Designing a multi-modal and variable-echelon delivery system for last-mile logistics

Christopher Bayliss[1,*], Tolga Bektaş[1], Vernon Tjon-Soei-Len[2], Remo Rohner[2]

[1]Operations and Supply Chain Management, University of Liverpool Management School, Liverpool, Chatham Street, L69 7ZH, UK
christopher.bayliss@liverpool.ac.uk, t.bektas@liverpool.ac.uk
[2]Darwin Evolution Technologies, UK
vernon@bezos.ai, remo@bezos.ai

August 31, 2022

Abstract

This paper proposes a last-mile logistics delivery system which makes use of multiple localised storage depots and multi-modal delivery options. Multiple localised storage depots facilitate express and instant delivery services. Multi-modal delivery allows for use of alternative green vehicle types for performing deliveries where there may also be vehicle access restrictions. Additionally, when demand density is sufficiently high and parcel sizes small, utilising alternative delivery modes, such as electric cargo bike and porters, can be cost effective in their own right. The proposed model allows for vehicles to rendezvous at kerbside locations (mobile satellites) where parcels can be transferred between vehicles, a feature that is shown to reduce depot stem costs. For the purpose of generality and the potential for higher quality solutions, no fixed echelon or hierarchical structure is placed on the sequence of vehicles transporting any parcel, that is, the problem is one of variable-echelon. The last-mile delivery system described in the paper gives rise to a multi-modal delivery problem using a heterogeneous fleet of vehicles and with synchronisation constraints. The paper presents a mathematical formulation of the problem and a heuristic algorithm. Computational results are presented that validate the mathematical model and the heuristic on a set of benchmark instances, some of which are based on the literature. The paper also describes a new set of benchmark instances derived from real sales data in London, whose results demonstrate potential benefits from using the proposed delivery concept.

Key words: Transportation; variable-echelon vehicle routing; mobile satellites; cargo bikes; open routes

1 Introduction

Online retail is accounting for an ever increasing portion of all retail sales. In the UK alone, the volume of the parcel market reached 2.6 billion items in the 2018-19 financial year (Ofcom, 2019), with online retail sales exhibiting roughly 1% a year increase over the last decade, accounting for 19% for all retail

*Author to whom all correspondence should be addressed (e-mail: christopher.bayliss@liverpool.ac.uk).

sales in January 2020, with a further sharp peak of 33% in May 2020 due to the Covid-19 pandemic (Office of National Statistics, 2020). Online sales channels have led to increased competition with retailers trying to increase their market share by offering lower prices and cheap fast delivery to the customer's doorstep Goldberg (2019). 20% of online retail sales consist of large white-goods, such as fridges and washing machines often requiring 2 person delivery due to weight and installation, while 80% of online sales represent smaller parcels (Global Data, 2018).

With an ever-growing strain on existing logistics and transportation networks and increasing concerns regarding pollution (noise and vehicle exhaust fumes) in densely populated urban areas, we are presented with an opportunity to redesign last-mile logistics for the good of all parties. When demand density is sufficiently high, hiring small storage spaces in areas close to points of demand can become cost effective, since it reduces depot-to-delivery-zone stem costs, which occur when inventory is stored in remote areas. Also, when demand density is sufficiently high and package sizes are small, it can become cost effective to make use of smaller delivery vehicles, such as cargo bikes and porters (walkers), which are also more environmentally friendly in comparison to larger delivery trucks. When the vehicle fleet is highly diversified and some vehicles have limited carrying capacity or range, mobile satellites (kerb side spaces) can be used for transferring parcels between delivery modes. In such a scenario, vehicle tours need to be synchronised.

In this work we address a routing problem that arises within a logistics network composed of multiple localised storage depots and a diverse fleet of vehicles responsible for delivering parcels to the customer doorstep. Our aim is to develop a model for coordinating third party couriers, for performing regular parcels deliveries. Delivery of an item may be made using a sequence of different transport modes. No fixed or hierarchical structure is imposed upon the paths of parcels from storage depot to customer doorstep; that is parcels can be transferred to and from any pair of vehicle types, instead the proposed model determines the best paths for all parcels, hence we have a variable echelon structure. This work also focuses on the case where vehicles follow open routes, where the start and end point of vehicle tours are not predetermined but are to be decided as part of the design. A similar delivery system has been trialled by Ford (2019), who developed proprietary software which *identifies optimal places for van drivers to pull over near multiple drop-off points, and where pedestrian and cycle couriers perform the last leg of the delivery*.

The contributions of this work are as follows:

- We introduce a multi-modal and a variable-echelon delivery system for last-mile delivery.
- We describe a mathematical programming formulation for the problem to obtain exact solutions.
- A scalable two-phase heuristic algorithm is proposed for instances that are not within the reach of the mathematical programming formulation. The heuristic is validated by comparison with exact and benchmark solutions.
- A series of computational experiments are presented which demonstrate the benefits of multi-modal and variable echelon delivery and yield insights into the best fleet compositions when both customer density and average parcels sizes are varied.

The remainder of this document is structured as follows. Section 2 reviews closely related literature. Section 3 provides a problem description. Section 4 provides a mathematical formulation. Section 5

presents a scalable heuristic algorithm. Section 6 presents the results to a series of experiments. Finally, Section 7 summarises the main conclusions and outlines directions for future research.

2 Literature review

In this review we firstly explore existing last-mile delivery strategies. We then review the existing vehicle routing and location-routing models most closely related to the problem considered in this work. This section concludes with a review of the most promising solution methodologies.

2.1 Last-mile delivery strategies

One direction in last-mile delivery research is the use of environmentally friendly vehicles such as electric vehicles or porters (walkers). Allen et al. (2018) analyse data from delivery routes of two different parcel carriers operating in London, with the aim of evaluating the potential benefits of utilising porters in conjunction with truck deliveries. It was found that, on average, trucks spend 60% of the time stationary, while drivers make deliveries on foot, and that handing over parcels to porters, at relatively few drop off points has the potential to reduce vehicle time by 55% while increasing vehicle utilisation. McLeod et al. (2020) also analyse the potential benefits of utilising porters and cargo bikes alongside vans in last-mile delivery. In their model vans remain responsible for the delivery of heavy and bulky items, as well as returns. The remaining parcel deliveries are assigned to porter clusters, where a routing problem is solved for each cluster. While also focusing on a London case study, they found that portering based delivery models reduced costs, emissions, driving time and the time vehicles spend stationary kerbside.

Arnold et al. (2018) develop a simulation model for comparing three alternative last-mile delivery models in the city of Antwerp, focusing on the trade-offs between costs to logistics service providers (LSPs) and negative externalities such as congestion and tail pipe emissions. In comparison to the traditional diesel powered truck delivery model, encouraging customers to opt to pick up parcels from pick up points reduces LSP costs at the expense of increased congestion due to customers travelling to pickup points (Liu et al. (2019), Kedia et al. (2020) and Lin et al. (2020) each consider the impacts of the locations of collection delivery points (CDPs) on externalities in more detail). On the other hand, making use of cargo bikes in conjunction with trucks for deliveries reduces negative externalities but increases LSP costs.

Regarding the impact of the operating environment on the structure of network architecture and delivery model Janjevic and Winkenbach (2020) provide a review covering both mature and emerging markets. Janjevic and Winkenbach (2020) enumerate the variables which categorise different urban last-mile delivery strategies. de Mello Bandeira et al. (2019) perform a systematic literature review to identify the most promising vehicle types for improving last-mile delivery operations in urban areas. They identified electric light delivery vehicles (eLDVs) and electric tricycles as strong candidates for improving postal delivery operations in Rio de Janeiro across three "bottom-line" criteria: economic, environmental and social.

Parcel lockers provide an alternative delivery option for customers and offer several benefits. For example, customers may not be at home at the delivery time, so delivery failure can be avoided. Also, multiple customer orders can be fulfilled with a single visit to a parcel locker station. Ballare and Lin (2020) investigates combining parcel locker networks and crowd shipping as a last-mile delivery

approach, a combination not yet field tested. The concept is that crowd shippers pick up packages, take them to the parcel locker and sort them by zip code. Intra-zonal parcels are delivered within the same service area using crowd shippers, while inter-zonal packages which are delivered to the target service area using traditional trucks.

Another concept gaining ground in last mile delivery is that of horizontal cooperation between different last-mile carriers, whose depots or service areas overlap. Allen et al. (2017) introduce the concept of a *Freight Traffic Controller* (FTC) who is a trusted third party who decides how carriers can cooperate, in such a way that all carriers benefit proportionally to their contribution to a coalition. The approach can also lead to environmental benefits. Our approach can be used to coordinate a range of third-party couriers.

For a comprehensive review of last-mile delivery concepts see Boysen et al. (2021), who also provide a compact notation for describing last-mile delivery concepts. A delivery concept is described by all the possible sequences of storage locations, modes of transport and final handover method. In the delivery concept proposed in this paper, the number of sequences is not explicitly bounded, described here as multi-modal and variable-echelon.

2.2 Single and multi-echelon vehicle routing

Vidal et al. (2020) provide a recent review of the existing variants of the vehicle routing problem (VRP), and focus on three main perspectives: i) objective functions, including costs, reliability and environmental considerations; ii) integration with other combinatorial optimisation problems; and iii) increasing model fidelity. They highlight routing integrated with districting, facility location, fleet composition and inventory management. Dündar et al. (2021) provide a review on recent research on sustainable urban routing, and focus on work which addresses: i) economic dimensions; ii) economic and environmental dimensions; iii) economic and social dimensions; and iv) economic, environmental and social dimensions.

Examples of single-echelon last-mile delivery routing problems include Martinez-Sykora et al. (2020) and Bayliss et al. (2020a). Martinez-Sykora et al. (2020) develop an optimisation model for last-mile delivery in London using a combination of walking and driving. Performing sequences of deliveries on foot, within a vehicle route, can be helpful in densely populated areas where kerbside parking space is hard to find. When retail store deliveries and online customer parcel deliveries are integrated within the same routes, thereby increasing vehicle utilisation, we get the omni-channel vehicle routing problem (Bayliss et al., 2020a). The model proposed here allows for the use of vans and porters, as well as the use of intermediate storage facilities within vehicle routes.

Vehicle routing problems are increasingly defined in terms of the number of echelons they consist of. In general, multi-echelon models (Quetschlich et al., 2021) represent attempts to integrate usually disparate planning problems, in order to gain efficiencies through coordination between adjacent echelons. In the current context, a single-echelon is one phase in a possibly multi-phased delivery process and is characterised by a starting depot, where inventory is loaded into a delivery vehicle. The subsequent delivery locations may be final customers or intermediate facilities which are the start points for deliveries made in a subsequent echelon. Crainic et al. (2009) first proposed a two-echelon city logistics system, in which large polluting vehicles, are excluded from city centres, and deliver cargo to satellite depots located on the city outskirts from which smaller "green" vehicles are employed to make deliveries

within the densely populated urban areas.

Zhang et al. (2018) presents a simulation study comparing standard parcel delivery, a two-echelon cargo bike distribution system and a pick-up point based delivery system. Based on a Wilmersdorf-Berlin case study, they find that compared to a typical parcel vehicle approach, utilising cargo bikes to perform deliveries starting from micro depots reduces the required number of parcel vehicles, increases the number of items delivered per vehicle tour, while reducing vehicle miles and vehicle operating time. Utilising parcel lockers reduces the number of vehicle tours, vehicle miles and time, while increasing the number of items delivered per vehicle tour. See Cuda et al. (2015) and Sluijk et al. (2022) for surveys on two-echelon routing problems.

In contrast to existing works, which propose two-echelon delivery systems, our model diverges from this approach. Instead, our model focusses on the case where all vehicle types can possibly be acceptable for delivering parcels within urban areas and transferring parcels to different vehicles. Our model allows multi-modal delivery routes via the use of mobile satellites and micro-hubs as locations for parcels transfers between vehicles, and places no-hierarchical constraints on the flow directions of parcels between different vehicle types. Instead, it allows the optimisation process to discover the best delivery structure, based on the fundamental characteristics of the vehicle fleet and physical characteristics of the problem instance. The resulting routing solutions may then exhibit, for example, a mix of single, two and three-echelon routes.

2.3 Multi-echelon location-routing

When the start points of next-echelon routes are not predetermined, we have a multi-echelon location-routing problem. In the current problem, the locations of any vehicle rendezvous are decision variables. The vast majority of existing multi-echelon location-routing models are strict two-echelon models, e.g. (Cao et al., 2021; Yu et al., 2020; Pichka et al., 2018). Specifically, Cao et al. (2021) consider the two-echelon location-routing problem for a biomass logistics system, where biomass must be collected and processed at intermediate facilities, before being transported to the main biomass refinery. Yu et al. (2020) considers the two-echelon open location-routing problem for urban delivery. The first echelon consists of trunk routes from the main depot to loading-unloading zones, like mobile satellites. In the second echelon, delivery routes are not required to return to the start point, and are therefore *open routes*. Similarly, Pichka et al. (2018) consider a two-echelon open location-routing problem in which deliveries in both echelons utilise third party logistics providers.

Other authors consider the location-routing problem from a strategic perspective and focus on deciding where facilities should be located, while only approximating route costs, since this vastly increases the sizes of problem that can be solved. Janjevic et al. (2021) consider a three echelon last mile delivery network, where echelons represent different delivery speeds, instant, express and same day.

