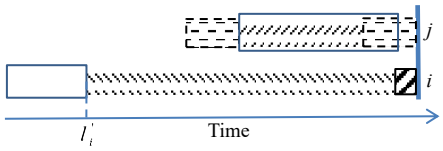
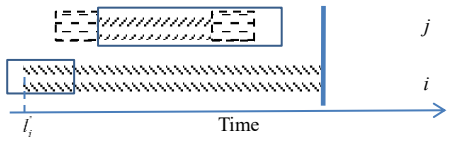
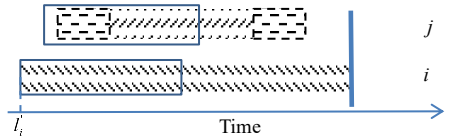
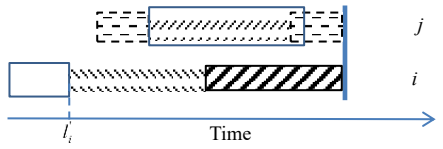
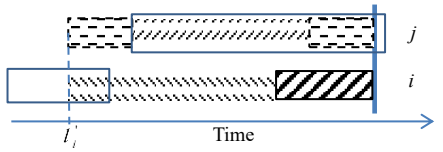
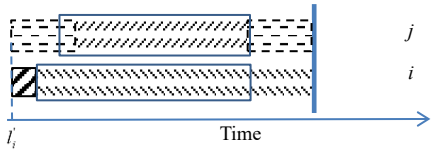
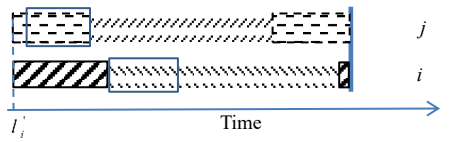
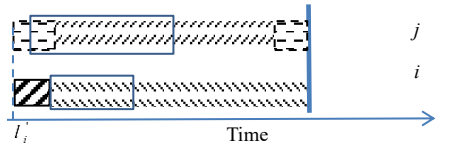
leaving time and $l_i^{'}$, which is described in equations (25).

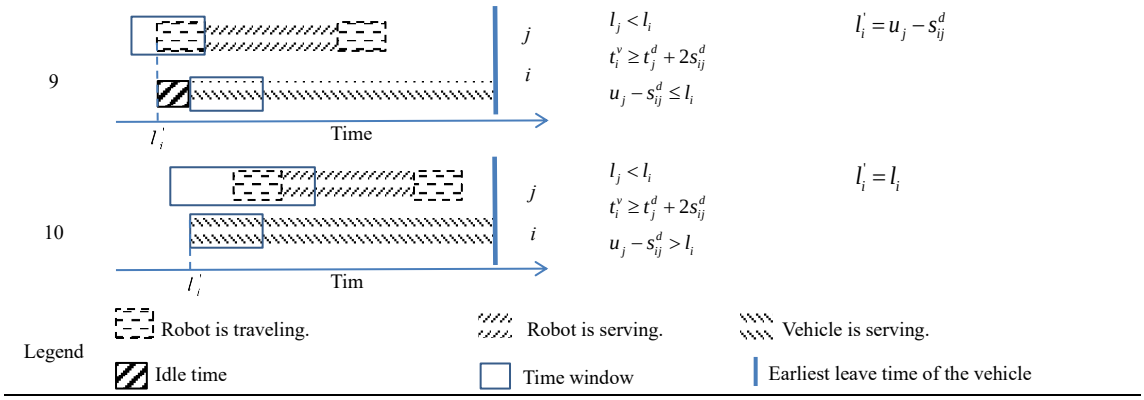
$$t_i^{v'} = \max(l_i + t_i^v, l_j + t_j^d + s_{ij}^d) - l_i^{'} \tag{25}$$

It is noted that deploying a robot for customer *j* from customer *i* would not deteriorate the objective if the resulted dummy time window and service time are equal to the original time window and service time of customer *i*. That is $l_i^{'} = l_i$, $u_i^{'} = u_i$ and $t_i^{v'} = t_i^v$. Similarly, it would not increase the objective when there is no idle time.

It is hard to solve the VRPTWDR optimally even for medium-sized instances. Therefore, an advanced metaheuristic algorithm (i.e., ALNS) is proposed to achieve high quality solutions with acceptable computational run times.

Table 3: Different conditions and calculation equations for lower bound $l_i^{'}$

Number	Condition: Illustration	Condition: Mathematical description	Equation
1		$l_j \geq l_i$ $t_i^v \geq t_j^d + 2s_{ij}^d$ $l_j + s_{ij}^d + t_j^d - t_i^v \geq u_i$	$l_i^{'} = u_i$
2		$l_j \geq l_i$ $t_i^v \geq t_j^d + 2s_{ij}^d$ $l_i < l_j + s_{ij}^d + t_j^d - t_i^v < u_i$	$l_i^{'} = l_j + s_{ij}^d + t_j^d - t_i^v$
3		$l_j \geq l_i$ $t_i^v \geq t_j^d + 2s_{ij}^d$ $l_j + s_{ij}^d + t_j^d - t_i^v \leq l_i$	$l_i^{'} = l_i$
4		$l_j \geq l_i$ $t_i^v < t_j^d + 2s_{ij}^d$ $l_j - s_{ij}^d \geq u_i$	$l_i^{'} = u_i$
5		$l_j \geq l_i$ $t_i^v < t_j^d + 2s_{ij}^d$ $l_i < l_j - s_{ij}^d < u_i$	$l_i^{'} = l_j - s_{ij}^d$
6		$l_j \geq l_i$ $t_i^v < t_j^d + 2s_{ij}^d$ $l_j - s_{ij}^d \leq l_i$	$l_i^{'} = \max(l_j - s_{ij}^d, l_i + t_i^v - t_j^d - 2s_{ij}^d)$
7		$l_j < l_i$ $t_i^v < t_j^d + 2s_{ij}^d$ $u_j - s_{ij}^d \leq l_i + t_i^v - 2s_{ij}^d + t_j^d$	$l_i^{'} = u_j - s_{ij}^d$
8		$l_j < l_i$ $t_i^v < t_j^d + 2s_{ij}^d$ $u_j - s_{ij}^d > l_i + t_i^v - 2s_{ij}^d + t_j^d$	$l_i^{'} = l_i + t_i^v - t_j^d - 2s_{ij}^d$



4. A metaheuristic algorithm for the VRPTWDR

An improved Adaptive Large Neighborhood Search (ALNS) algorithm for the VRPTWDR is proposed in this research. The framework of the ALNS was introduced by Ropke and Pisinger (2006) as a variant of LNS algorithm of Shaw (Shaw, 1998). It has been adapted for a wide range of VRPs and its efficiency has also been proven (Franceschetti et al., 2017; Sacramento et al., 2019). With the help of SA acceptance function, it improves a current solution by looking for a better one (Demir et al., 2012). By destroying and repairing a relative part of current solution, the algorithm changes large part of the solution at each iteration. ALNS generally contains multiple operators that are chosen based on their improvement performance.

We begin with providing the overall procedure of the ALNS used in this paper. Then, we discuss how an initial solution is created and explain the adaptive mechanism. Finally, the operators are introduced in section 4.4.

4.1 A framework of the metaheuristic algorithm

The general ALNS framework used in this research is illustrated in Fig.2. Begin with an initial solution s_0, iterations are bundled into segments which are run until the termination criteria is met. With a current solution s, a neighborhood solution s' is generated using both types of operators. These include one removal operator to destroy a feasible solution and one insertion operator to make the solution feasible again. Both types of operators are selected using an adaptive mechanism described in Section 4.3. Then the s can only be updated by an s' based on the SA acceptance criterion. This process repeats until the termination criteria is satisfied.

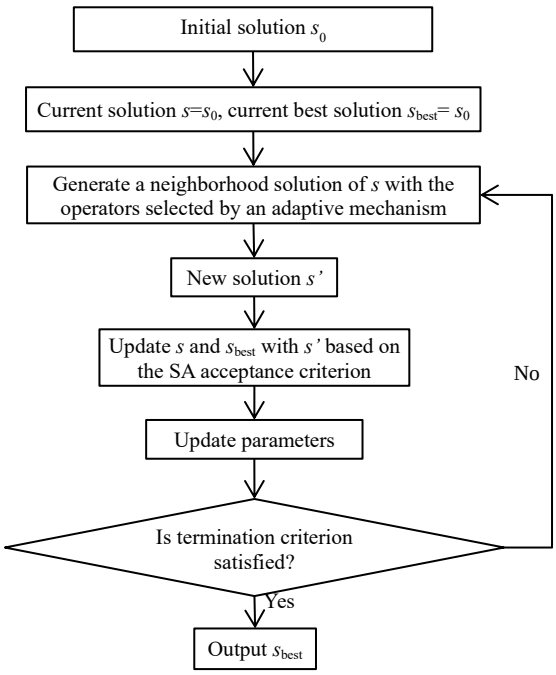


Fig.2. An illustration of the general framework of ALNS algorithm.

The objective value is denoted by $f(s)$. A new solution s' is always approved if $f(s') \leq f(s)$ and can be approved with a probability function controlled with a parameter T (temperature) if $f(s')>f(s)$. The probability function is calculated as $e^{-(f(s')-f(s))/T}$. The value of T is originally set to $T_0=\eta \cdot f(s_0)$ and is decreased after each iteration with the following formula $T=c \cdot T$, where $0<c<1$ represents the cooling rate. Instead of using a fixed maximum termination criterion, our ALNS stops when a predefined number of iterations is run without any improvement in the value of best solution.