In the cases of Yu et al. (2020) and Pichka et al. (2018) first and second echelon tasks are distinct and separate. Anderluh et al. (2021) consider a case where there is some overlap. Anderluh et al. (2021) consider a two-echelon vehicle routing problem with vehicle synchronisation. Customers are assigned to one of three areas in a city: i) outer city, where deliveries are made by first echelon vehicles; ii) inner city, where deliveries are made by second echelon delivery vehicles; or iii) grey zone, where deliveries can be made by either first or second echelon delivery vehicles. The grey zone also contains satellite locations for parcel transhipments from first echelon vehicles to second echelon vehicles. While

Anderluh et al. (2021) represents the closest existing model to that proposed here, Anderluh et al. (2021) still focus on two-echelons of vehicle routing problems, while also considering multiple objectives, namely minimisation of cost, greenhouse gas emissions and noise pollution.

When parcel transhipments between vehicle's are allowed enroute, vehicle routes must be synchronised. Drexl (2012) provides a classification and review of VRPs with multiple synchronisation constraints. As an archetypal example, the VRP with trailers and transhipments (VRPTT) is used to define all types of synchronisation. The main types of synchronisation are: i) task synchronisation (which vehicle performs each task); ii) operation synchronisation (when both the task and support vehicles are required to be present for transhipment operations); iii) Movement synchronisation (the availability of a cab to tow a trailer); iv) load synchronisation (the requirement that transhipment operations are required to be capacity feasible for both vehicles); and v) resource synchronisation (when vehicles use the same resources).

2.4 Solution methodologies

The complexity of the city-logistics delivery systems considered in the previous sections, and the scale of real world problem instances, necessitates heuristic solution methodologies, which sacrifice guarantees of optimality in return for faster solution times. While some authors propose genetic algorithms, such as Zhou et al. (2018), who consider a multi-depot two-echelon vehicle routing problem with customer delivery options, the vast majority turn to local search based solution methodologies (Dündar et al., 2021). Cao et al. (2021) (Section 2.3) develop a hybrid heuristic combining variable neighbourhood search (VNS) and tabu search (TS). To validate their approach, experimental results show that the heuristic performs well compared to an exact solver in small instances. McLeod et al. (2020) (Section 2.1) employ a 2-opt local neighbourhood based local search approach. While local search provides a search intensification mechanism, an extension of local search is large neighbourhood search, which introduce destroy and repair operators, with the aim of providing a diversification mechanism into the search. Enthoven et al. (2020), Hemmelmayr et al. (2012), Grangier et al. (2016), Anderluh et al. (2021) and Li et al. (2020) each develop large neighbourhood search approaches for two-echelon vehicle routing problems.

Similar to the body of literature reviewed above, this paper also proposes a heuristic solution methodology that includes local search. Both 2-opt and cut-and-insert search neighbourhoods are used in this work, applied within a VNS framework (See Section 5.2), particularly as Sluijk et al. (2022) point out that 2-opt has been successfully applied in 15 out of 23 recent 2E-VRP heuristics, while the cut-and-insert neighbourhood has been successfully applied in 19 out 23 recent 2E-VRP heuristics. Yu et al. (2020) and Pichka et al. (2018) (Section 2.3) both develop simulated annealing algorithms, which provide a mechanism to escape local optima, in which worse neighbouring solutions have a probability of being accepted as the new current solution. In this work, we use a biased-randomised (Bayliss et al., 2020b) constructive heuristic, which provides the initial solution for the local search phase, and as such embeds diversification in the construction phase of the algorithm.

3 Problem description

Demand is generated by customers who each require delivery of a set of parcels of arbitrary sizes and weights which may originate at different storage depots. A fleet of vehicles consisting of vans, cargo

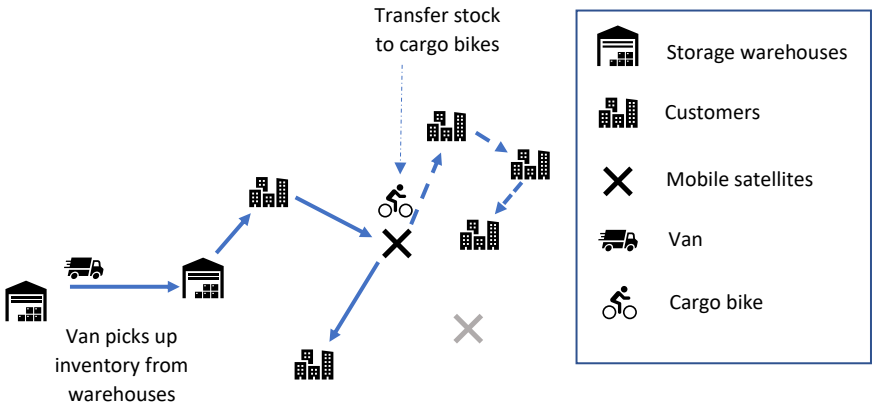

Figure 1: Example of a multi-modal delivery plan making use of mobile satellite location for a vehicle rendezvous.

bikes and porters (walkers) are available for making deliveries. Each vehicle type has their own maximum capacity (volume), maximum weight, drop times, maximum tour length, maximum tour duration and travel speed. Although we focus on three vehicle types in this paper, the model extends to several vehicle types in a straightforward way. Tours can begin at any depot or kerbside handover point (mobile satellite) with no storage capacity of its own, where parcels can be transferred between vehicles that are simultaneously present. Tours can end anywhere, i.e., open-routes, which reflects the use of third-party couriers. Returns can be modelled by treating return customers as small depots and the corresponding return depot as a customer.

We make the following assumptions on the operational aspects of the problem. First, parcel deliveries to customers must be made within hard time windows specific to and as specified by each customer, and cannot be split. Deliveries can be made either: (i) directly, from a depot to a customer using one mode of transport or (ii) using multi-modal transport, where parcels can be transferred from one vehicle to another en route. Like the use of mobile satellite locations as rendezvous locations, parcels can be transferred from one vehicle to another by dropping it off at a depot, where it can be picked up at a later time by another vehicle, this latter type of vehicle rendezvous has the advantage that both vehicles need not be present at exactly the same time. Conversely, mobile satellite type vehicle rendezvous' have the advantage that there are a greater number of potential locations where the vehicle rendezvous can take place. Both types of rendezvous introduce additional vehicle synchronisation constraints into the vehicle routing problem. No constraint is placed on the number of vehicles that can simultaneously transfer parcels. Figure 1 illustrates an example delivery plan which makes use of a mobile satellite location where parcels are transferred from a van to a cargo bike.

Routes incur travel costs dependent upon tour length (fuel) and duration (wages), and each vehicle used invokes a fixed cost. The objective is to plan a set of van, cargo bike and porter routes which are feasible and transport all parcels to the required locations, at a minimum total cost.

Figure 2 illustrates a step-by-step solution to the considered problem on an example instance where there are 5 customers, each demanding a parcel, all of the same size and weight. There are three vehicles available for performing the deliveries. Vehicle 1 has a maximum range of 3 units of distance, and space for five parcels. Vehicle 2 has a maximum range of 1.5 and space for 1 parcel. Vehicle 3 has a maximum range of 0.25 units of distance and space for 1 parcel.

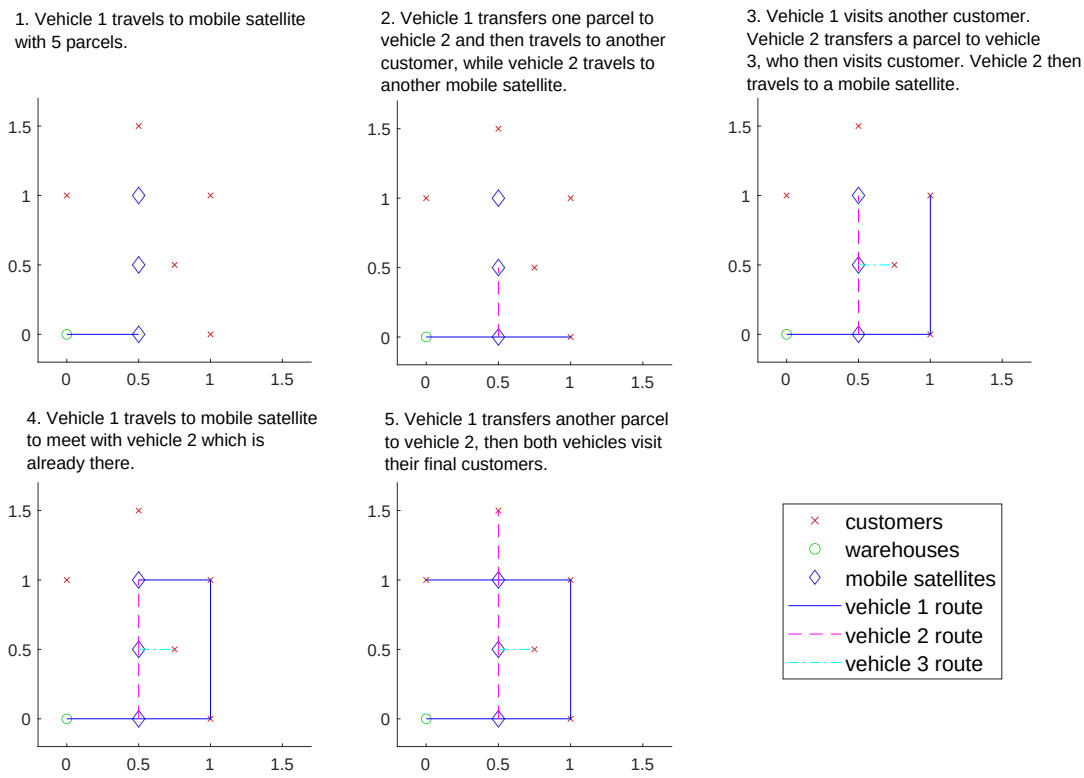

Figure 2: Example solution involving vehicle synchronisation and three vehicle types.

The optimal solution shown in Figure 2 requires vehicle 1 to pick up all five parcels from the storage depot indicated by the circle. During vehicle 1's tour it must visit three customers and meet with vehicle 2 twice, which visits one customer and transfers a parcel to vehicle 3, who then makes one delivery. The example demonstrates a scenario where the model accounts for synchronisation, which is required for vehicle rendezvous at mobile satellites.

4 Mathematical formulation

The routing problem is defined on a graph $G' = (N', A')$ where N' is the set including the set C of nodes that correspond to customer locations and set D of depots from where parcels are delivered. The graph has a set A' of arcs along which vehicles can travel between pairs of nodes as dictated by the real topology of the network. By defining a set T of time intervals over which the problem is to be solved, we convert G' into a time-space network $G = (N, A)$, where N is the set of nodes formed by replicating each node in set N' for each time slot in set T. We then define the new arc set A by replicating each arc $(i, j) \in A'$ such that $(v, t, i, j) \in A$ represents a vehicle, $v \in V$, setting off from a node, $i \in N$, towards a another node, $j \in N$, at a particular time, $t \in T$. The set of arcs A^{vt}, denotes the subset of all arcs A that vehicle v can traverse at time t, which allows for the possibility that not all arcs are available to all vehicles at all times, for instance, some arcs may not be available to large trucks due to vehicle restrictions. The full notation used for the model is presented in Table 1. We now present the model for the problem.