4.2 The first feasible solution

The initial solution is generated with the *Greedy Insertion method* which mainly has two steps. First (step 1), the customer node with the highest demand is chosen to be served by a van in a new route as we considered customers for insertion in decreasing order of their demands. Then (in step 2), we calculate the insertion cost for each and every customer that is not included in existing routes yet. The smallest insertion cost is chosen, and the corresponding insert operation is implemented. Positions for a customer in the current route refer to two types of services: van-visit and robot-visit. The upper limit of inserting a customer is to serve it with a new van individually. Step 2 is redone until all customers are inserted into feasible route(s).

An insertion cost for a customer in this paper is calculated as the difference in the objective value after inserting it into existing routes. For example, let f represent the objective value of current partial solution without customer i, and f' represent the one for the solution after i is inserted, then the insertion cost for i is $f'-f$. Since a delivery robot's payload capacity violation is always the main cause that makes a customer inaccessible for robots, the customer with the highest demand is a reasonable choice to begin the procedure.

4.3 The adaptive mechanism

Each operator has a score that implies its probability to be selected at each iteration and this score changes according to its performance at the end of every segment. Initially, operators start with same value (i.e., 1). After segment i has finished, scores of operators are updated as $w_{mi}=w_{m(i-1)}(1-r)+r\pi_{mi}/\theta_{mi}$, where $r\in[0, 1]$ is defined as reaction factor, π_{mi} is a quantification of performances of operator m in segment i, and θ_{mi} shows how many times the operator is used in segment i. As suggested by Ropke and Pisinger (2006), three scores ($\sigma_1, \sigma_2, \sigma_3$) are defined to reward operators' performances when the solution newly generated is accepted and π_{mi} is the sum of all scores gained by operator m in segment i. The score of an operator is only rewarded when it obtains a new solution which is accepted. The three scores are used when $f(s') < f(s_{\text{best}})$, $f(s') < f(s)$, and $f(s') \geq f(s)$.

4.4 Removal and insertion operators

Begin with a null removal set and a predetermined removal number Γ, the removal operator selects customers and puts them into the removal set until all elements in the set is no less than Γ. Then, the insertion operator is used to reinsert all removed customers into a partially destroyed solution until the set becomes empty again. We proposed seven removal and five insertion operators.

4.4.1 Removal operators

It is stipulated that the customers visited by robots from customer i's location are removed at the same time when i is removed. Thus, the final number of removed customers may exceed Γ, although customers to be removed are selected one by one. Additionally, to diversify the search, we introduced a certain level of randomness for the selection of customers in operators. We used a random number y generated from a uniform distribution function between zero and one. The $[y^{\delta}n]$-th customer on the list is chosen, where δ is a parameter controlling the degree of the randomness and n is the number of the potential customers that need to be removed. In total, we used seven operators for destroying a solution. Except the sixth operator which is exclusive for the VRPTWDR, the others are either adapted or inspired by existing work (see Shaw, 1998; Pisinger & Ropke, 2007; Demir et al., 2012, Eshtehadi et al. 2020).

(1) *Shaw removal*: It is logical to pull out a set of customers that are somehow related as initially suggested by Shaw (1998). The relatedness concept attempts to make sure that the re-insertion of removed nodes is dependent of other nodes. The relatedness between customers i and j in this study is quantified by $RC(i,j)$ as follows:

$$RC(i,j)=\varphi_1 d_{ij}+\varphi_2(|l_i-l_j|+|u_i-u_j|)+\varphi_3|q_i-q_j|+\varphi_4 sr_{ij}. \quad (26)$$

The relatedness measure consists of four terms: distance, time window, demand and route. φ_1, φ_2, φ_3, and φ_4 are the weights for them, respectively. Let sr_{ij}= -1 if i and j belong to the same route, and zero o/w. Here, d_{ij}, ($|l_i-l_j|+|u_i-u_j|$), and ($|q_i-q_j|$) are normalized such that $RC(i,j)\in[0, \varphi_1+\varphi_2+\varphi_3+\varphi_4]$. The smaller $RC(i,j)$ means higher relatedness between customers.

First, a customer is randomly chosen, and then the relatedness between a set of customers and the already

removed ones are calculated. Finally, based on the ascending order of the relatedness value, the next customer to be removed is determined, which is repeated until there are predefined number of customers in the removal set.

(2) *Distance removal*: With this operator, customers that are related from a distance perspective is removed from the available routes. It is similar to the previous operator with φ_1=1, and φ_2= φ_3= φ_4=0.

(3) *Time Window removal*: Like first two operators, this operator considers the relatedness in terms of time windows. In the VRPTWDR, both feasible exchanges and robot service deployments are more likely to be implemented among customers with similar time windows. Equation (26) can also be used with φ_2 =1 and φ_1= φ_3= φ_4=0.

(4) *Random removal*: It randomly selects customers to be removed and selects one customer each time. The process repeats until the total number of removed customers is no less than Γ.

(5) *Worst removal*: It attempts to remove customers that contribute most to the objective value. First, removal cost for each customer is calculated by the difference between the objective values of routes with and without it. Because the corresponding customers visited by a robot are removed when the customer visited by a van is removed, the removal cost of a van-visit customer is divided by the total number of customers removed at a time.

Removal customers are selected based on the descending sequential of their removal cost. As in our problem the removal cost may change after some customers are removed, one customer is selected each time and the removal cost is recalculated after each removal operation. The removal of customers continues until the number of customers in the removal set is no less than Γ.

(6) *Worst robot-visit removal*: As discussed in Section 3.2, deploying a robot-visit may not contribute an increase to the objective value necessarily, or just contribute a little (the idle time), which results in removal costs as zero or a very small value. Generally, the removal cost of robot-visit customers is smaller than that of van-visit customers. Therefore, it seems unlikely to select a robot-visit customer solely. For this reason, the worst robot-visit removal is deliberately designed. This operator divides the removal procedure into two sequential parts. In the first part, robot-visit customers are removed base on the same scheme of worst removal and only the customers with positive removal costs are considered. The procedure repeats until there are enough customers in the removal set or there is no robot-visit customer with positive removal cost. Then the worst removal procedure is implemented if the number of customers in the removal set is less than Γ.

(7) *Route removal*: It discards a complete route and is helpful in reducing the total number of routes (vans). The procedure randomly picks a route whose customers are no more than Γ. When no such route exists in current solution, the procedure randomly chooses a route.

4.4.2 Insertion operators

The insertion operators reinsert the removed customers into the current partial solution one by one. As starting a new route is the upper limit of inserting a customer, the operators ensure the feasibility of the new solution. For the greedy-oriented operators (e.g., best insertion and greedy insertion), the corresponding versions

with noise functions are introduced as well, because the noise operators are useful in avoiding alternative solutions when same customers are removed, especially for the myopic insertion heuristics.

(1) *Best insertion*: It reinserts a removed customer into its best possible position in existing routes or create a new route with it. The two types of services in the VRPTWDR: with robot and with driver (vehicle), are considered simultaneously in this operator. Meanwhile, we considered four variants of this operator: the first three are with a slight difference on the criterion to determine the sequence of customers to be inserted and the fourth one is featured by its insertion priority. They are random selection, time window priority selection, vehicle visit priority selection and robot visit priority insertion, respectively. Random selection picks a node randomly from the removal set then inserts it into the position with a minimum insertion cost. In the second method, customers in the removal set are selected according to the width of their time windows and the one with the tightest time window is selected in each iteration. In the third method, driver priority selection first chooses customers where robot-visits are unaccepted. The logic behind is that the more vehicle-visit customers in routes, the more possible insertion positions are spared for the uninserted customers. Finally, the fourth method considers deploying a robot-visit for the customer to be inserted unless there is no feasible one. Then van-visit positions are considered. A weight is set for each variant. They are g^1, g^2, g^3, g^4 ($g^1+g^2+g^3+g^4=1$). A roulette-wheel mechanism is applied at each time the operator is implemented.

(2) *Best insertion with a noise function*: It is a special variant of the previous operator, which looks for the so-called best location for a removed customer by using a degree of freedom. The additional value is calculated as $NewOBJ=RealOBJ+\bar{d}\mu\varepsilon$, where $\bar{d}$ is the highest travel time among customers, μ is a noise control parameter, and ε is a random number from the interval of real numbers [-1, 1].

(3) *Greedy insertion*: In each iteration of inserting a customer, insertion costs for inserting each customer in the removal set into every feasible position are calculated. The customer and the insertion position related to the minimum insertion cost are selected and then the corresponding insertion operation is carried out.

(4) *Greedy insertion with a noise function*: It is a variant of the greedy insertion operator and also utilizes the same noise function as provided for the second insertion operator.

(5) *Regret insertion*: It aims to enhance the solution quality by using a look-ahead information for choosing the customer to be inserted. 2-regret criterion is used in this paper. Let Δf_i denote the difference between its minimum and second minimum insertion costs, the operator decides the next customer to be inserted using the formula $i^*=\text{argmax}\{\Delta f_i\}$.

5. Computational experiments

This section provides the computational results related to the performance of the ALNS. We also present sensitivity analyses of applying delivery robots as assistances in last mile delivery of city logistics. First, we discuss the generation of test instances and how we set the values of parameters. And finally, computational results are presented.

The ALNS was implemented in MS Windows with MATLAB R2019b (MathWorks, 2019), and run on a laptop computer with Intel i5-3610QM CPU @ 2.30GHz processor with 4GB RAM. The mathematical model

is solved with IBM CPLEX 12.10.0 solver (IBM, 2019).

5.1 Benchmark instances

As the VRPTWDR is a new extension to the VRPTW, there are no available benchmark instances in the literature that can fit to our problem setting. First, we have created new instances based on the randomly chosen Postcodes from Cardiff, United Kingdom. We have used the geographic zones of CF10, CF14, CF23, CF24, CF51, and CF52 postcodes to generate a number of instances, ranging from 15-node to 200-node customers. After deciding the postcodes of customers, we have randomly generated demand and time windows for each customer. We have initially set 90% to decide the customer preferences on robot-visits. These instances are named as *Cardiff_k_l*, where *k* presents the number of customers and *l* represents the order of instance in each set.

Second, the well-known VRPTW benchmark instances, Solomon's instances (Solomon, 1987), were incorporated, which have three sets with regards to geographical locations of customers' nodes. These sets are called as Random (R), Clustered (C) and Random Clustered (RC). One the one hand, the standard instances with 100 customers were used to show effectiveness of the ALNS on the VRPTW. On the other hand, we modified the instances with 25 customers to analyze the performance of the algorithm on the VRPTWDR. We only used C and RC type instances in this paper as they are closer to the real delivery scenario in urban logistics.

Technology-related parameters of delivery robots considered in the mathematical formulation of the VRPTWDR may impact the efficiency of using delivery robots in collaboration with the traditional delivery vehicles in city logistics. Although there is no exact value for those parameters, reasonable values can be estimated according to the features of previous promising delivery robot applications. For example, the Starship robot (Starship Technologies, 2014) travels at pedestrian speed (5 km/h) and weighs no more than 100 pounds (45 kilograms) within a 4-mile (6 km) radius. Its payload is limited to less than 2.6 kg (Kottasova, 2016). The ANYmal (ANYbotics AG, 2019) weighs 66 pounds (30 kg) and is capable of carrying a parcel with a weight up to 10 kg. It can also travel at 1 m/s with batteries for more than 2 hours working time (ANYbotics, 2019). The FedEx SameDay Bot, Roxo™ (FedEx, 2019) has a maximum speed of 10 mph (16 km/h) and 3 miles' radius. It is capable of trundling over unpaved surfaces and tackling steps for home deliveries (Vincent, 2019).

Therefore, it is reasonable to set the maximum payload capacity of robots to 10 kg in our numerical experiments. A customer i with a demand larger than the maximum load capacity of a robot is set to be inaccessible for robots (i.e., f_i^d=0). As delivery vans are usually adopted in last mile delivery service, 1,200 kg is a fair approximation of their payload capacity for the delivery of parcels. Although the Mercedes-Benz futuristic electronic van has an ability of eight delivery robots (McFarland, 2016), we set the maximum robots in a van as four, which may be more realistic for the applications in the near future. The speed of both vehicles is defined based on average speed values. Given the delivery van travels on urban street at a congestion speed, a speed of 32 km/h is justifiable. In terms of robots' speed, 10 km/h is expected to make the delivery with robots more promising. Finally, delivery robots generally have enough battery power for a relatively long autonomy, however, a maximum of 0.5 km radius ensure that the robot is generally within the driver's sight when they are

used as a kind of delivery assistant.

Finally, to tentatively understand the difference between delivery systems using robots and drones as assistants, we generated another 150 instances following the study of Poikonen and Golden (2020). In their multi-visit drone routing problem (MDVRP), all customers were served by drones which need to be launched/retrieved at specialized locations other than customer locations. Therefore, additional potential parking places are incorporated. In terms of their instances, all potential parking places (V) and all locations (C) are randomly chosen from a uniform distribution over a 25km by 25km square area and 100 instances were generated based on the combinations of |V|={25, 50} and |C|={25,50}. Moreover, the average objective values for different drone parameters were listed and analyzed. Since, in the VRPTWDR, the delivery van is allowed to park at customer locations, we expanded the combinations through setting |V|={0, 25, 50}. Because those instances are used to show the difference between two assistant application modes, specifically, the influence of the introduction of potential parking places, we used the same parameters for the assistant in the VRPTWDR as that in Poikonen and Golden (2020) in those instances.

5.2 Performance of the ALNS

The tuning of parameters is presented in Appendix, while this section investigates the efficiency of the dynamic range for the number of removal customers and the performance of the operators.

5.2.1 An analysis of dynamic range for removing customer nodes

The number of customers removed by the selected removal operator in each iteration is a crucial parameter in the ALNS. If only few customers are removed, it might not be possible to escape from a local minimum. If many customers are removed, the insertion operators may fail finding a better solution and it also costs higher computational effort. Generally, diversity is more important at the start stage of the algorithm to expand the solution space for searching, while a quick convergence is expected at later stage. To this end, we conceived a scheme of dynamic range for the number of removal nodes: the upper bound changes at the end of each segment while the lower bound stays the same during the course of the algorithm.

Table 4: Results on different ranges of the allowable number of removal nodes.

Instance	[2, ⌈0.4|*N*|⌉]	[2, ⌈0.15|*N*|⌉]	[⌈0.15|*N*|⌉~⌈0.4|*N*|⌉]	0.15|*N*|	Dynamic Range
Cardiff_100_01	1.27	1.71	3.97	6.52	0.00
Cardiff_100_02	0.00	0.32	5.28	1.73	0.58
Cardiff_100_03	1.91	2.71	2.68	4.03	0.00
Cardiff_100_04	3.03	0.00	2.75	3.15	0.35
Cardiff_100_05	0.00	0.89	1.72	1.23	1.93
Cardiff_100_06	0.64	0.00	2.93	2.79	0.53
Cardiff_100_07	3.01	2.03	1.24	0.00	1.41
Cardiff_100_08	0.00	1.37	4.99	4.34	0.90
Cardiff_100_09	1.02	0.69	0.00	1.39	0.67
Cardiff_100_10	6.03	0.10	2.42	2.30	0.00
Cardiff_100_11	0.97	1.70	0.00	2.18	0.51
Cardiff_100_12	0.44	0.13	2.87	3.01	0.00
Cardiff_100_13	1.47	2.85	3.47	0.00	2.03
Cardiff_100_14	0.40	0.85	1.74	1.77	0.00
Cardiff_100_15	4.63	0.00	4.08	4.88	0.21
Cardiff_100_16	1.70	0.58	1.36	0.00	0.40
Cardiff_100_17	3.40	1.24	4.05	3.55	0.00
Cardiff_100_18	0.63	0.00	2.96	3.02	2.66
Cardiff_100_19	0.00	3.09	3.03	2.73	1.90
Cardiff_100_20	1.43	0.35	1.57	1.76	0.00

Average	1.60	1.03	2.66	2.52	**0.70**

We set the maximum number of removal nodes as $MaxR_0=\lceil 0.3|C| \rceil$ and minimum removal nodes as $MinR=\lceil 0.06|C| \rceil$. At the end of segment i, the maximum number of nodes are changed as $MaxR_0=\max(MinR, \lceil MaxR_0 \cdot \zeta \rceil)$, where ζ is a parameter controlling the decrease speed of the upper bound. To validate this, we conducted experiments on all Cardiff instances with 100 customers. Several typical ranges were set as the control group according to existing literature of ALNS and the results (percentage of the deviation) are presented in Table 4. To ensure the fairness of the comparison, we set a total time limit in the experiments, which is 600 seconds.

5.2.2 Operators

We studied the computational performance of all operators. We solved all instances from the set of Cardiff instances. We defined a metric %IBest/%Usage to measure the performance of operators, where, %IBest is the percentage of iterations in which a new feasible solution is better than the current best solution, and %Usage is the percentage of total iterations in which a removal/insertion operator is utilized. Because the new accepted solution is rewarded as well as the new best solution in the adaptive mechanism for selecting operators, the most used operators are not necessarily those which get the new best-solutions most times. This metric helps to find out which operators are likely to obtain new best solutions.

Table 5: Relative performance of the operators (%IBest/%Usage).

Operators	Number of customers							
	15	25	50	75	100	150	200	Average
Removal								
Shaw	1.04	0.90	1.12	1.04	0.90	1.56	0.89	1.07
Distance	2.00	1.09	1.02	1.04	1.14	1.23	0.92	1.21
Time window	0.94	1.62	0.87	0.93	0.85	1.11	1.11	1.06
Random	0.78	1.09	1.21	1.02	0.92	1.02	1.52	1.08
Worst	1.31	1.18	1.53	1.25	1.37	1.29	0.97	1.27
Worst robot-visit	0.42	0.78	0.39	0.62	0.94	0.44	0.67	0.61
Route	0.39	0.01	0.24	1.30	0.77	0.67	0.96	0.62
Insertion								
Best	1.53	1.88	1.48	1.28	1.25	1.14	1.09	1.38
Best with noise	0.86	0.40	0.31	0.43	0.64	0.92	0.38	0.56
Greedy	1.46	1.48	1.18	1.15	1.19	1.35	1.14	1.28
Greedy with noise	0.42	0.36	0.53	0.57	0.97	0.44	1.41	0.67
Regret	1.23	0.44	0.34	0.53	0.87	0.42	1.11	0.71

We presented the results in Table 5. In terms of removal operators, the worst removal is the one that contributes most to the new best solutions while the route removal and worst robot-visit removal contribute more to the solution diversity. For insertion operators, the best insertion and greedy insertion have a strong ability in finding the new best solutions, while their noise versions are responsible for adding diversity. What's more, the regret insertion operator does not perform very well, compared to other operators with noise function.