Table 1: Summary of the notation

		Sets
D	:	The set of depots.
K	:	The set product types.
C	:	The set of customers.
M	:	The set of available mobile satellites.
V	:	The set of vans.
B	:	The set of cargo bikes.
P	:	The set of porters.
R	:	The set of customer nodes with return requests.
T	:	The set of time intervals in the space time graph formulation.
O^i	:	The set of vehicle types that can visit customer i.
N_v	:	The set of nodes that vehicle v can visit.
N_v^{j*}	:	The set of nodes vehicle can visit immediately after node j.
N_v^{*j}	:	The set of nodes vehicle can visit immediately before node j.
A	:	The set of arcs.
A^{vt}	:	The set of arcs which begin at time t of vehicle v.
		Parameters
τ_{ijv}	:	The travel time between node i and j for vehicle v.
c_{ijv}	:	The travel cost for traversing the arc between node i and j for vehicle v.
d_{ij}	:	The distance between node i and j.
θ_v	:	Cost per unit time of vehicle v.
σ_v	:	Cost per unit distance of vehicle v.
δ	:	The time required by a vehicle to make a delivery or transfer parcels to another vehicle, referred to as *drop time*.
e_v	:	Dummy end node for vehicle v, which a distance of 0 from all nodes.
L_v	:	The maximum feasible duration of any route for vehicle v.
R_v	:	The maximum range (or mileage) of vehicle v.
a_v	:	The maximum volume that can be carried by vehicle v at any one time.
ϕ_v	:	The maximum weight carried by vehicle v.
H_v	:	Fixed cost of vehicle v.
q_{ik}	:	Stock of product type k available at depot i.
h_i	:	Capacity of depot i.
f_{ik}	:	Demand of node i for product type k.
n_k	:	Volume of storage space consumed by one unit of product type k.
m_k	:	Weight of one unit of product type k.
α_i	:	Start time of customer i's delivery time window.
β_i	:	End time of customer i's delivery time window.
ε	:	Small valued constant weight for symmetrical solution suppressing terms in the objective function.
		Decision variables
x_{vtij}	:	Binary variable which takes a value of 1 if the edge joining nodes i to j is traversed beginning at time t in the route of vehicle v
γ_{vit}	:	Binary variable which takes a value of 1 if vehicle v starts their tour from node i at time t.
z_v	:	Binary variable which takes a value of 1 if vehicle v is assigned to a route.
u_{vtik}	:	Continuous variable indicating the amount of product type k picked up by vehicle v at time t from node $i \in N \setminus C$.
w_{vtik}	:	Continuous variable indicating the amount of product type k dropped off by vehicle v at time t at node i.

$$\text{minimise} \sum_{v\in V\cup B\cup P}\sum_{t\in T}\sum_{(i,j)\in A^{vt}} x_{vtij}c_{ijv} + \sum_{v\in V\cup B\cup P} H_v z_v + \varepsilon\left(\sum_{v\in V\cup B\cup P}\sum_{i\in N_v}\sum_{t\in T} t\gamma_{vit} + \sum_{v\in V\cup B\cup P}\sum_{t\in T}\sum_{i\in N_v}\sum_{k\in K}(u_{vtik}+w_{vtik})\right), \tag{1}$$

$$\gamma_{vjt} + \sum_{i\in N_v^{*j}|\left(t-\tau_{ijv}\right)\geq 0} x_{v\left(t-\tau_{ijv}\right)ij} - \sum_{l\in N_v^{j*}} x_{vtjl} = 0,\ \forall v\in V\cup B\cup P,\ \forall t\in T,\ \forall j\in D\cup M\setminus\{e_v\}, \tag{2}$$

$$\sum_{i\in N_v^{*j}|\left(t-\tau_{ijv}\right)\geq 0} x_{v\left(t-\tau_{ijv}\right)ij} - \sum_{l\in N_v^{j*}} x_{vtjl} = 0,\ \forall v\in V\cup B\cup P,\ \forall t\in T,\ \forall j\in N_v\setminus\{e_v\cup D\cup M\}, \tag{3}$$

$$\sum_{t\in T}\sum_{i\in N_v}\gamma_{vit} \leq z_v,\ \forall v\in V\cup B\cup P, \tag{4}$$

$$\sum_{t\in T}\sum_{(i,j)\in A^{vt}} x_{vtij} \leq Gz_v,\ \forall v\in V\cup B\cup P, \tag{5}$$

$$\sum_{t\in T}\sum_{i\in N_v}\gamma_{vit} \leq 1 - \sum_{(i,j)\in A^{v0}} x_{v0ij},\ \forall v\in V\cup B\cup P, \tag{6}$$

$$\sum_{t\in T}\sum_{i\in N_v\setminus\{e_v\}} x_{vtie_v} = z_v,\ \forall v\in V\cup B\cup P, \tag{7}$$

$$\sum_{v\in O^j}\sum_{t\in T|\alpha_j\leq t\leq\beta_j}\sum_{i\in N_v^{j*}\setminus\{j\}} x_{vtji} = 1,\ \forall j\in C, \tag{8}$$

$$\sum_{v\in O^j}\sum_{i\in N_j^{*j}}\sum_{t\in T|\alpha_j\leq t+\tau_{ij}\leq\beta_j} x_{vtij} = 1,\ \forall j\in C, \tag{9}$$

$$\sum_{t\in T}\sum_{(i,j)\in A^{vt}} \tau_{ijv}x_{vtij} \leq L_v,\ \forall v\in V\cup B\cup P, \tag{10}$$

$$\sum_{t\in T}\sum_{(i,j)\in A^{vt}} d_{ij}x_{vtij} \leq R_v,\ \forall v\in V\cup B\cup P, \tag{11}$$

$$u_{vtik} \leq H\sum_{j\in N_v^{i*}\setminus\{e_v\}} x_{vtij},\ \forall t\in T,\ \forall v\in V\cup B\cup P,\ \forall i\in N_v\setminus\{e_v\},\ \forall k\in K, \tag{12}$$

$$w_{vtik} \leq H\sum_{j\in N_v^{i*}} x_{vtij},\ \forall t\in T,\ \forall v\in V\cup B\cup P,\ \forall i\in N_v\setminus\{e_v\},\ \forall k\in K, \tag{13}$$

$$\sum_{i\in N_v}\sum_{l=1}^{t}(u_{vlik}-w_{vlik}) \geq 0,\ \forall v\in V\cup B\cup P,\ \forall t\in T,\ \forall k\in K, \tag{14}$$

$$\sum_{k\in K}\sum_{i\in N_v}\sum_{l=1}^{t} n_k\left(u_{vlik}-w_{vlik}\right) \leq a_v,\ \forall v\in V\cup B\cup P,\ \forall t\in T, \tag{15}$$

$$\sum_{k\in K}\sum_{i\in N_v}\sum_{l=1}^{t} m_k\left(u_{vlik}-w_{vlik}\right) \leq \phi_v,\ \forall v\in V\cup B\cup P,\ \forall t\in T, \tag{16}$$

$$\sum_{t\in T}\sum_{i\in N_v}(u_{vtik}-w_{vtik}) = 0,\ \forall v\in V\cup B\cup P,\ \forall k\in K, \tag{17}$$

$$\sum_{v\in V\cup B\cup P} u_{vtik} = \sum_{v\in V\cup B} w_{vtik},\ \forall i\in M,\ \forall k\in K,\ \forall t\in T, \tag{18}$$

$$\sum_{v\in V\cup B\cup P} u_{vtik} \leq \sum_{l=1}^{t-1}\sum_{v\in V\cup B\cup P} w_{vlik} - \sum_{l=1}^{t-1}\sum_{v\in V\cup B\cup P} u_{vlik} + q_{ik},\ \forall t\in T,\ \forall i\in D,\ \forall k\in K, \tag{19}$$

$$\sum_{v \in V \cup B \cup P} \sum_{k \in K} n_k \left(w_{vtik} - u_{vtik} \right) \leq h_i - \sum_{k \in K} n_k q_{ik}, \ \forall i \in D, \ \forall t \in T, \tag{20}$$

$$\sum_{v \in O^i} \sum_{t \in T} w_{vtik} = f_{ik}, \ \forall i \in C, \ \forall k \in K, \tag{21}$$

$$x_{vtij} \in \{0,1\}, \ \forall v \in V \cup B \cup P, \ \forall t \in T, \ \forall (i,j) \in A^{vt}, \tag{22}$$

$$z_v \in \{0,1\}, \ \forall v \in V \cup B \cup P, \tag{23}$$

$$\gamma_{vit} \in \{0,1\}, \ \forall v \in V \cup B \cup P, \ \forall i \in D \cup M, \ \forall t \in T, \tag{24}$$

$$u_{vtik}, w_{vtik} \in \mathbb{R}^+, \ \forall v \in V \cup B \cup P, \ \forall t \in T, \ \forall i \in D \cup M \cup C, \ \forall k \in K. \tag{25}$$

The objective (1) is to minimise the solution cost, which is a sum of travel costs and vehicle fixed costs. Travel costs are assumed to be proportional to travel time (driver wages) and distance (fuel costs). The cost of vehicle v traversing arc (ij) is expressed as $c_{ijv} = \sigma_v d_{ij} + \theta_v \tau_{ijv}$. The ε weighted term of the objective function plays a similar role to a symmetry breaking constraint and is designed to reduce solution times. The value of ε is set as small as necessary to avoid changing the optimal solution from that when $\varepsilon = 0$. This term suppresses the symmetrical solutions characterised by cases where $u_{vtik} > 0$ and $w_{vtik} > 0$, i.e., where the same product type is simultaneously dropped off and picked up by a vehicle at a given time and location. This term also suppresses symmetrical solutions where vehicles can feasibly wait, unnecessarily, at any time during their tour. That is, all required travelling is performed at the earliest opportunity. The value of ε needs to account for the upper bound of the terms in parentheses and the smallest non-zero difference in solution costs between two feasible solutions. If the value of epsilon is set too high, the model may characterise a higher cost solution, compared to the solution where $\varepsilon = 0$, one which involves fewer vehicles starting at a time $t > 0$ (due to the first term in parentheses), and involving fewer vehicle rendezvous (due to the second term in parentheses).

Constraint (2) specifies that non-depot nodes visited by vehicle v are immediately exited. This constraint is convenient to report first because arc variables not included in consistent instances of this constraint are ruled out from inclusion in any of the subsequent constraints. Consistent instances of this constraint have at least one arc variable in two of the three terms. One example of an inconsistent instance of this constraint occurs for vans when $t = 0$ when it is not possible for a vehicle to enter and then exit a customer node because vehicle routes cannot leave a previous node before time $t = 0$. In general, if no arc leads to node j at time t then no arc can leave node j at time t. Constraint (2) allows for vehicles who start their tours at $t > 0$. Constraint (2) introduces a binary variable γ_{vjt} to indicate that vehicle v starts their tour at node j at time t. Constraint (2) forces γ_{vjt} to take values of 0 or 1, (note that γ_{vjt} can be set as continuous variables when the model is solved using a linear programming based solver, because this constraint forces it to a value of 0 or 1 due to the other binary variables in the constraint). Arc traversal times τ_{ijv} include any service time required before leaving node i. Constraint (3) plays the same role as Constraint (2) for nodes that can be visited by vehicle v, not including depots or mobile satellites. Constraint (4) specifies that a vehicle can start a tour at most once. Constraint (5) specifies that a vehicle is used if it traverses at least one arc, which invokes a fixed cost for using

that vehicle in the objective function. In this constraint, G is a sufficiently large number at least as great as the largest number of arcs in a single vehicle's tour, otherwise the model is not valid in all situations. Constraint (6) specifies that if a vehicle starts their tour at time zero, the γ_{vjt} variables for that vehicle are all zero. Without this constraint a vehicle can start another (infeasible) route after entering the dummy end node at the end of its first tour. Constraint (7) specifies that vehicles, if used, must enter their dummy end node once, at no travel cost ($t_{ieb} = 0,\ \forall i \in N_b$), thus terminating the vehicle's open tour. Constraint (8) specifies that customer nodes are exited exactly once by one vehicle. O^j denotes the set of allowable vehicles for visiting customer j, this approach accounts for customers who reside in locations with vehicle restrictions. It can also model cases where customers request delivery by a specific vehicle type. N_v^{j*} denotes the set of nodes that vehicle v can visit after visiting node j. This constraint allows vehicles to wait at a customer node, which may be required for satisfying customer delivery time windows. Constraints 7 and 8 ensure that each vehicle is assigned to most one tour. Constraint (9) specifies that each customer must be visited once within their delivery time window. This constraint is required to prevents deliveries being made at unacceptable times of the day, since tours starting at vehicle rendezvous can begin at any time. Constraint (9) can also be used to model narrow customer specific time windows. Constraint (10) specifies that no vehicle tours exceed the maximum tour duration. Constraint (10) requires that arc traversal variables (x_{vtij}) are 0 after the ends of routes. Constraint (11) specifies that no vehicle routes exceed the maximum range of the vehicle. Constraint (12) specifies that a vehicle cannot pick up stock from a node it does not visit, where H is a sufficiently large number, in this case, at least greater than the sum of all demands over all nodes. Constraint (13) specifies that a vehicle cannot drop off stock to a node it does not visit. Constraints (12) and (13) require that vehicles are present simultaneously at mobile satellite rendezvous, thereby enforcing synchronisation. Constraint (14) specifies that the amount of stock carried a vehicle is never negative. Constraint (15) specifies that the volume of stock carried by a vehicle never exceeds its capacity. This constraint also ensures that the total amount of stock delivered never exceeds the total amount of stock picked up at any stage of any route. Constraint (16) specifies that the weight of stock carried by a vehicle never exceeds its maximum carrying capacity. Constraint (17) specifies that all vehicles end their tours empty. Constraint (18) specifies that the amount of stock picked up by vehicles at a mobile satellite equals the amount of stock dropped off by other vehicles that visit that mobile satellite. This means that the model covers one-to-one, many-to-one and one-to-many types of vehicle rendezvous. Constraint (19) specifies that the total amount of stock picked up from a depot can at no time exceed the amount of stock stored at that depot plus the amounts of stock dropped off at that depot previously, minus the amount of stock picked up from that depot previously. This constraint models the use of depots as locations where stock can be delivered to a depot, stored temporarily, and then picked up by a different vehicle, all in the same day. Constraint (20) specifies that the amount of inventory at a depot never exceeds the depot's capacity. Constraint (21) specifies that the amount of stock delivered to customers equal their demands. Constraints (22), (23), (24) and (25) impose domain restrictions on the variables concerning arc traversal, vehicle use, route start time-location and parcel transfer amounts respectively.