5.2.3 Performance of the ALNS on VRPTW instances

To verify the performance of the ALNS, we first applied it on 100 customer nodes Solomon's VRPTW instances. The results are listed in Table 6, comparing to the results obtained by an optimal algorithm (OPL.), which are compiled from Milthers (2009) and Baldacci et al. (2011). Table 6 reports their objective values (Obj.),

computational times (CPU) and the percentage deviations (Dev.) on objective values of the best-found solutions by the developed ALNS algorithm. Because our implementation of the ALNS is not deliberately designed for the VRPTW, there are gaps between our results and the optimal ones on a few instances. However, for all instances, the gaps are no greater than 1.95%, which is good and reasonable for a newly implemented algorithm.

Table 6: Results of the algorithm on VRPTW instances.

Solomon_100	OPL.	ALNS			Solomon_100	OPL.	ALNS		
	Obj.	Obj.	CPU(s)	Dev. (%)		Obj.	Obj.	CPU(s)	Dev. (%)
C101	827.3	827.3	72.1	0.00	RC101	1,619.8	1,633.2	134.3	0.83
C102	827.3	827.3	65.6	0.00	RC102	1,457.4	1,457.4	119.8	0.00
C103	826.3	826.3	69.8	0.00	RC103	1,258.0	1,282.5	130.7	**1.95**
C104	822.9	822.9	68.4	0.00	RC104	1,132.3	1,132.3	118.4	0.00
C105	827.3	827.3	67.9	0.00	RC105	1,513.7	1,513.7	128.5	0.00
C106	827.3	827.3	72.6	0.00	RC106	1,372.7	1,392.1	115.3	1.41
C107	827.3	827.3	71.8	0.00	RC107	1,207.8	1,209.3	119.4	0.12
C108	827.3	827.3	71.4	0.00	RC108	1,114.2	1,121.3	117.6	0.64
C109	827.3	827.3	68.7	0.00	RC201	1,261.8	1,261.8	117.8	0.00
C201	589.1	589.1	53.8	0.00	RC202	1,092.3	1,092.3	123.2	0.00
C202	589.1	589.1	55.5	0.00	RC203	923.7	930.9	121.1	0.78
C203	588.7	588.7	58.9	0.00	RC204	783.5	783.5	108.2	0.00
C204	588.1	588.1	59.7	0.00	RC205	1,154.0	1,159.1	129.8	0.44
C205	586.4	586.4	58.3	0.00	RC206	1,051.1	1,054.1	116.4	0.29
C206	586.0	586.0	53.3	0.00	RC207	962.9	973.3	105.3	1.08
C207	585.8	585.8	56.7	0.00	RC208	776.1	782.1	102.9	0.77
C208	585.8	585.8	54.6	0.00					
Average			63.48	**0.00**				119.29	**0.52**

5.2.4 Comparing the ALNS with the commercial optimizer on small-sized VRPTWDR instances

Hereby, we studied the computational performance of our ALNS on the VRPTWDR. We used all instances with 15 and 25 customers from the Cardiff instances set to compare the ALNS algorithm with CPLEX using the MILP formulation. Without losing reasonability, we set 30 minutes limit on the computational time of CPLEX.

Table 7: Performance of the metaheuristics for Cardiff instances with 15-node and 25-node.

	Cardiff_15							Cardiff_25						
	CPLEX			ALNS				CPLEX			ALNS			
Instance	Obj.	CPU (sec)	RV	Obj.	CPU (sec)	RV	Dev. (%)	Obj.	CPU (sec)	RV	Obj.	CPU (sec)	RV	Dev. (%)
01	113.44	11.44	4	113.44	0.44	4	0.00	170.04	80.40	8	170.57	1.46	8	0.31
02	118.17	7.12	3	118.17	0.20	3	0.00	136.82	69.71	11	136.98	2.21	11	0.12
03	128.27	21.28	3	128.27	0.47	3	0.00	175.19	1,800*	8	175.19	4.58	8	0.00
04	139.86	9.87	2	139.86	0.23	2	0.00	172.28	22.82	6	172.28	1.51	6	0.00
05	132.33	6.87	3	132.33	0.21	3	0.00	169.50	34.23	9	169.50	1.66	9	0.00
06	118.73	8.48	4	118.73	0.23	4	0.00	169.22	47.22	9	169.22	2.32	9	0.00
07	124.33	7.01	3	124.33	0.81	3	0.00	185.00	105.57	8	185.00	2.44	8	0.00
08	121.54	7.21	3	121.54	0.22	3	0.00	161.78	7.61	9	161.78	0.57	9	0.00
09	144.43	6.45	1	144.43	0.21	1	0.00	158.38	85.12	9	158.38	2.87	9	0.00
10	126.00	11.12	3	126.00	0.22	3	0.00	162.68	15.19	9	162.68	1.42	9	0.00
11	118.56	7.09	4	118.56	0.47	4	0.00	155.65	12.44	10	155.65	3.73	10	0.00
12	121.22	8.72	3	121.22	0.68	3	0.00	179.55	23.57	7	179.55	1.36	7	0.00
13	118.46	10.16	3	118.46	0.76	3	0.00	184.89	1,800*	7	181.90	4.36	**7**	-1.62
14	134.78	13.88	2	134.78	0.20	2	0.00	162.87	47.83	8	162.87	5.10	8	0.00
15	90.67	6.75	7	90.67	0.21	7	0.00	167.96	945.27	10	167.96	1.68	10	0.00
16	135.24	9.48	2	135.24	0.20	2	0.00	179.65	105.13	6	179.65	1.47	6	0.00
17	144.60	9.60	1	144.60	0.20	1	0.00	166.31	13.38	9	166.90	2.52	9	**0.35**
18	126.61	7.41	2	126.61	0.22	2	0.00	159.30	38.45	9	159.30	3.03	9	0.00
19	130.46	9.06	2	130.46	0.21	2	0.00	152.54	15.59	11	152.54	1.88	11	0.00
20	126.39	8.48	3	126.39	0.23	3	0.00	190.00	1,296.73	6	190.00	2.08	6	0.00
Average		9.27	2.9		0.33	2.9	**0.00**		-	8.45		2.41	8.45	**-0.04**

*: best solution within a time limit of 30 minutes.

We show the results in Table 7. Besides Obj., CPU and Dev., the number of robot-visits deployed (RV) in solutions is also reported. The metaheuristics can reach an optimal solution of all instances with 15 customers

using a relatively less CPU times than CPLEX. Among the twenty Cardiff instances with 25 customers, CPLEX cannot provide optimal solutions for two instances by the defined time limit. Except three with minor gap (no more than 0.35%) to the optimal solution obtained by CPLEX, the metaheuristics can also obtain solutions same with or even better (one instance) than those of CPLEX. Moreover, the numbers of robot-visits deployed in instances with 25 customers show to be larger and more stable than that of instances with 15 customers. It is explained that since the robot travels at a speed lower than the van, the geographical locations and distribution of customers have an important role in the effectiveness of drone application as delivery assistants and higher density of customer distribution is preferred.

Table 8: ALNS outperforms OPL on Solomon_25 instances.