4.1 Valid inequalities

Constraints (26), (27) and (28) are symmetry breaking constraints, for vans, cargo bikes and porters respectively, and ensure that vehicles are used in ascending numerical order. Symmetry breaking con-

straints are designed to help reduce the time required by MIP solvers to prove the optimality of solutions.

$$z_v \geq z_{v+1},\ \forall v \in \{1,2,\ldots,|V|-1\}, \tag{26}$$

$$z_b \geq z_{b+1},\ \forall b \in \{1,2,\ldots,|B|-1\}, \tag{27}$$

$$z_p \geq z_{p+1},\ \forall p \in \{1,2,\ldots,|P|-1\}. \tag{28}$$

Constraint (29) bounds, above and below, the numbers of units of each product type dropped off and picked up at depots and mobile satellites. They are non-negative and less than the maximum number of units of each product type that fit in a vehicle according to the maximum capacity and weight of the vehicle. These constraints are similar to those that were found to improve lower bounds calculated for capacitated vehicle routing problems in Letchford and Salazar-González (2015). Constraint (30) specifies that the amount of each product type delivered to a customer never exceeds the amount they have demanded.

$$0 \leq w_{vtik},\ u_{vtik} \leq \min\left(\frac{a_v}{n_k},\ \frac{\phi_v}{m_k}\right),\ \forall v \in V \cup B \cup P,\ \forall i \in D \cup M,\ \forall k \in K,\ \forall t \in T, \tag{29}$$

$$0 \leq w_{vtik} \leq f_{ik},\ \forall v \in V \cup B \cup P,\ \forall i \in C,\ \forall k \in K,\ \forall t \in T. \tag{30}$$

Constraints (31) and (32) are strengthened version of constraints (15) and (16), since the right hand side is multiplied by the vehicle use variable z_v. In Section 6.3.1 we examine the impact that the valid inequalities presented in this section have on solution times.

$$\sum_{k \in K} \sum_{i \in N_v} \sum_{l=1}^{t} n_k \left(u_{vlik} - w_{vlik}\right) \leq a_v z_v,\ \forall v \in V \cup B \cup P,\ \forall t \in T, \tag{31}$$

$$\sum_{k \in K} \sum_{i \in N_v} \sum_{l=1}^{t} m_k \left(u_{vlik} - w_{vlik}\right) \leq \phi_v z_v,\ \forall v \in V \cup B \cup P,\ \forall t \in T. \tag{32}$$

5 Heuristic solution approach

In this section a heuristic solution methodology is presented which is composed of two consecutive phases, a constructive phase that uses biased randomisation, followed by an improvement phase using local search. While time remains, this procedure is repeated and the algorithm generates more solutions, always keeping track of the lowest cost solution generated overall. This schema is outlined in Algorithm 1.

5.1 Construction phase

The constructive heuristic is based on building a solution, one step at a time, where in each step, we find a way modify the current solution such that another customer's delivery is fulfilled in the routing solution, and continues until all customer deliveries are fulfilled. The ways in which another delivery can be included in the current solution, referred to as the set *Decisions*, are identified in a procedure outlined in Section 5.1.1. Once the possible decisions have been identified, they are sorted in ascending cost order,

Algorithm 1 Two-phase heuristic

```
1: Inputs: solutionDeadline, all parameter of Table 1.
2: while time < solutionDeadline do
3:    //Generate empty solution.
4:    if All customers served then
5:       //Construction phase.
6:       if Customers can be added using methods specified in Section 5.1.1 then
7:          Select and implement decision using biased randomised criteria.
8:       else
9:          Cul-de-sac reached, start again from line 2.
10:      end if
11:   else
12:      //Local search improvement phase.
13:      while Local search neighbourhoods contain improved solutions do
14:         if There are improving two-opt moves then
15:            Implement best improving two-opt move.
16:         else
17:            if There are improving cut-and-insert moves. then
18:               Implement best cut-and-insert move.
19:            else
20:               Update best overall solution, start again from line 2.
21:            end if
22:         end if
23:      end while
24:   end if
25: end while
26: Outputs: bestOverallSolution
```

and one is selected according to a geometric distribution (see Equation (30) in Bayliss et al. (2020b)) with parameter β, a process often referred to as *biased randomisation*. Setting $\beta = 1$ makes the constructive algorithm always choose the lowest cost decision, progressively lower values introduce more and more randomness, thereby introducing diversification into the iterative procedure. If no feasible decisions are available, the algorithm has effectively run into a cul-de-sac, whereby the solution cannot be feasibly completed, at which point the algorithm starts a new iteration. If all deliveries have been included in a routing solution, a local search improvement phase begins (Section 5.2).

5.1.1 Enumerating available decisions during the constructive heuristic phase

This section outlines a logical order in which different ways to insert a hitherto non-served customer into a current partial solution, such "ways" are referred to as feasible decisions thereafter. Firstly, the simplest ways are considered, and if no ways to include another delivery in the current solution are identified, we consider the next simplest way to include another delivery in the current solution. That is, the operations are applied in a hierarchical fashion, continuing only until a way to include another delivery in the current solution has been identified. At this point, the constructive heuristic selects and implements a feasible decision using the biased randomisation criteria. The logically ordered sequence of operations, line 6 of Algorithm 1, are enumerated as follows:

1. *Add a delivery to an already utilised vehicle's route*: This operation considers the possible ways of

adding a delivery to the end of existing vehicle's (open) route, or inserting a delivery somewhere within the route of an existing vehicle. Within this procedure, we first seek the most efficient option for each non-served customer considered, if that option is not feasible, all feasible options are considered and added to the list of available decisions.

2. *Change the type of an already utilised vehicle to enable another delivery*: In this operation similar options are considered to those in Operation 1, but where either the type of an existing vehicle is changed to a different one, or where two existing vehicles swap routes.

3. *Use an existing vehicle rendezvous to generate extra opportunities to make another delivery*: This operation considers including existing rendezvous locations in existing vehicle routes such that new delivery opportunities are generated. To do this, we may start a new vehicle route from an existing vehicle rendezvous because the remaining customers are out of range of the vehicles already involved in the existing vehicle rendezvous. We may also include an existing rendezvous in the route of a vehicle that starts their tour from a different depot from that of the vehicle already involved in the existing rendezvous.

4. *Change the type of an already utilised vehicle and use an existing vehicle rendezvous to generate extra delivery opportunities*: This operation combines operations 2) and 3), seeing whether converting or swapping already utilised vehicles and making use of existing vehicle rendezvous creates new delivery opportunities.

5. *Make another delivery using a new vehicle directly or in conjunction with an existing or new vehicle rendezvous*: If the constructive algorithm resorts to this operation, then another delivery cannot be feasibly added to the current solution without introducing either a new vehicle, a new vehicle rendezvous, or both. This operation considers the use of a new vehicle, starting its route from an existing rendezvous location (including an origin depot), which will invoke an additional fixed charge associated with the new vehicle. At this point we also consider any benefits of creating new rendezvous opportunities, involving existing, converted or swapped vehicles, or new vehicles.

6. *Add new vehicle rendezvous in the hope that applying the above operations again yield a feasible decision*: Upon reaching this operation, we determine that visiting more customers must depend on the creation of larger chains of vehicle rendezvous, as might be the case when a customer resides at distances of multiple times the maximum range of a vehicle from the necessary origin depot. In such a scenario, we consider creating new rendezvous opportunities, in the hope that it will be possible to make another delivery in the next, or later, iterations of the constructive heuristic. We give up adding new rendezvous if no required vehicle type is available, and no vehicle can be swapped or converted to a required type, or if no rendezvous can be added whether violating tour range or duration constraints.

The logic behind the ordering of operations is that later ones are generally redundant if earlier ones return feasible decisions. For example, there is no need to swap vehicle types (Operation 3) without a direct reason, there is no need to use new vehicles (Operation 5) if existing vehicles can be used, and there is no need to arrange a new vehicle rendezvous (Operation 6) if we can make use of an existing one. Later ones are generally also more expensive.

5.2 Local search improvement phase

While the constructive heuristic provides diversification, a local search provides an intensification mechanism. The local search phase focuses on improving the route efficiency of sub-routes, which consist of customer sequences without vehicle rendezvous in between, thereby leaving the logistic backbone of the solution intact. The local search procedure uses two neighbourhood structures that have previously been successfully applied to solve routing problems (Section 2.4): i) *two-opt*; and ii) *cut-and-insert*. A swap neighbourhood, as a third neighbourhood, was found to offer no additional benefit.

The two-opt neighbourhood structure is designed to remove inefficient loops and crossovers within individual vehicle routes. This can be achieved by reversing the order of a sub-sequence of customers within a vehicle's route. In the current problem, we discard two-opt moves which lead to: increased travel time; vehicle range violation; or customer time window violation.

The cut-and-insert neighbourhood finds alternative positions in the same or a different vehicle's route, for example, move a customer from the ninth to the fourth position in a vehicle's route. The same constraint checks are performed as for two-opt moves, with the addition of a check to see that vehicle weight or capacity constraints are not violated. If no improving cut-and-insert moves are found, we resort to checking whether cutting-and-inserting sub-sequences of customers yields an improving move. At this point, all customer sub-sequences, not interrupted by vehicle rendezvous, are considered for cut-and-insert.

The two neighbourhoods are applied within a steepest ascent variable neighbourhood search framework (Mladenović and Hansen, 1997). The two-opt neighbourhood is always checked before the cut-and-insert neighbourhood. The logic behind the ordering of the two neighbourhoods is that solution improving two-opt moves are non-invasive, only modifying part of a single vehicle's route, in a way that is generally always obviously beneficial. The best improving move, in the first neighbourhood with an improving move, is implemented. Once an improving move has been implemented, a new iteration of local search begins starting from the two-opt neighbourhood. If no improving moves are found in any neighbourhood, one iteration of the overall algorithm has been completed and a check is performed to see if a new best overall solution has been found. If time remains, a new iteration is initiated.

5.3 Heuristic solution representation

While the basic logic of the proposed solution methodology is fairly straightforward, the key to addressing the complexities of the considered problem lies in the design of the heuristic solution representation. That is, finding the best way to store a solution, which facilitates efficient constraint checking and implementation of potential solution modifications. For the considered problem, solutions can exhibit a tree-like structure, which is due to the possibility of vehicles transferring parcels between one another at rendezvous locations. Therefore, a solution for the considered problem is characterised by the paths followed by all parcels from their origin depot to the customer location along sub-paths of vehicle routes. In the simplest case, a parcel path will be a sub-path of a single vehicle's path. In the most complex cases, a parcel path will involve a sequence of vehicle sub-paths connected by vehicle rendezvous at mobile satellites and depots.

In reflection of this, a solution consists of the set $utilisedVehicles$ of utilised vehicles, where each vehicle has a route which is defined as an ordered sequence of edges denoted by $edges$. An edge defines a

trip from one depot, mobile satellite or customer to another such location, and has an associated distance, travel time and set of parcels carried. Each parcel also has a route defined by an ordered sequence of edges denoted by $parcelEdges$, starting at an origin depot and ending at the associated customer's location. Edges also have an associated earliest departure time and a latest departure time, which together define an edge's slack time. Earliest departure times are based on assuming that all tasks are performed as early as possible and depend on the start times of customer time windows. Conversely, latest departure times are based on assuming that all tasks are performed as late as possible and are influenced by the end times of customer time windows, as well as maximum duty shift durations. The difference between the two is used to compute whether there is sufficient time for an additional customer to be inserted in a particular position in a vehicle's route. Earliest and latest departure times are also influenced by vehicle rendezvous. For vehicle rendezvous that occur at mobile satellites, this gives rise to a concept referred to here as *parallel edges*, whereby, the edges of vehicles leaving a mobile satellite vehicle rendezvous location, are tied together, since the vehicles involved in a mobile satellite rendezvous must be present at that location simultaneously. Sets of parallel edge must then have matching earliest and latest departure times, i.e., the overlap region of their respective earliest and latest departure times. Every time a solution is modified, using any of the methods outlined in Section 5.1.1, we recalculate the earliest and latest departure times of all edges in the current solution. Two problem specific recursive procedures are introduced for this task, a forwards pass, for calculating earliest departure times, and a backwards pass, for calculating latest departure times. The forwards pass is so called because the recursive procedure starts at the beginning of routes and cycles forwards through vehicle routes and parcel paths. The backwards pass does the reverse.

Recursive functions are those which include themselves in their definition. For the case of a single vehicle with no vehicle rendezvous or customer time windows, the recursive forwards pass procedure for updating the earliest departure times of all edges in a vehicle's tour, would be Algorithm 2, which simply cycles through a vehicle tour, setting the earliest time it can begin to traverse each edge in its tour, terminating when it reaches the end of the tour.