	Solomon_25 with 16-customer selected instances					Solomon_25				
	CPLEX		ALNS			CPLEX		ALNS		
Instance	Obj.	CPU (sec)	Obj.	CPU (sec)	Dev. (%)	Obj.	CPU (sec)	Obj.	CPU (sec)	Dev. (%)
C101	1,204.41	414.98	1,204.41	2.87	0.00	2,060.19	1,800*	2,060.19	7.78	0.00
C102	840.49	51.38	840.49	3.25	0.00	1,419.13	1,800*	1,479.29	10.16	4.24
C103	840.49	51.38	840.49	3.25	0.00	1,053.21	1,800*	1,032.72	6.40	-1.95
C104	822.66	40.37	822.66	2.24	0.00	994.38	1,800*	956.36	11.43	-3.82
C105	937.95	34.56	951.18	4.05	1.41	1,625.72	1,800*	1,581.98	14.55	-2.69
C106	1,158.92	171.48	1,158.92	2.24	0.00	2,039.19	1,800*	2,039.19	9.34	0.00
C107	832.64	26.73	832.64	3.40	0.00	1,140.88	1,800*	1,139.92	6.52	-0.08
C108	720.83	48.65	720.83	3.32	0.00	1,074.17	1,800*	1,056.26	8.31	-1.67
C109	676.01	10.22	676.01	2.40	0.00	869.11	1,800*	910.43	13.52	4.76
C201	1,106.64	258.42	1,106.64	4.15	0.00	2,353.75	1,800*	2,419.07	12.32	2.77
C202	939.13	9.81	939.13	3.74	0.00	2,004.27	1,800*	2,019.43	8.24	0.76
C203	805.97	17.57	850.48	3.43	0.00	1,994.43	1,800*	2,009.59	7.45	0.76
C204	783.62	13.49	783.62	3.58	0.00	1,045.62	1,800*	1,094.73	9.23	4.70
C205	1,014.71	55.86	1,014.71	3.60	0.00	2,075.38	1,800*	2,075.38	12.64	0.00
C206	920.82	17.84	920.82	3.55	0.00	1,733.10	1,800*	1,810.16	7.52	4.45
C207	818.88	23.58	818.88	3.11	0.00	1,611.70	1,800*	1,623.64	8.05	0.74
C208	868.41	18.62	868.41	3.49	0.00	1,331.08	1,800*	1,331.08	8.11	0.00
RC101	471.57	86.79	471.57	3.29	0.00	590.24	320.77	590.24	8.69	0.00
RC102	451.36	1,800*	451.36	3.72	0.00	564.92	1,800*	568.55	8.23	0.64
RC103	428.80	1,800*	428.84	3.82	0.00	542.19	1,800*	538.27	9.32	-0.72
RC104	417.72	1,800*	417.72	5.59	0.00	518.89	1,800*	518.37	15.12	-0.10
RC105	454.54	1,497.01	454.54	3.39	0.00	587.30	1,800*	586.60	12.00	-0.12
RC106	422.88	1,800*	422.88	3.41	0.06	537.94	1,800*	537.94	19.05	0.00
RC107	400.81	1,800*	400.81	3.44	0.00	490.82	1,800*	490.82	20.32	0.00
RC108	400.81	1,800*	400.81	3.54	0.00	488.42	1,800*	488.42	16.22	0.00
RC201	651.20	111.03	651.20	5.81	0.00	919.10	1,800*	893.03	17.80	-2.84
RC202	570.33	1,800*	570.33	3.88	0.00	869.30	1,800*	798.21	13.21	-8.18
RC203	429.62	1,800*	430.34	3.74	0.17	698.04	1,800*	651.67	15.76	-6.64
RC204	414.65	1,800*	414.65	3.76	0.00	687.00	1,800*	618.41	15.46	-9.98
RC205	510.13	1,800*	510.70	5.53	0.11	874.61	1,800*	806.32	11.69	-7.81
RC206	551.63	1,800*	549.95	3.99	-1.03	761.08	1,800*	717.13	7.64	-5.77
RC207	492.18	1,800*	492.18	5.64	0.00	696.46	1,800*	591.36	18.82	-15.09
RC208	370.72	855.78	370.72	4.50	0.00	465.18	1,800*	453.06	9.53	-2.60
Average				3.23	**0.02**				11.59	**-1.38**

*: best solution within a time limit of thirty minutes.

Additionally, experiments on Solomon instances with 25 customers are conducted. As the optimal solutions of those instances for the VRPTW are known, the numbers of vans in the fleet are restricted to that in the optimal solutions. Because only few of them can be solved to optimality considering the computational time limit, for each instance we randomly picked 16 customers to generate smaller instances. We show the details of results in Table 8, which indicates that CPLEX's performance is related to data characteristics because the computational times for the same size instances fluctuate dramatically. By contrast, the ALNS is more stable. In terms of solution quality, the heuristics generates solutions that no worse than those obtained by CPLEX on 90.9% of the instances. Among the few instances, the maximum deviation between the objective values of solutions obtained

by the metaheuristics and CPLEX is less than 1.41% and the average deviation on all instances is only 0.02%. The results of Solomon instances with 25 customers are also listed in Table 8, showing the better performance of the ALNS both on computational time and solution quality. Among all instances, only one can be solved to optimality by the CPLEX solver in 30 minutes and the metaheuristics can obtain better solutions for many instances, resulting negative deviations.

5.2.5 Comparing the ALNS with a Variable Neighborhood Search based algorithm

Variable Neighborhood Search (VNS) is also a well-known metaheuristic algorithm for tackling complex optimization problems. Different variants of VNS have been applied to various VRPs (de Freitas & Penna, 2020; Huber & Geiger, 2017; Polat, 2017; Sze et al., 2017). Specially, de Freitas and Penna (2020) proposed a VNS based heuristic algorithm. It is named as a hybrid general VNS (HGVNS) for the TSP with drones. We adapted the HGVNS to the VRPTWDR and compared the results with the ALNS in Table 9. The values in Table 9 are means of 20 instances and the column Gap reports the improvements on objective values by the ALNS compared to that of the HGVNS, which is calculated as $\text{Gap}=\frac{\text{Obj}_{\text{HGVNS}}-\text{Obj}_{\text{ALNS}}}{\text{Obj}_{\text{HGVNS}}}\times 100\%$.

As it is deliberately designed for the VRPTWDR, the ALNS outperforms the HGVNS on all Cardiff instances in terms of both objective values and computational times. Further, the Gap becomes larger when the size of the instance increases.

Table 9: Comparing the ALNS with HGVNS on Cardiff instance sets

Number of customers	HGVNS		ALNS		
	Obj.	CPU(s)	Obj.	CPU(s)	Gap (%)
15	127.04	5.93	125.70	0.33	-1.06
25	176.28	9.45	167.90	2.41	-4.76
50	258.72	27.53	243.08	10.68	-6.06
75	342.38	36.62	318.67	18.18	-6.93
100	390.66	53.89	351.99	36.55	-9.90
150	510.84	99.71	468.05	67.89	-8.38
200	595.13	139.62	527.09	98.71	-11.43

5.2.6 Improvements by the ALNS with respect to initial solutions

To ensure the efficiency of the ALNS algorithm when utilizing delivery robots in last mile, we compared the solutions to the initial feasible solution generated by the greedy insertion method discussed in Section 4.2. We run the ALNS five times on each instance with an execution time of t^{max}=5 min. Fig.3 shows the percentage improvements with regards to their initial solutions.

From the figure, we can observe that the solution obtained by greedy insertion method is improved largely by the ALNS, even up to 77.60%. As expected, the average improvement becomes larger when there are more customers, however it gets flat when the number of customers exceeds 100. The improvement with regards to the first feasible solution for small instances (with 15 and 25 customers) is under 30%. Furthermore, there are 65% instances with 15 customers which only achieve improvements less than 10%, and one instance with no improvement (the greedy insertion method generates an optimal solution).

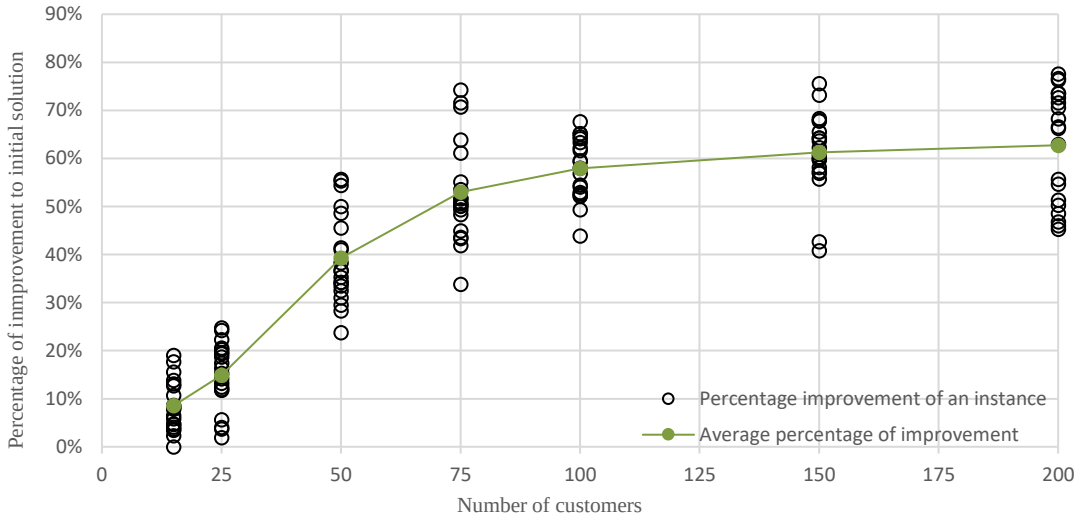


Fig.3. Percentage improvement of the best-known solution with regards to the initial feasible solution

Besides the effectiveness of the ALNS, the relatively large improvement on objective values of the VRPTWDR also attributes to the collaboration delivery of the two types of vehicles. Fig.4 presents the relationship between the increase use of robots and the percentage improvement on objective value. The value on the horizontal axis of each spot represents the difference of number of the robot-visits in the initial and the best solutions. The y-coordinate represents the percentage improvement. Generally, the increase on the number of robot-visit customers shows a clearly positive correlation with the percentage of the improvement. It also confirms the benefit brought by introducing delivery robots to the traditional last mile delivery.

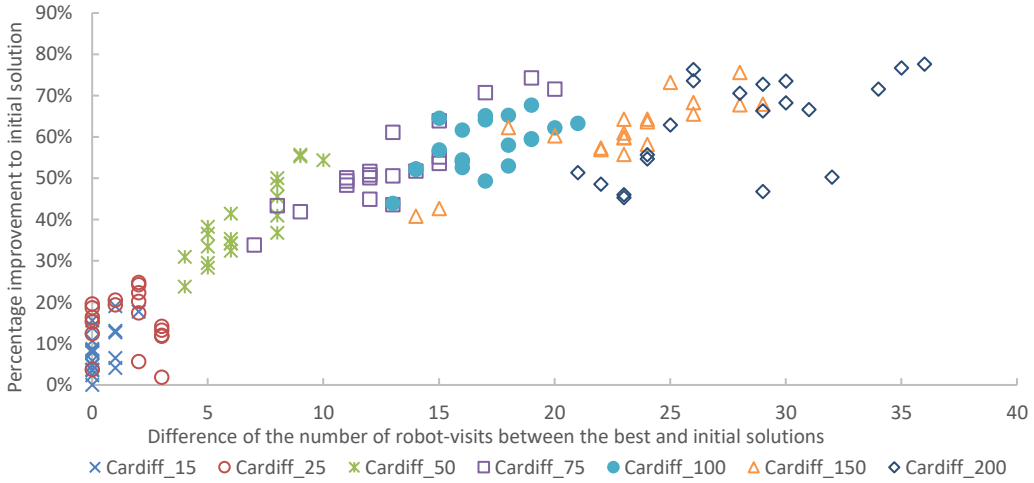


Fig.4. Deploying robot-visits results in improvement on objective value

5.3 Comparing the VRPTWDR with the MDVRP

In Poikonen and Golden (2020), the authors conducted experiments on instances with different number of drones installed into a van. However, we only took the scenario with eight assistants as example for comparison. Further, in the literature, only the average objective values were given, which we compared in Table 10. As indicated, more potential parking places result in lower objectives. In the VRPTWDR, besides the potential

parking places (V), the customer locations (C) are also available for parking the van. In this regard, for a certain combination of |V| and |C|, the actual number of available parking places is |V|+|C|. Therefore the average objective value of combinations of |V|={0,25} and |C|={25,50}, which is presented in column average in Table 10, is comparable with the values given in Poikonen and Golden (2020) (column MDVRP).

Table 10: Results under scenario with extra parking places

Assistant speed	MDVRP	VRPTWDR			
		|V|=0	|V|=25	average	|V|=50
10 m/s	9,939.76	14,069.18	6,493.58	10,281.38	6,043.64
15 m/s	6,830.86	10,955.45	4,312.38	7,633.915	4,004.08
Improvement	26.00%	22.13%	33.59%	27.86%	33.75%

The average objective value of the MDVRP is lower than that of the VRPTWDR. Although this is not a very accurate comparison, it could be concluded that allowing different launch and retrieve places improves the delivery efficiency to some extent. On the other hand, higher assistant speed amplifies the Gap as the MDVRP has a 3.32% lower value than that of the VRPTWDR at speed 10 m/s. This number goes up to 10.52% when the speed increases to 15 m/s. Moreover, when there are more available parking locations, the improvement resulting from increasing the assistant's speed is larger.

5.4 Sensitivity analyses

We now analyze the efficiency of applying the collaborative delivery by the van and robots, including circumstances (number of robot-accessible customers and time window width), and technical limitations (number of robots available and the delivery robot's speed).

5.4.1 The number of robot-accessible customers

Technological issues including customer inaccessibility and robot's capacity limitation and the attitude of customers to the newly emerging delivery mode are two main factors that determine the number of customers who are available for robot services. Sometimes, the two factors interact. For example, when a customer's demand excesses the robot's capacity, it may be split into smaller parcels and served by more than one robot if the attitude of the customer to the deliver robot is positive. And our model can easily handle this scenario through setting virtual customers with the same parameters (i.e., location, time window, service time) to share the exceeded demand.

To assess the influence of the number of robot-accessible customers, we introduced a parameter χ, defining the percentage of it to that of all customers in each instance. With the other parameters unchanged, for each instance we generated six variants with different values of χ, say 0, 0.2, 0.4, 0.6, 0.8 and 1, to define different scenarios. For example, χ=0 means no customer can be visited by a robot and the problem becomes a standard VRPTW. Further, χ=1 means each and every customer can be visited by a robot. We conducted experiments on all instances in Cardiff_15, Cardiff_25, Cardiff_50, Cardiff_100, Cardiff_150, and Cardiff_200 instance sets.

Fig.5 illustrates the distribution of the objective values, represented by the percentage to the value of the VRPTW (χ=0). While the number of robot-accessible visits increases (larger value of χ or larger instances), more savings compared to the VRPTW have been obtained. Given a slower speed than the van, the main

advantage of the robot delivery is that they can be operated as swarm logistics mode. Meanwhile, the benefit resulted from the swarm logistics is limited by the available number of delivery robots.

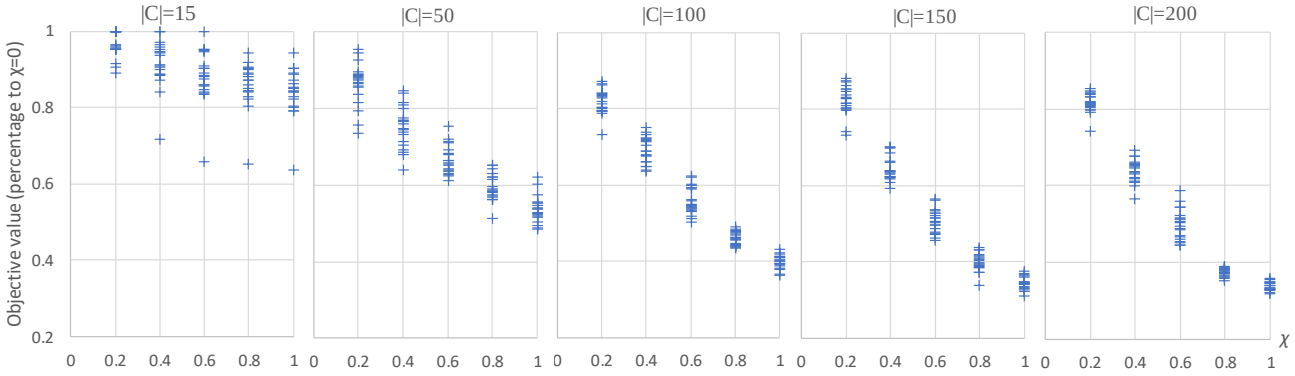


Fig.5. Distribution of objective values of different number of robot-accessible customers scenarios

Additionally, the difference between χ=1 and χ=0.8 is smaller when it is compared to that of other adjacent values of χ because of the theorical limit on the number of robot-visits. In the VRPTWDR, the robot needs to be dispatched at a customer location that can be visited by the van, hence the maximum number of customers that visited by robots is limited as $|C|-\lceil|C|/(|DR|+1)\rceil$. We sorted the number of robot-visits of solutions to different values of χ and displayed them in Fig.6. The value increases as the number of robot-visit available customers increases. Especially, more robot services are assigned in scenario with χ=1 than that with χ=0.8, although the theoretical limit number of the robot visits for them are the same. This is because the scenario with χ=1 provides higher possibility to deploy robot visits. In addition, similar to the objective values, this increase turns to be more significant on instances with larger number of customers.

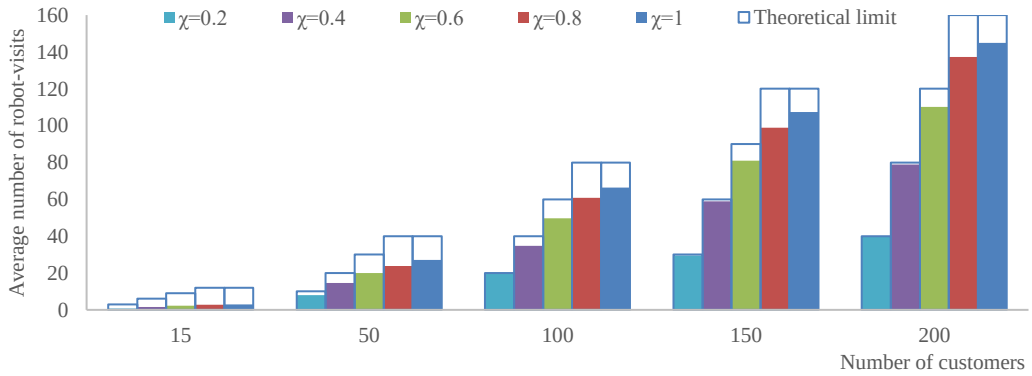


Fig.6. Average number of robot-visits in scenarios with different percentages of robot-accessible customers.

5.4.2 Analysis on the width of time window

To investigate how time window constraints influence the robot application, instances with different widths of time windows are generated, i.e., 1h, 2h, 4h, 6h and 8h. Because the total operation period for the depot is nine hours, a scenario with 8h time window constraint is almost equivalent to that without time window constraint in our experiments here. For each instance set, we randomly generated the lower of the time window,

and then the upper is determined based on the predefined width. Fig.7 illustrates both average objective value of instances with various time windows widths and average number of customers served by robots. To facilitate comparison, we normalized those values. The average objective of instances with one hour time windows and the number of robot-visits of instances with 8 h time windows are set to be one. And other values are showed using the corresponding percentages to them.