Algorithm 2 setEarliestDepartureTimeSimpleCase$(edge, newEarliestDepartureTime)$

1: $edge.earliestDepartureTime = newEarliestDepartureTime$
2: **if** $edge.nextEdge$ exists **then**
3: **setEarliestDepartureTimeSimpleCase**$(edge.nextEdge, newEarliestDepartureTime + edge.time)$
4: **end if**

In the present case, the forwards pass procedure is implemented by firstly setting the earliest departure times of all edges to minus infinity, and then calling Algorithm 3 for each vehicle in the set $utilisedVehicles$ with the first edge of the vehicle's route and 0 as the input arguments. Algorithm 3 sets the earliest departure time of an edge and then recursively calls itself telling the next edges, in vehicle routes, what their earliest departure times are, while accounting for the edge travel time and the customer time window associated with the current edge. Algorithm 3 has two main parts, lines 3 to 15 which update the earliest departure time of the current edge, while accounting for any parallel edges. The earliest departure time of parallel edges are all set to the maximum earliest departure time of those parallel edges, and recursive calls to Algorithm 3 are made for each of the parallel edges. Then, lines 17 to 24, which invokes calls of Algorithm 3 to edges that directly depend on the current edge, while

Algorithm 3 *setEarliestDepartureTime*$(edge, newEarliestDepartureTime)$

Inputs: $edge, newEarliestDepartureTime$
if $newEarliestDepartureTime > edge, earliestDepartureTime$ **then**
  if *edge* is leaving a mobile satellite vehicle rendezvous **then**
    //Find the set of edges parallel with *edge* (E).
    $E \leftarrow getParallelEdges(edge)$
    //Set *newEarliestDepartureTime* to the maximum earliest departure time of the parallel edges.
    $newEarliestDepartureTime \leftarrow \max(p.earliestDepartureTime,\ \forall p \in E,\ newEarliestDepartureTime)$
    $edge.earliestDepartureTime = newEarliestDepartureTime$
    //Recursive call of forwards pass for parallel edges.
    for $p \in E \setminus \{edge\}$ **do**
      $setEarliestDepartureTime(p, newEarliestDepartureTime)$
    end for
  else
    $edge.earliestDepartureTime = newEarliestDepartureTime$
  end if
  //Account for travel time and the customer time window of parcels delivered by *edge*, before moving on to edges that depend directly on *edge*.
  $newEarliestDepartureTime = newEarliestDepartureTime + edge.time$
  $newEarliestDepartureTime = \max(newEarliestDepartureTime,\ edge.customer.timeWindowStartTime)$
  //Find the set of edges E that are the next edges of parcels carried by along *edge*, including that of the current vehicle and those of parcels transferred to different vehicles.
  $E \leftarrow nextEdges()$
  //Recursive call of forwards pass for next edges.
  for $e \in E$ **do**
    $setEarliestDepartureTime(e, newEarliestDepartureTime)$
  end for
end if
Outputs: no return value

accounting for the travel time and the time window start time of the customer at the end of the current edge.

The backwards pass is performed after the forwards pass and is largely analogous barring a few details. The backward pass is initiated by firstly setting the latest departure times of all edges to ∞, and then calling a $setLatestDepartureTime(edge, newLatestDepartureTime)$ algorithm, starting from the last edge of each vehicle's route, with the maximum end of duty shift time as the initial latest departure time input. In contrast to Algorithm 3, the $setLatestDepartureTime(edge, newlatestDepartureTime)$ algorithm begins by updating the value of *newLatestDepartureTime* first, setting it the minimum of that value and the end of the time window of the customer served by the edge, and then subtracting the edge traversal time. Following this everything is the same as in Algorithm 3, except that $\max()$ operations are replaced with $\min()$ (since a vehicle can't visit a customer after the end of their time window or the latest departure time of any parallel edge), *_EarliestDepartureTime_* replaced with *_LatestDepartureTime_* and *nextEdges* replaced with *previousEdges* (since the backwards pass cycles backwards through vehicle and parcel paths).

For ensuring that potential solution modifications do not violate vehicle capacity constraints, values for slack (remaining capacity) weight and volume are maintained for each edge on each vehicle's route, as solutions are built. Every time the parcel paths, satisfying a customer's order, are identified, a check is performed in order to ascertain whether all of the parcels will simultaneously fit in the vehicles transporting them without violating any capacity constraints, all the way from the origin depot(s) to the customer's location.

For each edge and rendezvous location, we maintain the set of all possible parcel paths connecting them to origin depots. Some of those paths might feature in the current solution while others might not. The set of possible parcel paths emanate from origin depots along the paths of vehicles that leave those depots during their tour. Whenever a vehicle visits a rendezvous location, an opportunity for parcels to be transferred between vehicles is created, in such cases, the possible parcels paths associated with the vehicles arriving at the rendezvous location split and continue along the routes of each vehicle leaving the rendezvous location.

When considering if a customer can be inserted between two locations joined by an edge, the set of parcel paths associated with that edge are used to identify the parcel paths that can be used to satisfy the customer's demand. The capacity feasible parcel paths that cause the least delay are selected, and ties are broken by selecting the paths involving the fewest vehicle rendezvous. Similarly, the set of possible parcel paths associated with rendezvous locations are used to identify the parcel paths that can be used to satisfy a customer's demand in ways which rely on the creation of new vehicle rendezvous.

For parcel paths that are not currently employed in the current solution, their use may introduce new parallel edges. As a result, it is necessary to perform forwards and backwards passes for the sets of possible parcel paths, to ensure that their use will not lead a violation of time constraints, where their use would lead to an edge with an earliest departure time in excess of their latest departure time.

When considering the creation of new vehicle rendezvous, it is vital to check that the move makes temporal sense. For example, if the current solution already involves the use of a vehicle rendezvous, then a temporal dependence exists between the edges of the involved vehicles. There are edges that occur after the rendezvous and those that do not. To avoid moves that make no sense temporally (i.e., those that would imply time travel), we cannot arrange a new rendezvous that mixes edges from before (after) any existing rendezvous with edges of another vehicle occurring after (before) that rendezvous. Since any vehicle can rendezvous with any other vehicle. Temporal dependence between pairs of edges may span any number of vehicle routes through sequences of vehicle rendezvous. Therefore, to avoid temporally infeasible moves, it is necessary to store (in a list or otherwise), for each edge, which edges depend on it temporally. To this end, whenever a new edge is added to a solution, we cycle back through all of the parcel paths available to the new edge and add the new edge to those lists.

6 Computational results

The aim of this section is to computationally validate the mathematical model of Section 4 and to assess the performance of the heuristic of Section 5. For the former, we use a set of 14 development instances described in Section 6.1. The latter is done in two ways, namely (i) by comparing the performance of the heuristic against existing methods for instances of the Open Routing Problem as a special case of the problem described in this paper, and (ii) by evaluating the heuristic against the exact solutions provided

by the mathematical model on small scale and real-world instances of the multi-modal and variable-echelon problem, described in Sections 6.2 and 6.3.1, respectively. We then move on to solving larger-scale instances of the problem and analyse the solutions in Sections 6.3.2 and 6.3.3. All experiments were conducted on a laptop with a 2.80 GHz CPU and with 8Gb RAM. The heuristic was coded in Java.

6.1 Validation of the formulation and the heuristic

In this section, exact solutions for the mathematical model presented in Section 4 are compared with heuristic solutions for a set of 14 small "development" instances, the optimal solutions of which are depicted in Figure 3. The instances were designed such that the optimal solutions were known in advance, owing to their small size and tight constraints, enabling a validation of the mathematical formulation and initial validation of the heuristic method. The 14 instances, labelled D1–14, gradually increase in complexity by introducing more vehicle types and the need for vehicle rendezvous. Instances D1–4 feature a single vehicle type, instances D5–10 feature two vehicle types, and instances D11–14 feature three vehicle types. Instances D1, D3, D6, D11 and D12 are instances where no vehicle rendezvous are required. Instance D2 requires that a single vehicle picks up parcels from two different depots during its delivery route. Instances D4, D7 and D8 each require single-vehicle to single-vehicle rendezvous at depots or mobile satellites. Instance D9 requires that a single vehicle transfers parcels simultaneously to two vehicles at a mobile satellite. Instance D10 requires two vehicles to rendezvous twice during their routes, while instance D14 is the same with the need for an additional rendezvous in between. Instance D13 requires that one vehicle transfers parcels to a second vehicle at a mobile satellite, who must then transfer some parcels to a third vehicle at another mobile satellite.

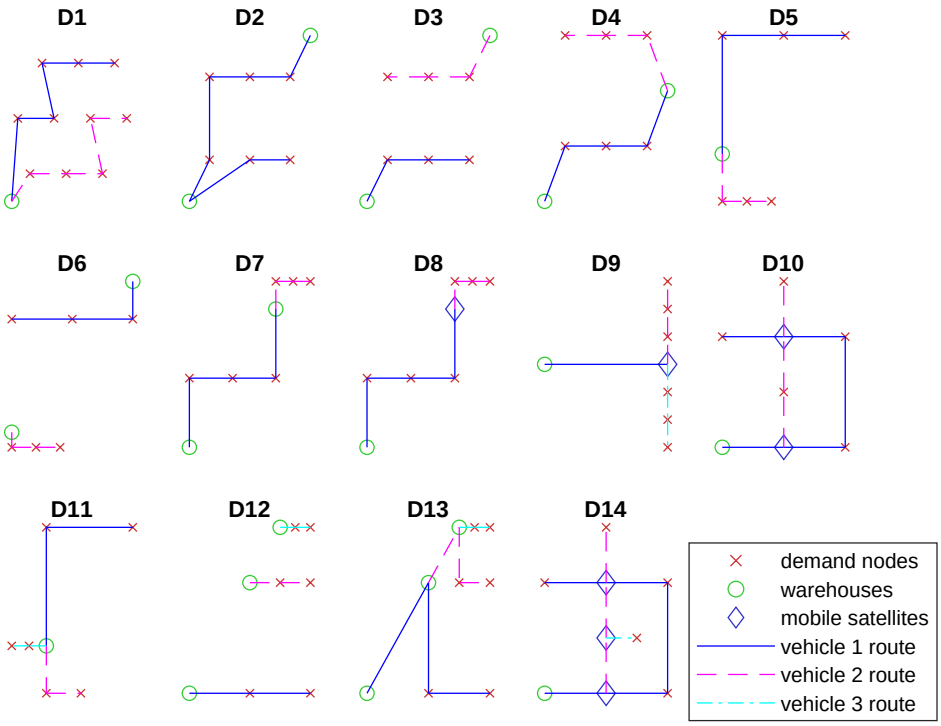

Figure 3: Optimal solutions to a set of "development" instances.

For instances D1–14, the mathematical model was solved to optimality using the commercial opti-

miser CPLEX version 12.10, while the heuristic was implemented as a single threaded Java application employing 1000 iterations. Table 2 shows that the heuristic was able to find the optimal solutions for all of the development instances in similar amounts of time, thereby providing a validation of the mathematical formulation and an initial validation of the consistency of the heuristic solution method.

Table 2: Performance of the mathematical model and the heuristic on the development instances

	Solution time (s)		Solution cost		Vehicles
Instance	Exact	Heuristic	Exact	Heuristic	used
D1	16.19	2.25	26.45	26.45	(2 0 0)
D2	1.67	1.86	15.32	15.32	(1 0 0)
D3	0.73	1.11	23.87	23.87	(2 0 0)
D4	2.04	1.16	24.47	24.47	(2 0 0)
D5	0.21	0.67	14.50	14.50	(1 1 0)
D6	0.24	0.65	14.50	14.50	(1 1 0)
D7	0.27	0.81	15.00	15.00	(1 1 0)
D8	0.30	1.19	15.00	15.00	(1 1 0)
D9	0.68	0.22	15.40	15.40	(1 2 0)
D10	1.06	0.33	21.00	21.00	(1 1 0)
D11	0.26	0.78	14.00	14.00	(1 1 1)
D12	0.28	0.84	15.00	15.00	(1 1 1)
D13	1.59	0.93	17.12	17.12	(1 1 1)
D14	29.26	0.51	22.50	22.50	(1 1 1)

6.2 Comparison with benchmark solutions for the open routing problem

In this section, we seek to validate the quality of the proposed heuristic by comparison with existing algorithms. Due to a lack of benchmark instances and solutions for the proposed problem, in this section we apply the proposed heuristic algorithm to benchmark problem instances for the open routing problem, which is a sub-problem of the problem introduced in this work. The open routing problem considers a single vehicle type with range and capacity constraints, one depot and customers with known demands for a single product type. The open routing problem instances and best known solutions (BKS) considered in the following can be found in Ruiz et al. (2019). In particular, we consider two sets of instances, firstly a set of 14 instances (C1–14), originally published in Christofides (1979), featuring nodes with a random spatial distribution, the best known solution are the result of 12 previous algorithms, as shown in Ruiz et al. (2019). Secondly, a set of 8 instances (KO1–8), originally published in Golden et al. (1998), featuring nodes arranged in circles centred upon a depot node, the best known solutions of which are based on 2 previous algorithms, also shown in Ruiz et al. (2019). For these experiments, the heuristic was implemented in a single thread mode, and the time limits was set to the same used to find the best known solutions while allowing for processor speed. The instances, described in Table 3, are characterised by the number n of customers, the capacity Q of a vehicle, the maximum tour duration T_{max} for each vehicle and service time s for each customer, shown under columns 2–4.