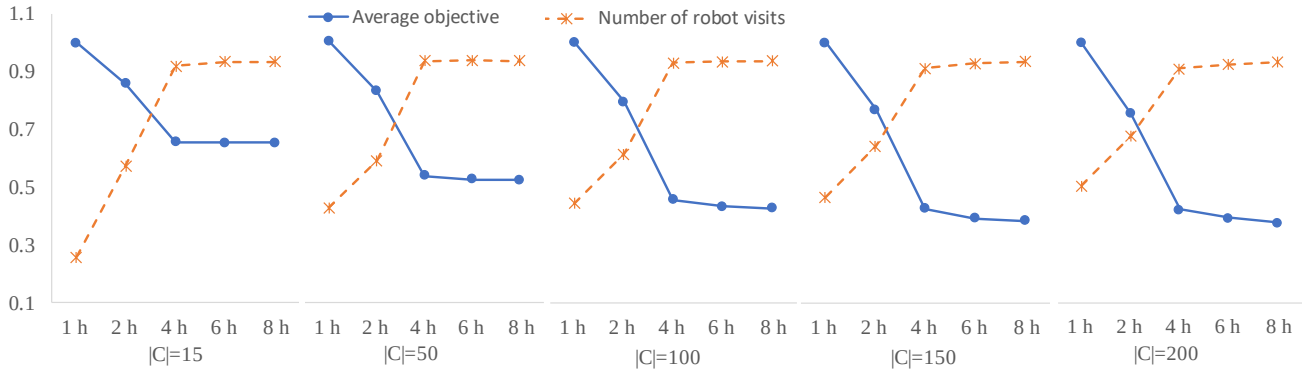


Fig.7. Solutions with different time window width.

Fig.7 highlights that the objective value is improved when widening the time window. This change is very significant at the beginning, e.g. extending from 1 hour to 2 hours, and then the curve becomes very flat after the width of time windows reaches 4 hours. Correspondingly, the smaller the width is, the less robot-visits are deployed, especially when the total number of customers in instances is small. For example, some of the instances with 15 customers deploy no robot-visit at all.

5.4.3 Analysis on the number of delivery robots

The availability of delivery robots installed in a van is another critical factor that determines how much the benefits can be gained, as the VRPTWPD benefits from the parallel deliveries implemented by robots. We conducted experiments on instances using different values for the maximum number of robots (*NR*) that are installed in a van. We set zero, one, two, four, six and eight for *NR* respectively. Obviously, the problem becomes a VRPTW when *NR*=0. So, as showed in Fig.8, we set the average objective value of each instance group with *NR*=0 as 1 and other average objective values for other values of *NR* are presented by their percentages to it.

As expected, objective values decrease when values of *NR* increase and decreasing marginal returns was observed, which is the same as the findings for vehicle-drone delivery system in Poikonen and Golden (2020). The biggest drop occurs when *NR* increases from 0 to 1, which approves the benefits brought by the application of the robot delivery. Then, the decrease on the objective value becomes smaller as *NR* grows until it reaches 0. However, the sensitivity in instances with different number of customers are different. In instances with less customers, the improvements turn to 0 at a small value of *NR*. For all instances with 15 customers, increasing *NR* from 2 to 3 does not result in a smaller objective value, while for instances with 50 customers, slight improvements are observed when increasing *NR* from 4 to 6: the average improvement of the 20 instances is 0.41% with 5 of them are zero. Although the improvement is always seen in instances with bigger number of

customers (over 100) when *NR* increases, it becomes small gradually.

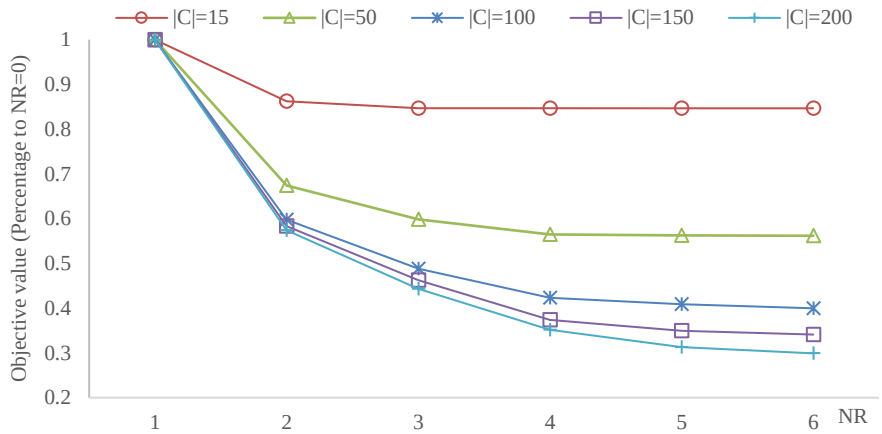


Fig.8. Average objective values of instances with different values of *NR*

5.4.4 Analysis on the robot speed

In delivery systems with delivery assistants, the assistant's speed plays an important role in determining the practical benefit brought by adopting it (Murray & Chu, 2015; Poikonen et al., 2017). Poikonen and Golden (2020) stated that the improvement is up to 26% when the drone speed is increased from 10 m/s to 15 m/s in the MDVRP in which all customers are visited by the drone. The decrease of the objective value results from the cut of the drone flying time. However, in the VRPTWDR, when robots' speed increases, the waiting time of the van for robots at customers' sites decreases necessarily, which results in not only the improvement on the objective value directly, but also more possible robot-visit operations. This is confirmed by the following experiments. Based on the robot speed (*sr*) set to 10 km/h in the main setup, we conducted experiments considering other five speeds for the robot: 3 km/h, 5 km/h, 8 km/h, 12 km/h and 15 km/h.

Fig.9 reports the average objective values of instances and the average numbers of robot-visits under different robot speeds. For all instances with different number of customers, increasing the robot speed results in improvements on objective values with more or the same number of robot-visits deployed. Decreasing marginal returns is also observed. Actually, when *sr*=3 km/h, three out of the twenty instances with 15 customers have no robot visits. Therefore, it is known that when the robot's speed is low enough compared to the van's speed, there would be no robot visit at all. We did not consider speeds lower than 3 km/h in our case study, because delivery robots can easily move faster than 1 m/s (3.6 km/h) as far as discussed in the literature. The VRPTWDR seems to be more sensitive to the assistant's speed than the MDVRP in Poikonen and Golden (2020). In the MDVRP, when the drone speed increases from ten m/s to 15 m/s, the average improvement in the objective is 2.17% per km/h, while that value is 2.21% per km/h even when the robot speed increases from 10 km/h to 12 km/h and it is 4.82% per km/h in the speed interval [3, 15] km/h.

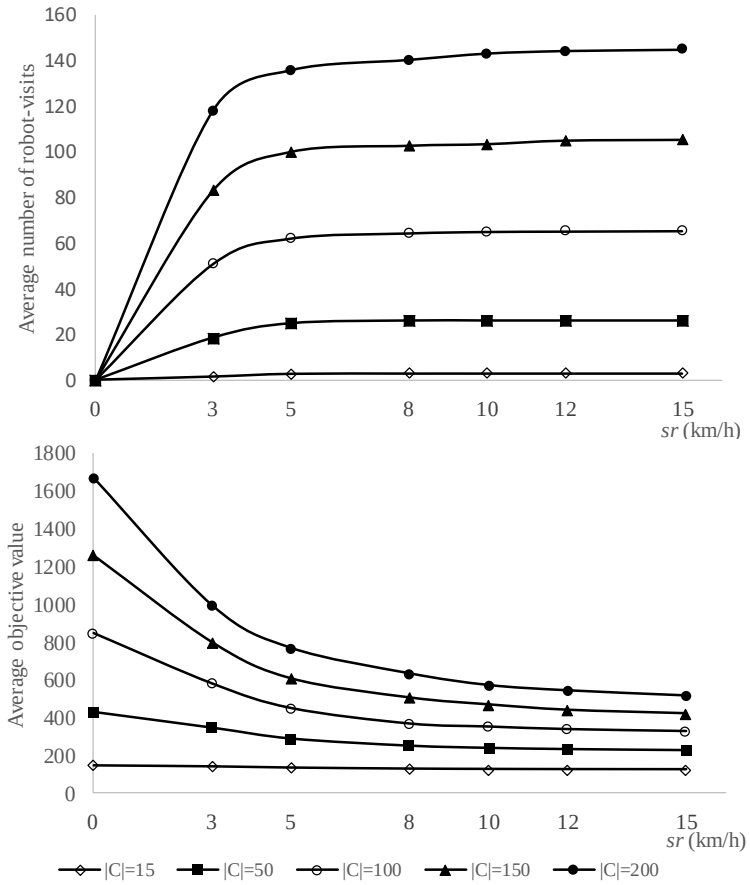


Fig.9. Results of the VRPTWDR on instances under different robots' speed

5.5 Managerial insights

The effective application of the ALNS allowed us to investigate the characteristics of the VRPTWDR on large scale instances. Our experiments demonstrated the benefits and operational limitations of incorporating robots into traditional routing problems. Based on the results we obtained in our study, we recommend LSPs to focus on the followings:

(i) In populated areas where available parking locations are limited, utilizing delivery robots as assistants in traditional delivery system is a promising alternative solution.

(ii) Allowing the assistant being dispatched and retrieved at different locations helps to improve the objective when the assistant's speed is big enough, e.g., two or three times higher than delivery van's speed.

(iii) Widening customers' time window and increasing customers' accessibility for the robot facilitate the deployment of the robot-visits and the implementation of parallel deliveries nearby.

(iv) Current technological challenges of delivery robots can limit the potential of this technology. Increase in speed and capacity can help companies to better use of this technology as assistant in last mile logistics.

6. Conclusions

We have developed an advanced metaheuristic algorithm for a new variant of the VRPTW, called VRPTWDR. In our problem settings, each delivery van is collaborating with several autonomous delivery robots.

These delivery robots travel on sidewalks to bring small- to medium-sized parcel deliveries to residents in an urban environment. By incorporating delivery robots along with the standard deliveries by a driver, this technology will improve the efficiency of deliveries and reduce the total route completion times. The introduced problem is also quite relevant nowadays since these robots can help to ensure social distancing to keep driver of vehicle and customers safe. The proposed algorithm is based on an application of the ALNS algorithm. It has performed very well and provided high quality solutions on instances up to 200-node instances.

Using the proposed ALNS, the characteristics of the problem have also been investigated and some preliminary conclusions have been derived. First, using robots for customers decrease the objective value dramatically, which indicates a remarkable advantage of adopting robots in urban delivery operations. Second, while the time window constraints reduce the possibility of parallel delivery services carried out by robots, increasing the number of robot-accessible customers improve the objective value. Third, with the further technological development of delivery robots, like higher speed, larger capacity, can help to boost the benefits brought by the utilization of delivery robots to some extent.

As using robots in delivery activities is promising, there are several extensions for related future research. For example, integrating pickup and delivery in the VRPTWDR is a practical and interesting research direction. However, it brings more constraints and increases the complexity of the problem. Moreover, there is a need for making these model more realistic by incorporating dynamic demand, the characteristic of real traffic network, and robots' battery recharging functions.

Acknowledgements

The authors would like to thank the Editor the two anonymous reviewers for provided suggestions and careful reading of the manuscript.

References

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization Approaches for the Traveling Salesman Problem with Drone. *Transportation* Science, *52*(4), 965–981.

Agatz, N., & Campbell, J. F. (2018). Preface: Special issue on Drone Delivery Systems. *Networks*, *72*(4), 409–410.

Alvarez, A., & Munari, P. (2017). An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. *Computers and Operations Research*, *83*, 1–12.

ANYbotics. (2019). *Robotic Package Delivery with ANYmal*. (Accessed on January 2021) https://www.anybotics.com/robotic-package-delivery-with-anymal/

Anymal, 2019. Copyright ©ANYbotics AG 2016. (Accessed on January 2021) https://www.anybotics.com/

Baldacci, R., Mingozzi, A., & Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, *59*(5), 1269–1283.

Baldacci, R., Ngueveu, S. U., & Calvo, R. W. (2017). The vehicle routing problem with transhipment facilities. *Transportation Science*, *51*(2), 592–606.

BBC News. (2019). *Google's Wing delivery drones head to Europe*. https://www.bbc.co.uk/news/technology-46456694

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

Campbell, J. F., Corberán, Á., Plana, I., & Sanchis, J. M. (2018). Drone arc routing problems. *Networks*, *72*(4), 543–559.

Carlsson, J. G., & Song, S. (2018). Coordinated Logistics with a Truck and a Drone. *Management Science*, *64*(9), 4052–69.

Chang, Y. S., & Lee, H. J. (2018). Optimal delivery routing with wider drone-delivery areas along a shorter truck-route. *Expert Systems with Applications*, *104*, 307–317.

Chen, C., Demir, E., Huang, Y. & Qiu R. (2020). The adoption of self-driving delivery robots in last mile logistics. *Transportation Research Part E: Logistics and Transportation Review*, 146, 102214.

Chiang, W.-C., Li, Y., Shang, J., & Urban, T. L. (2019). Impact of drone delivery on sustainability and cost: Realizing the UAV potential through vehicle routing optimization. *Applied Energy*, *242*, 1164–1175.

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, *12*(4), 568–581.

de Freitas, J. caria, & Penna, P. H. V. (2020). A variable neighborhood search for flying sidekick traveling salesman problem. *International Transactions in Operational Research*, *27*, 267–290.

Demir, E., Bektaş, T., & Laporte, G. (2012). An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. *European Journal of Operational Research*, *223*(2), 346–359.

Diaz, J. (2019). *8 robots racing to win the delivery wars of 2019*. (Accessed on January 2021) https://www.fastcompany.com/90291820/8-robots-racing-to-win-the-delivery-wars

Dorling, K., Heinrichs, J., Messier, G. G., & Magierowski, S. (2017). Vehicle Routing Problems for Drone Delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*(1), 70–85.

Es Yurek, E., & Ozmutlu, H. C. (2018). A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies*, *91*, 249–262.

Franceschetti, A., Demir, E., Honhon, D., Van Woensel, T., Laporte, G., & Stobbe, M. (2017). A metaheuristic for the time-dependent pollution-routing problem. *European Journal of Operational Research*, *259*(3), 972–991.

Francis, S. (2019). *Solving the tricky last mile: Delivery robots to rescue e-commerce*. (Accessed on January 2021)

Ham, A. M. (2018). Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and m-visit using constraint programming. *Transportation Research Part C: Emerging Technologies*, *91*, 1–14.

IBM ILOG. 2019. Copyright ©International Business Machines Corporation, 1987.

Huber, S., & Geiger, M. J. (2017). Order matters – A Variable Neighborhood Search for the Swap-Body Vehicle Routing Problem. *European Journal of Operational Research*, *263*(2), 419–445.

Hutter, M., Gehring, C., Lauber, A., Gunther, F., Bellicoso, C. D., Tsounis, V., Fankhauser, P., Diethelm, R., Bachmann, S., Bloesch, M., Kolvenbach, H., Bjelonic, M., Isler, L., & Meyer, K. (2017). ANYmal - toward legged robots for harsh environments. *Advanced Robotics*, *31*(17), 918–931.

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

Kottasova, I. (2016). *Forget drones, here come delivery robots*. (Accessed on January 2021) https://money.cnn.com/2015/11/03/technology/starship-delivery-robots/?iid=EL

Lin, S. W., Yu, V. F., & Lu, C. C. (2011). A simulated annealing heuristic for the truck and trailer routing problem with time windows. *Expert Systems with Applications*, *38*(12), 15244–15252.

Matlab R2019b. (2020) Copyright © The MathWorks Inc, USA.

McFarland, M. (2016). *This Mercedes-Benz van will carry a fleet of delivery robots*. (Accessed on January 2021) http://money.cnn.com/2016/09/07/technology/starship-robot-mercedes-benz/index.html

Milthers, N. P. M. (2009). Solving VRP using Voronoi Diagrams and Adaptive Large Neighborhood Search. *Department of Computer Science, University of Copen-hagen*, 2009.

Murray, C. C., & 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., & Smith, A. E. (2020). Foreword: COR special issue on computational operations research for drone systems. *Computers and Operations Research*, *115*(1069), 104837.

Nguyễn, T. B. T., Bektaş, T., Cherrett, T. J., McLeod, F. N., Allen, J., Bates, O., Piotrowska, M., Piecyk, M., Friday, A., & Wise, S. (2018). Optimising parcel deliveries in London using dual-mode routing. *Journal of the Operational Research Society*, *70*(6), 998–1010.

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

Pelletier S, Jabali O, Laporte G. Goods distribution with electric vehicles: Review and research perspectives. Transportation Science, 2016, 50(1): 3–22.

Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers and Operations Research*, *34*(8), 2403–2435.

Poikonen, S., & Golden, B. (2020). The Mothership and Drone Routing Problem. *Journal on Computing*, *32*(2), 249–262.

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

Polat, O. (2017). A parallel variable neighborhood search for the vehicle routing problem with divisible deliveries and pickups. *Computers & Operations Research*, *85*, 71–86.

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

Roxo™, 2019. Copyright © FedEx 1995-2020. (Accessed on January 2021) https://www.fedex.com/en-us/innovation/roxo-delivery-robot.html.

Sacramento, D., Pisinger, D., & 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.

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

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In M. Maher & J.-F. Puget (Eds.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 1520, pp. 417–431). Springer Berlin Heidelberg.

Simoni, Michele D., Erhan Kutanoglu, and Christian G. Claudel. 2020. "Optimization and Analysis of a Robot-Assisted Last Mile Delivery System." *Transportation Research Part E: Logistics and Transportation Review*.

Solomon, M. (1987). Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraint. *Operations Research*, *35*, 254–265.

Starship, 2014. Copyright ©Starship Technologies, 2014. (Accessed on January 2021) https://www.starship.co/.

Sze, J. F., Salhi, S., & Wassan, N. (2017). The cumulative capacitated vehicle routing problem with min-sum and min-max objectives: An effective hybridisation of adaptive variable neighbourhood search and large neighbourhood search. *Transportation Research Part B: Methodological*, *101*, 162–184.

Vincent, J. (2019). *FedEx unveils autonomous delivery robot*. (Accessed on January 2021) https://www.theverge.com/2019/2/27/18242834/delivery-robot-fedex-sameday-bot-autonomous-trials

Vincent, J., & Gartenberg, C. (2019). *Here's Amazon's new transforming Prime Air delivery drone*. (Accessed on January 2021) https://www.theverge.com/2019/6/5/18654044/amazon-prime-air-delivery-drone-new-design-safety-transforming-flight-videoAgatz, N., Bouman, P., & Schmidt, M. (2018). Optimization Approaches for the Traveling Salesman Problem with Drone. *Transportation Science*, *52*(4), 965–981.

Wang, Z., & Sheu, J.-B. (2019). Vehicle routing problem with drones. *Transportation Research Part B: Methodological*, *122*, 350–364.

Yu, S., Puchinger, J., & Sun, S. (2020). Two-echelon urban deliveries using autonomous vehicles. *Transportation Research Part E: Logistics and Transportation Review*, *141*.