The solution cost column of Table 3 provides the costs of the solutions generated using the proposed algorithm, while the BKS column provides the best known solutions from previous algorithms. The main results provided by Table 3 are that the proposed algorithm provides solutions with an average of 3.37% higher solution cost than best known solutions for the first set of benchmark instances and

Table 3: Heuristic solution times for different numbers of customers and numbers of available vehicles (with different ranges)

Instance	n	Q	T_{max}	s	Vehicles used	Solution cost	BKS	%gap	Solution time (s)	Iterations
C1	50	160	∞	0	5	434.56	412.96	5.23	18.71	4214
C2	75	140	∞	0	11	591.82	564.06	4.92	25.61	2648
C3	100	200	∞	0	8	658.17	639.26	2.96	136.24	4543
C4	150	200	∞	0	12	769.42	733.13	4.95	324.71	3695
C5	199	200	∞	0	17	921.56	869.00	6.05	835.71	5729
C6	50	160	200	0	6	415.56	412.96	0.63	17.91	4870
C7	75	140	160	0	11	577.44	567.59	1.74	31.27	3376
C8	100	200	230	0	8	661.11	644.63	2.56	114.84	4253
C9	150	200	200	0	12	778.40	743.51	4.69	372.11	4828
C10	199	200	200	0	17	945.29	873.89	8.17	1070.98	7101
C11	120	200	∞	0	7	710.01	676.40	4.97	498.77	13407
C12	100	200	∞	0	10	536.53	534.24	0.43	55.17	1737
C13	120	200	720	50	10	863.26	862.74	0.06	356.50	5121
C14	100	200	1040	90	11	554.09	554.67	–0.10	81.33	2635
Average								3.37		
KO1	240	550	650	0	10	4662.06	4629.21	0.71	5095.82	14835
KO2	320	700	900	0	11	7606.28	7285.98	4.40	6200.96	6329
KO3	400	900	1200	0	12	10441.58	9837.78	6.14	6200.95	2723
KO4	480	1000	1600	0	12	13183.57	12431.50	6.05	6202.76	1352
KO5	200	900	1800	0	6	6177.15	6004.65	2.87	2454.98	8298
KO6	280	900	1500	0	8	8045.29	7731.21	4.06	6200.23	8229
KO7	360	900	1300	0	10	9619.41	9188.11	4.69	6200.49	3786
KO8	440	900	1200	0	12	10883.92	10488.50	3.77	6200.03	1938
Average								4.09		

4.09% higher solution cost for the second set. Additionally, the proposed heuristic found 1 best known solution, for instance C14. It is interesting to note that instances C13 and C14 are identical to instances C11 and C12 respectively, except for the addition of maximum tour length constraints, and that our algorithm found the same solutions for these respective instances, which was because the tour length constraints were never the binding constraints, the capacity constraints were. We conclude that the proposed heuristic can find reasonable solutions for a main sub-problem and attribute the non-negligible %gap to the specialisation of the heuristic's design for the considered problem.

6.3 Experiments with multi-modal variable-echelon problem instances

For the experiment results of Sections 6.3.1, 6.3.2 and 6.3.3, the customer demand details are based on real sales data within London. A customer order comprises a delivery location and a set of parcels, each of which has a size (scalar volume) and a weight. The average, minimum and maximum parcel volumes were 5680cm^3, 3cm^3 and 226800cm^3 respectively, while the average, minimum and maximum parcel weights were 0.8kg, 0.001kg and 38kg respectively. The spatial characteristics of the instance are depicted in Figure 4, where a depot located on the outskirts of London, and five mobile satellite locations are distributed across the city. The vehicle characteristics are given in Table 4. The values are in-line with those used by McLeod et al. (2020) in their London case study, and reflect the fact that

larger vehicles have higher range, capacity and cost compared to smaller ones.

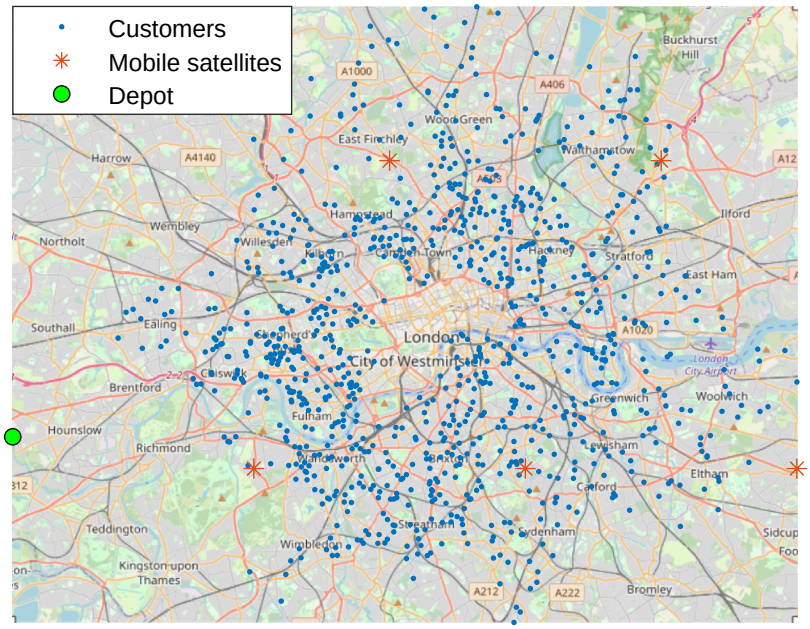

Figure 4: Node positions of instances based on real London sales data.

Table 4: Vehicle characteristics

Vehicle type	Van (type 1)	Cargo bike (type 2)	Porter (type 3)
Range (miles)	300	50	15
Maximum durations (hours)	8	8	8
Capacity (m^3)	12	1	0.35
Maximum weight (Kilograms)	1000	100	50
Average speed (miles per hour)	7.5	5	2.5
vehicle fixed costs (£)	28.9	15	5
Time variable costs (wages) (£ per hour)	15	12	10
Distance variable cost (fuel) (£ per mile)	0.15	0.05	0.01
Customer service time (minutes)	3	2	1
depot service time (minutes)	5	5	5
Mobile satellite service time (minutes)	5	5	5

6.3.1 Comparison of heuristic with exact solutions for small instances

In this section, we compare the quality of the heuristic against the solutions obtained by the mathematical programming formulation of the problem. The aim in this section is to determine the limits of the exact approach and to provide further validation of the heuristic method. For this purpose, twenty problem instances are considered, where the number of customers is increased from 1 to 20 and are labelled E1–E20. In each instance, the customers are selected at random from the sales data described in Section 6.3. The solver was CPLEX implemented in parallel mode with up to 8 threads and was given a maximum solution time of 3600 seconds. The results are presented in Table 5.

Table 5: Vehicle use and costs as the spatial density of customers is increased.

	Number of	Solution time (s)		Solution cost		% gap	
Instance	customers	Exact	Heuristic	Exact	Heuristic	MIP optimality	Heuristic
E1	1	0.34	0.64	46.31	46.31	0	0
E2	2	0.31	0.46	63.28	63.28	0	0
E3	3	0.46	0.47	80.67	80.67	0	0
E4	4	1.73	0.42	81.20	81.20	0	0
E5	5	2.41	0.45	89.50	89.50	0	0
E6	6	8.90	0.42	103.30	103.30	0	0
E7	7	22.75	0.44	103.78	103.78	0	0
E8	8	76.52	0.48	104.32	104.32	0	0
E9	9	192.90	0.50	115.90	115.90	0	0
E10	10	814.16	0.55	125.71	125.71	0	0
E11	11	1734.19	0.60	129.30	129.30	0	0
E12	12	3611.06	0.56	134.47	131.94	15.74	–1.88
E13	13	3614.16	0.80	133.08	133.08	22.99	0
E14	14	3619.44	0.64	138.95	138.90	21.46	–0.04
E15	15	3626.44	0.66	139.70	139.65	21.44	–0.04
E16	16	3634.50	0.76	160.45	140.69	36.23	–12.31
E17	17	3646.61	0.83	153.76	142.97	32.44	–7.01
E18	18	3658.70	0.83	182.98	158.50	35.14	–13.38
E19	19	3674.68	0.83	504.31	160.78	75.49	–68.12
E20	20	3697.36	1.04	537.30	161.53	77.95	–69.94

Table 5 shows that when the number of customers is between 1 and 11, the exact approach is able to provably find the optimal solutions, as indicated by the MIP optimality gap values. In those cases, the heuristic was able to find the optimal solutions as well, as indicated by the heuristic gap values. For instance E13, the heuristic yielded the same solution value as the MIP, where the latter terminated with a gap of 22.99% after an hour of computation. Relaxing the time limit for this instance established that the reported value is indeed optimal, but the solution time was nearly 4 hours. The heuristic solution times only exceeded 1 second for instance E20, which highlights the huge scalability advantage of the heuristic approach.

The exact approach scalability issues can be attributed to the large number of variables and constraints of the proposed mathematical model. For reasons of tractability reasons, a very coarse discretisation of time was used for instances E1–20 in order to avoid an excessive number of arc variables.

Regarding the valid inequalities presented in Section 4.1, it was found that adding symmetry breaking constraints (26), (27) and (28) had the effect of reducing solutions times by 8% on average, however the average solution costs, for instances that could not be solved in an hour, were 17% higher on average. Similar results occurred when the strengthened constraints (31) and (32) were included, there was a 2% reduction in solution times and a 5% increase in solution costs on average. Also, when the bound constraints (29) and (30) were included, there was a 3% reduction in solution times and a 5% increase in solution costs on average. An explanation for this result is that Constraints (15) and (16) alone are tighter constraints than Constraint (29), while Constraint (21) is tighter than Constraint (30). Overall, the expense of adding additional constraints outweighs the benefit. It is also worth noting that CPLEX has built in facilities for detecting symmetries in the branch and bound tree, as well as for removing

unnecessary constraints and variables from a MIP model.

6.3.2 Multi-modal variable echelon instances based on sales data in London

Three main types of scenarios are considered, which differ in the way that vehicles can be used to make deliveries. For each of the three types of scenarios there are six problem instances, which differ in the number of customer orders that must be fulfilled. The number of customers considered are 49, 95, 181, 340, 602 and 985 (corresponding to samples of 50, 100, 200, 400, 800 and 1600 individual parcels respectively) and the same sets of customers are considered in the respective instances in each of the three scenarios considered. The three types of scenarios are as follows:

1. Just vans are allowed to make deliveries and no rendezvous are allowed at mobile satellite locations. The instances in this set are labelled S1-6 respectively.

2. Vans, cargo bikes and porters are allowed to make deliveries, but no rendezvous are allowed at mobile satellite locations. The instances in this set are labelled M1-6 respectively.

3. Vans, cargo bikes and porters are allowed to make deliveries, and rendezvous are allowed at mobile satellite locations. The instances in this set are labelled MM1-6 respectively.

The aim behind the design of this set of instances is two-fold: i) to demonstrate the potential benefits of deploying multi-modal and variable-echelon delivery systems under realistic scenarios; and ii) to provide an initial set of benchmark instances for the defined routing problem. The customer delivery time windows represent typical delivery hours of the day.

Table 6: Three scenario experiment results

Customers	Instance	Solution cost (£)	Solution time (s)	Vehicles used	Total miles	Van miles	Cargo bike miles	Iterations
49	S1	264.72	6.39	(2)	77.98	77.98	0	4000
	M1	253.52	6.07	(1 1 0)	78.00	44.92	33.08	4000
	MM1	241.23	14.57	(1 1 0)	72.82	41.92	30.90	4000
95	S2	407.37	37.86	(3)	114.26	114.26	0	4000
	M2	394.24	39.33	(2 1 0)	115.78	83.57	32.21	4000
	MM2	380.45	87.58	(2 1 0)	108.96	76.83	32.13	4000
181	S3	659.34	264.73	(5)	173.42	173.42	0	4000
	M3	629.48	274.04	(3 2 0)	174.15	122.76	51.38	4000
	MM3	610.02	515.54	(3 2 0)	164.81	99.18	65.63	4000
340	S4	985.01	2678.42	(7)	241.38	241.38	0	4000
	M4	943.20	2760.22	(5 2 0)	245.43	190.03	55.40	4000
	MM4	898.51	4207.03	(5 2 0)	223.78	165.20	58.58	4000
602	S5	1521.38	7210.12	(11)	343.37	343.37	0	1212
	M5	1462.14	7201.60	(8 3 0)	352.23	275.71	76.52	1141
	MM5	1342.28	7219.09	(7 3 0)	308.27	225.78	82.49	920
985	S6	2196.14	7287.29	(15)	467.51	467.51	0	203
	M6	2014.31	7232.74	(9 7 0)	463.36	287.41	175.95	188
	MM6	1916.97	7292.63	(10 5 0)	412.48	290.02	122.46	158

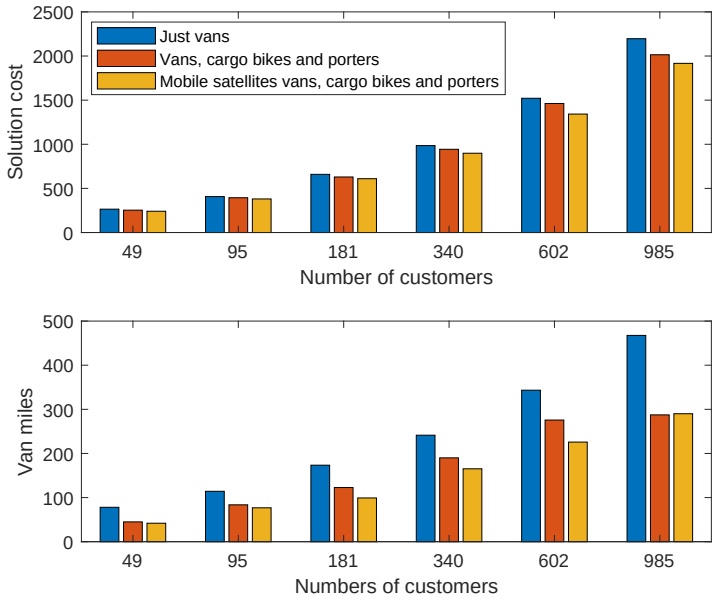

Figure 5: Solution costs (top) and van miles (bottom) for instances S1-6, M1-6 and MM1-6.

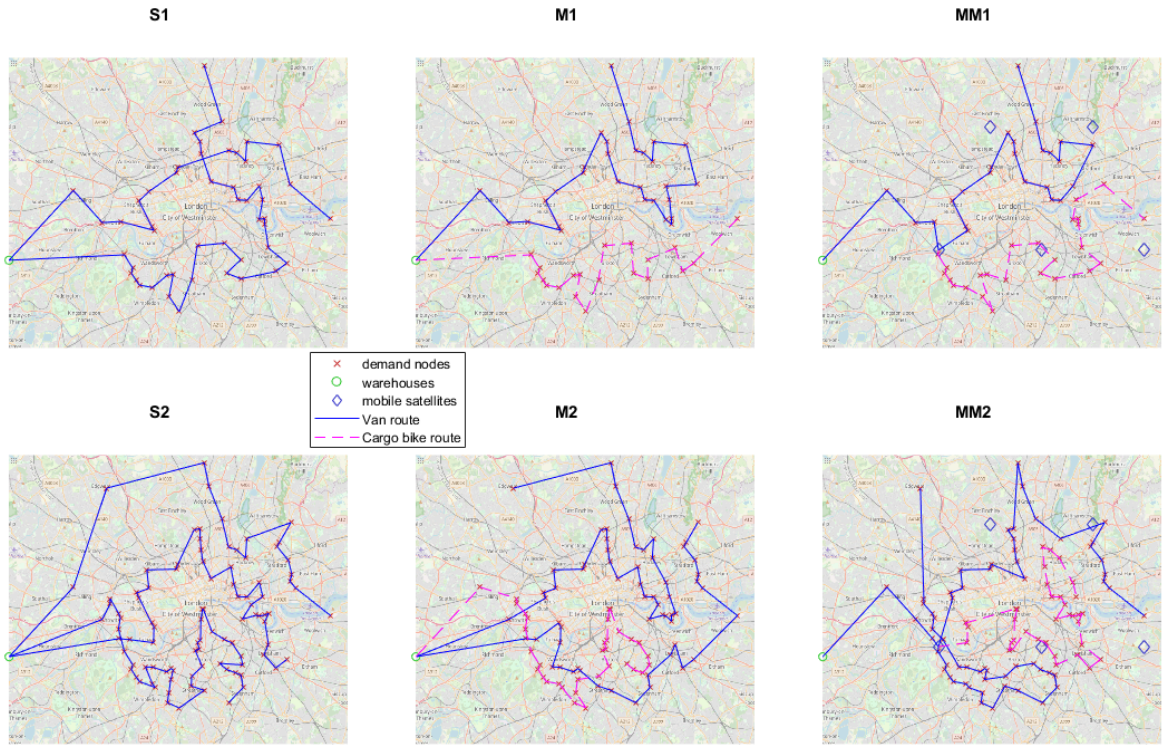

Figure 6: Vehicle route solutions for instances involving 49 (top) and 95 (bottom) customers.

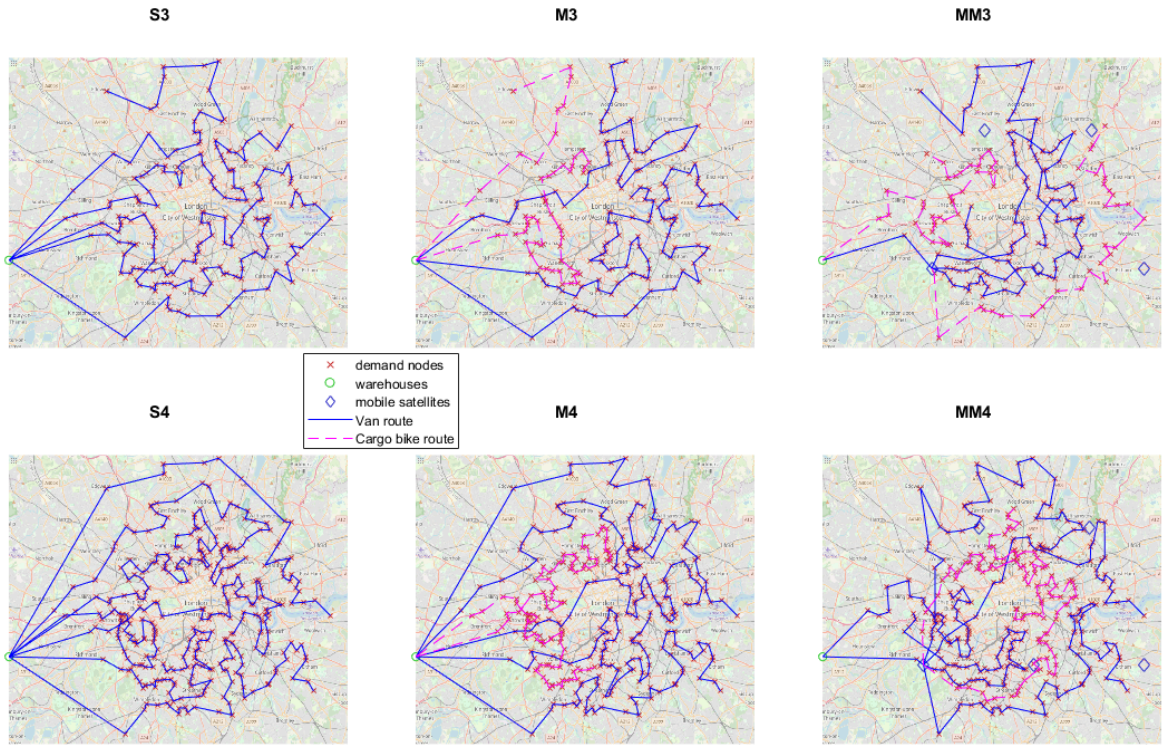

Figure 7: Vehicle route solutions for instances involving 181 (top) and 340 (bottom) customers.

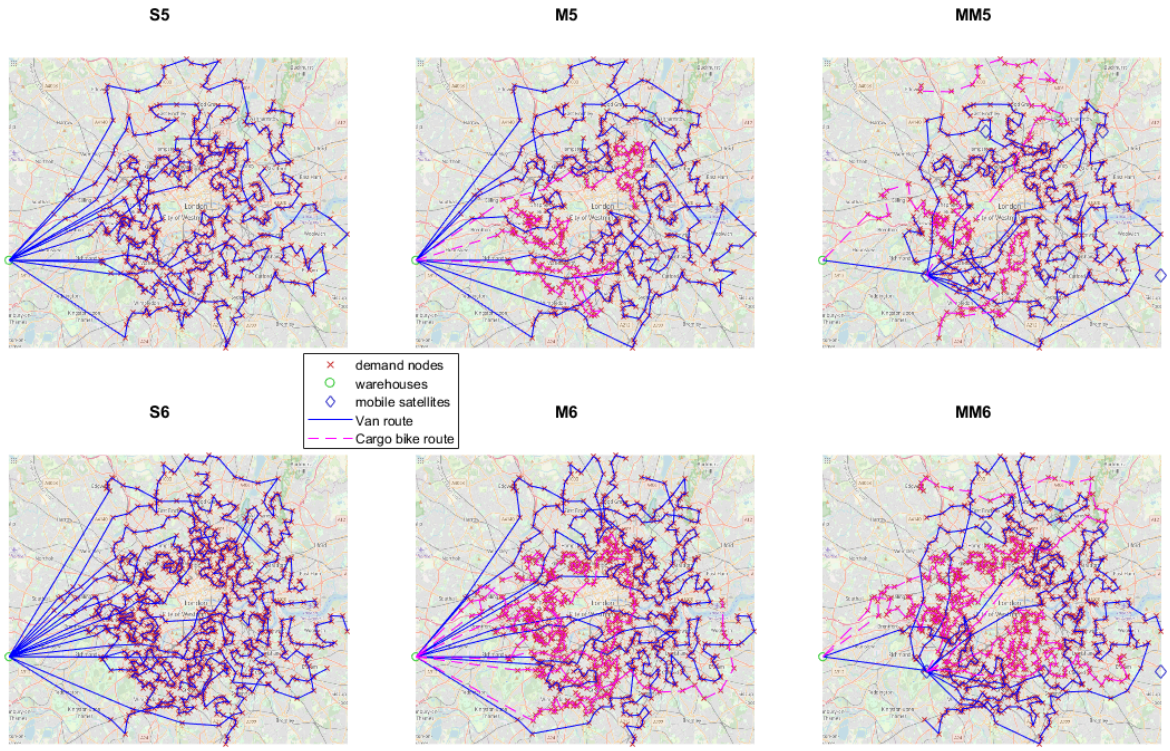

Figure 8: Vehicle route solutions for instances involving 602 (top) and 985 (bottom) customers.

To allow for larger instance sizes and more meaningful results, the 10-minute time limit was replaced with a 2 hour time limit. 4 threads were run in parallel, and the total number of iterations was limited to 4000. Table 6 provides solution costs, solution times, the numbers of vehicles of each type used in each solution, total miles, van miles and cargo bike miles and the number of iterations completed before the algorithm's termination criteria were met. The results show that for each problem size, scenario 3, i.e., allowing multiple vehicle types and the use of vehicle rendezvous at mobile satellites, leads to the lowest solution costs. Similarly, scenario 2, always provides lower cost solutions than scenario 1, thus highlighting the potential benefits of utilising smaller vehicles with lower fixed and variable costs. These results can be seen clearly in Figure 5 (top). Table 6 shows how the relative savings, between the three scenarios, were made. Scenarios 2 and 3 both exploits vehicle types with lower fixed and variable costs, and as a result reduces van miles, as emphasised by Figure5 (bottom), and therefore tailpipe emissions. Scenario 3 can also significantly reduce the total distance that must be travelled, which can be attributed to reduced depot stem costs. The numbers of vehicle rendezvous, for instances MM1-MM6, is 1, 2, 3, 5, 8 and 11 respectively. Figures 6, 7 and 8 provide solution diagrams which illustrate these points.

Solution times increase with problem size. In instances where the number of customers was 602 and 985, the 2-hour time termination criterion came into play. Whilst we allow the heuristic to run for a long time, our computational investigation indicated that it is able to identify fairly good solutions even within the first few iterations, with the remaining iterations showing diminishing returns. For example, for instance MM3, a solution within 0.86% of the best overall was found with 2.55% of the total run-time. For instance S5, a solution within 1.37% of the best overall was found within 3.53% of the total run-time. It can also be seen that in these cases, the fewest iterations were completed for corresponding scenario 3 instances, and the most were completed for corresponding scenario 1 instances and can be explained by the relative sizes of the solution spaces of the three scenarios. Scenario 2 introduces a degree of freedom regarding vehicle types, while scenario 3 introduces a degree of freedom regarding where vehicle tours start. Table 6 also indicates that as the number of customers and their density increases, the average cost per delivery decreases, indicating an economy of scale effect.

6.3.3 Demand density

In this section we examine the impact of average parcel size and customer density on the resulting fleet composition. Using the 181 customer instance MM3 of Section 6.3.2 as the control problem, we modify the the spatial distribution of customers by applying scale factors 1.5, 1, 0.8, 0.6, 0.4, 0.2, 0.1 to all coordinates. A scale factor of 1.5, for example, increases the distance between all pairs of locations by 50%, and a scale factor of 0.1 reduces the distance between all pairs of locations to 10% of their original values. We also apply scale factors of 0.1, 0.5, 1, 5, 10, 20, 30 to the sizes and weights of all parcels. This two-factor experiment has 49 individual instances. The time limit for each instance was set at 10 minutes, and the vehicle fleet composition results are provided in Table 7.

Table 7 has columns for each parcel size scale factor, and rows for each spatial scale factor. The third row also reports the total weights of all customer parcels after applying the parcel size scale factors. In general, Table 7 shows that higher parcel sizes and weights increase the required number of larger vehicles, especially vans, which is due the corresponding higher capacity requirements in such scenarios. When the spatial distribution of customers is lower, the number of required vehicles decreases. When parcel sizes are small and customers are near one another, fleets tend to be composed of smaller vehicle

Table 7: Fleet compositions (vans, cargo bikes, porters) for different spatial scale factors and total parcels weights applied to instance MM3.

Parcel size scale factors	0.1	0.5	1	5	10	20	30
Spatial scale factors	Total weight of all parcels (kilograms)						
	12.14	60.7	121.41	607.03	1214.07	2428.14	3642.21
1.5	(5 1 0)	(5 1 0)	(5 1 0)	(6 0 0)	(6 1 0)	(7 0 0)	(7 0 0)
1	(3 2 0)	(3 2 0)	(3 2 0)	(3 2 0)	(4 1 0)	(4 1 0)	(5 0 0)
0.8	(2 2 0)	(2 2 0)	(2 2 0)	(3 1 0)	(3 1 0)	(3 1 0)	(4 0 0)
0.6	(0 4 0)	(0 4 0)	(0 4 0)	(2 1 0)	(3 0 0)	(3 0 0)	(4 0 0)
0.4	(0 3 0)	(0 3 0)	(0 3 0)	(1 2 0)	(2 1 0)	(3 0 0)	(4 0 0)
0.2	(0 1 1)	(0 1 1)	(0 1 1)	(2 0 0)	(2 0 0)	(3 0 0)	(4 0 0)
0.1	(0 0 2)	(0 0 2)	(0 1 1)	(1 1 0)	(2 0 0)	(3 0 0)	(4 0 0)

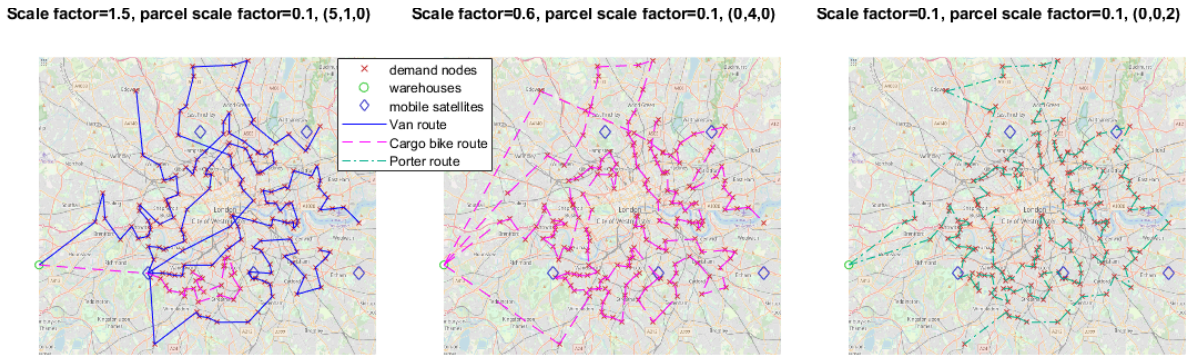

Figure 9: Vehicle route solutions for instance MM3 With parcel size scale factor 0.1 and spatial scale factors 1.5, 0.6 and 0.1.

types such as porters and cargo bikes, because the capacity and range requirements are lower in such scenarios, and such vehicle types have both lower fixed and variable costs. When parcel sizes are very high (last column) reducing the spatial distribution of customers does not reduce the required number of vans, because in those scenarios, cargo bikes and porters do not have sufficient carrying capacity.

In the examples considered here, porters were used very little, which can be explained by their relatively low carrying capacities and higher (full duration tour) average cost per mile, which owes to their relatively low average speed, which increases tour times and therefore wage costs. Porters were only ever used in cases where customer density was highest and while, at the same time, parcel sizes and weights were smallest. Figure 9 shows the vehicle route solutions for the cases where the parcel size scale factor is 0.1, while the spatial scale factors are 1.5, 0.6 and 0.1. When the spatial scale factor is 0.1 porters are used, it is then likely, that in the non-scaled instances, that if the total demand were of the order of tens of thousands, porters would become a more cost effective delivery mode.

7 Conclusions

This work proposes a multi-modal and variable-echelon last-mile delivery system. Multi-modal delivery allows for parcels to be passed between different vehicle types, during the last-mile delivery. Variable-echelon means that parcels can be transferred both ways between any pair of vehicle types, during last-mile delivery. Parcel transfers are allowed at kerb-side transfer points. This work solves the routing problem, including parcel transfers and associated vehicle synchronisation, ensuring that all parcels are delivered. Compared to previous research which focus on fixed-echelon delivery system, the variable-echelon relaxation proposed here vastly increases the size of the combinatorial optimisation problem being solved.

A mathematical formulation is provided and solved for small instances. A scalable two-phase heuristic solution methodology is provided. The heuristic is validated firstly, by comparison to optimal solutions for small instances, and then by comparison with existing algorithms in a sub-problem of that considered here, in which the proposed heuristic is able obtain solutions that are, on average, within a few percent of the quality of the best-known solutions.

Computational results on instances with real sales data show that allowing the use of alternative vehicle types, such as cargo bikes, and making use of vehicle rendezvous at mobile satellites, not only reduces the overall cost but also decreases van mileage by shifting parcels onto more environmentally friendly modes of transport. The experiments demonstrate how slower and cheaper vehicle types, despite costing more per mile, provide the most preferable delivery modes when customer density is sufficiently high and when parcel sizes are small, which has the potential of improving the environmental performance of last-mile logistics systems. This in turn suggests that operational changes such as the multi-modal delivery system introduced in this paper may be a simpler-to-implement and a more viable alternative to other innovations described for urban logistics that require infrastructural changes, which can be costly and require a long time to build.

Since this work introduces a new problem variant, there are several avenues for further research. One is to develop more efficient mathematical models and exact solution algorithms for the problem to obtain optimal solutions for large-scale instances in relatively short computational times. For example, one may investigate the performance of a three-index continuous-time mathematical formulation that would

avoid the need for time discretisation, in comparison to the one described in this paper that uses four-index variables. Second, the proposed model could also be extended to account for stochastic or time-dependent travel times that may better characterise some urban environments. Finally, new benchmark instances could also be introduced for exploring the proposed delivery system, perhaps representing other cities and vehicle type options, and the generality of the proposed model should facilitate this.

Acknowledgements

Funding for this work has been by InnovateUK (Project No: 45624) which is gratefully acknowledged. Thanks are due to three anonymous reviewers for their constructive comments.

REFERENCES

Allen, J., Bektaş, T., Cherrett, T., Bates, O., Friday, A., McLeod, F., Piecyk, M., Piotrowska, M., Nguyen, T., and Wise, S. (2018). The scope for pavement porters: Addressing the challenges of last-mile parcel delivery in london. *Transportation Research Record*, 2672(9):184–193.

Allen, J., Bektaş, T., Cherrett, T., Friday, A., McLeod, F., Piecyk, M., Piotrowska, M., and Austwick, M. Z. (2017). Enabling a freight traffic controller for collaborative multidrop urban logistics practical and theoretical challenges. *Transportation Research Record*, 2609:77–84.

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

Arnold, F., Cardenas, I., Sörensen, K., and Dewulf, W. (2018). Simulation of b2c e-commerce distribution in antwerp using cargo bikes and delivery points. *European Transport Research Review*, 10(2):1–13.

Ballare, S. and Lin, J. (2020). Investigating the use of microhubs and crowdshipping for last mile delivery. *Transportation Research Procedia*, 46:277–284.

Bayliss, C., do C. Martins, L., and Juan, A. A. (2020a). A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem. *Computers & Industrial Engineering*, 148:106695.

Bayliss, C., Juan, A. A., Currie, C. S., and Panadero, J. (2020b). A learnheuristic approach for the team orienteering problem with aerial drone motion constraints. *Applied Soft Computing Journal*, 92:106280.

Boysen, N., Fedtke, S., and Schwerdfeger, S. (2021). Last-mile delivery concepts: a survey from an operational research perspective. *OR Spectrum*, 43:1–58.

Cao, J. X., Wang, X., and Gao, J. (2021). A two-echelon location-routing problem for biomass logistics systems. *Biosystems Engineering*, 202:106–118.

Christofides, N. (1979). *Combinatorial optimization*, volume 1. Wiley-Interscience, Chichester. Based on a series of lectures, given at the summer school in combinatorial optimization.

Crainic, T. G., Ricciardi, N., and Storchi, G. (2009). Models for evaluating and planning city logistics systems. *Transportation Science*, 43:432–454.

Cuda, R., Guastaroba, G., and Speranza, M. G. (2015). A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199.

de Mello Bandeira, R. A., Goes, G. V., Gonçalves, D. N. S., de Almeida D'Agosto, M., and de Oliveira, C. M. (2019). Electric vehicles in the last mile of urban freight transportation: A sustainability assessment of postal deliveries in rio de janeiro-brazil. *Transportation Research Part D*, 67:491–502.

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

Dündar, H., Ömürgönülşen, M., and Soysal, M. (2021). A review on sustainable urban vehicle routing. *Journal of Cleaner Production*, 285:125444.

Enthoven, D. L. J. U., Jargalsaikhan, B., Roodbergen, K. J., uit het Broek, M. A. J., and Schrotenboer, A. H. (2020). The two-echelon vehicle routing problem with covering options: City logistics with cargo bikes and parcel lockers. *Computers & Operations Research*, 118:104919.

Ford (2019). Ford new take on getting parcels to your door could help speed deliveries, ease congestion and improve air quality. Available at https://media.ford.com/content/fordmedia/feu/en/news/2019/02/18/ford_s-new-take-on-getting-parcels-to-your-door-could-help-speed.html.

Goldberg, J. (2019). 4 reason profits remain elusive for e-commerce retailers. https://www.forbes.com/sites/jasongoldberg/2019/08.14/four-reasons-why-profits-remain-elusive-for-e-commerce-retailers/?sh=1d4d88cc1b5e.

Golden, B. L.and Wasil, E. A., Kelly, P., J., and Chao, I.-M. (1998). *The impact of metaheuristics on solving the vehicle routing problem: Algorithms, problem sets, and computational results.*, pages 33–56. Springer.

Grangier, P., Gendreau, M., Lehuédé, F., and Rousseau, L.-M. (2016). An adaptive large neighborhood search for the two- echelon multiple- trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254:80–91.

Hemmelmayr, V. C., Cordeau, J.-F., and Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Transportation Science*, 39:3215–3228.

Janjevic, M., Merchán, D., and Winkenbach, M. (2021). Designing multi-tier, multi-service-level, and multi-modal last-mile distribution networks for omni-channel operations. *European Journal of Operational Research*, 294:1059–1077.

Janjevic, M. and Winkenbach, M. (2020). Characterizing urban last-mile distribution strategies in mature and emerging e-commerce markets. *Transportation Research Part A*, 133:164–196.

Kedia, A., Kusumastuti, D., and Nicholson, A. (2020). Locating collection and delivery points for goods' last-mile travel: A case study in new zealand. *Transportation Research Procedia*, 46:85–92.

Letchford, A. N. and Salazar-González, J.-J. (2015). Stronger multi-commodity flow formulations of the capacitated vehicle routing problem. *European Journal of Operational Research*, 244:730–738.

Li, H., Wang, H., Chen, J., and Bai, M. (2020). Two-echelon vehicle routing problem with time windows and mobile satellites. *Transportation Research Part B*, 138:179–201.

Lin, Y. H., Wang, Y., He, D., and Lee, L. H. (2020). Last-mile delivery: Optimal locker location under multinomial logit choice model. *Transportation Research Part E*, 142:102059.

Liu, C., Wang, Q., and Susilo, Y. O. (2019). Assessing the impacts of collection-delivery points to individual's activity-travel patterns: A greener last mile alternative? *Transportation Research Part E*, 121:84–99.

Martinez-Sykora, A., McLeod, F., Lamas-Fernandez, C., Bektaş, T., Cherrett, T., and Allen, J. (2020). Optimised solutions to the last-mile delivery problem in london using a combination of walking and driving. *Annals of Operations Research*, 295:645–693.

McLeod, F., Cherrett, T., Bektaş, T., Allen, J., Martinez-Sykora, A., Lamas-Fernandez, C., Bates, O., Friday, K. C. A., Piecyk, M., and Wise, S. (2020). Quantifying environmental and financial benefits of using porters and cycle couriers for last-mile parcel delivery. *Transportation Research Part D*, 82:102311.

Mladenović, N. and Hansen, P. (1997). Variable neighbourhood search. *Computers & Operations Research*, 24:1097–1100.

Ofcom (2019). Annual monitoring update on the postal market. Financial year 2018-19. Technical Report. Available at https://www.ofcom.org.uk/__data/assets/pdf_file/0028/186139/annual-monitoring-update-postal-market-18-19.pdf.

Pichka, K., Bajgiran, A. H., Petering, M. E., Jang, J., and Yue, X. (2018). The two echelon open location routing problem: Mathematical model and hybrid heuristic. *Computers & Industrial Engineering*, 121:97–112.

Quetschlich, M., AndréMoetz, and Otto, B. (2021). Optimisation model for multi-item multi-echelon supply chains with nested multi-level products. *European Journal of Operational Research*, 290:144–158.

Ruiz, E., Soto-Mendoza, V., Barbosa, A. E. R., and Reyes, R. (2019). Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm. *Computers & Industrial Engineering*, 133:207–219.

Sluijk, N., Florio, A. M., Kinable, J., Dellaert, N., and Woensel, T. V. (2022). Two-echelon vehicle routing problems: A literature review. *European Journal of Operational Research*, Available online.

Vidal, T., Laporte, G., and Matl, P. (2020). A concise guide to existing and emerging vehicle routing problem variants. *European Journal of Operational Research*, 286:401–416.

Yu, V. F., Winarno, Lin, S.-W., and Gunawan, A. (2020). Design of a two-echelon freight distribution system in an urban area considering third-party logistics and loading–unloading zones. *Applied Soft Computing Journal*, 97:106707.

Zhang, L., Matteis, T., Thaller, C., and Liedtke, G. (2018). Simulation-based assessment of cargo bicycle and pick-up point in urban parcel delivery. *Procedia Computer Science*, 130:18–25.

Zhou, L., Baldacci, R., Vigo, D., and Wang, X. (2018). A multi-depot two-echelon vehicle routing problem with delivery options arising in the last mile distribution. *European Journal of Operational Research*, 265:765–778.