Solving the vehicle routing problem with multi-compartment vehicles for city logistics

Reza Eshtehadi[†], Emrah Demir[‡,*], Yuan Huang[‡]

[†] *Faculty of Engineering and Natural Sciences, Sabanci University, Istanbul, Turkey*

[‡] *Logistics and Operations Management, Cardiff Business School, Cardiff University, Cardiff, United Kingdom*

[ι] *Panalpina Centre for Manufacturing and Logistics Research, Cardiff Business School, Cardiff University, Cardiff, United Kingdom*

Abstract

Logistics companies are under increasing pressure to overcome operational challenges and sustain profitable growth while dealing with the newest requirements of their customers. One of the remedies designed to cope with a higher number of shipments is to use multi-compartment city vans to ensure all forms of integration with deliveries. In the area of city logistics, the most common type of delivery involves storing inventory in a central warehouse and to deliver customers' orders with multi-compartment vehicles. The problem under study is denoted as the vehicle routing problem with multi-compartment vehicles which are to operate from a single depot to visit customers within the chosen time period by minimizing major operational costs. We propose an enhanced adaptive large neighborhood search algorithm for the investigated routing problem. The computational results highlight the efficiency of the proposed algorithm in terms of both solution quality and solution time and also provide useful insights for city logistics.

Keywords: City logistics, freight transport, vehicle routing problem, multi-compartment vehicles, time windows, metaheuristic algorithm

1. Introduction

Freight logistics deals with the process of the flow of goods, information and other related resources from an origin point to a destination to satisfy customers' requirements (Christopher, 2016). Logistics Service Providers (LSPs) manage and control such logistical activities to satisfy customers' requirements at the lowest possible cost and risk. It is also important to provide prompt delivery and environmentally-friendly solutions to sustain profitable growth.

According to Ghiani et al. (2013), there are two types of freight transportation, namely long-haul and short-haul transportation. In the first type of transportation, goods are transported over a relatively long distance, ranging from at least hundreds to thousands of kilometers. The second type of transportation refers to a relatively small distance within a city or small-sized country (Crainic et al., 2009). The core operational-level problem in short-haul transportation is the finding of efficient routes for vehicles (Toth and Vigo, 2014). When LSP has multiple customers and multiple requests at the same time, the problem is referred to as the Vehicle Routing Problem (VRP), as introduced by Dantzig and Ramser (1959). The

Email address: reza.eshtehadi@sabanciuniv.edu,demire@cardiff.ac.uk, huangy66@cardiff.ac.uk (Reza Eshtehadi[†], Emrah Demir[‡,*], Yuan Huang[‡])

* Corresponding author

VRP is generally solved by designing collection or delivery routes for a set of vehicles, considering several side constraints (Laporte, 2007).

With the increase in e-commerce delivery volumes, the role of customers has switched from a passive position to an active one, whereby the ability to satisfy customers' needs with regards to delivery times, delivery costs and emissions (i.e., greenhouse gases and air pollutants) has become a very important success factor in a highly competitive industry (Savelsbergh and Van Woensel, 2016). However, the ever-increasing number of shipments and time pressure has put a strain on LSPs, especially in urban environments. In order to tackle these challenges, city logistics has gained growing interest in the last decade. In the operations research literature, city logistics is used to describe distribution which takes place in urban (populated) areas and introduces new strategies and best practices that can help to improve transport efficiency while reducing negative externalities of transportation. This paper will investigate a real-life VRP where customers' preferences are taken into account while operational costs are minimized in an urban environment. We now briefly highlight the features of the routing problem studied in this paper.

First, in city logistics, customers can purchase different types of products online, considering a wide range of products that an online companies can sell (Taniguchi and Thompson, 2018a). Some of these products might require temperature-controlled transportation. For example, food and drink, floricultural industries might require vehicles with multiple compartments to ensure different temperature levels for the goods being transported. This makes the problem at hand more challenging since it is not just the capacity of a single vehicle that must be considered, but the capacity and volume of each compartment (Muyldermans and Pang, 2010). Moreover, additional costs to allow vehicles to provide certain temperature requirements from the loading of goods to the point of delivery should be taken into account in the planning phase.

Second, customers' delivery preferences and shopping habits heavily affect the success of distribution because city logistics is generally customer driven (Lee and Whang, 2001). To this end, this study considers three types of online customer profiles as broadly mentioned in Insight (2016). The first type of customers place their order before midnight and expects delivery on the following day. In the second category, customers place their orders on the day of delivery, from the midnight to noon, and can be served in one of the planning periods (or cycles, i.e., morning, afternoon or evening). The last group of customers place their orders on the day of delivery and these orders need to be served in the next planning period. The delivery plan for customers should accommodate the requirements of customers and therefore we study multiple delivery cycles in a single day. This dynamic behavior makes the VRP more practical and relevant for city logistics.

Third, the standard objective in VRPs is generally to minimize the total cost which is based on distance. However, in our case, we consider the minimization of the major operational costs in city logistics (Taniguchi and Thompson, 2018b). These include fuel costs, driver wage(s) and temperature costs, when products need to be kept at certain temperatures during transportation.

The studied routing problem is an extension of the classical VRP with time windows (VRPTW) because it aims to find vehicle routes for customers with delivery time preferences. It is also more complicated compared to the standard VRPTW because we propose three time planning periods for three different types of customer group. We also consider the delivery of multiple products within the same order from each customer. Therefore, the problem at hand concerns routing a fleet of multi-compartment vehicles with each compartment having a finite capacity to satisfy the various order preferences of customers.

The main contributions of this paper are the following: (*i*) we consider a realistic VRP for city logistics; (*ii*) we propose an enhanced adaptive large neighborhood search (ALNS) algorithm. ALNS was used in solving a variety of VRPs (see e.g. Hemmelmayr et al. (2012), Franceschetti et al. (2017)). We also introduce new removal and insertion operators which can also be used for other types of VRP. New operators are designed to consider the multiple compartment and periodic planning features of the investigated routing problem, specifically tailored for the feasibility of the problem. And finally, (*iii*)

we provide numerical results to highlight that our algorithm is fast and performing well under various problem settings.

The rest of our paper is structured as follows: Section 2 provides the recent and related VRP literature. The main features of the problem are presented in Section 3. In Section 4, we provide the details of ALNS algorithm whereas Section 5 presents the computational experiments. Conclusions, insights and possible future research directions are provided in Section 6.

2. Literature review

We now briefly discuss the recent literature on city logistics and Vehicle Routing Problems with multi-compartment vehicles.

2.1. City logistics

As widely discussed in the literature, city logistics is the process to optimize all transportation and logistics activities in an urban environment. Interested readers are referred to the review paper of Savelsbergh and Van Woensel (2016) who discuss the importance of city logistics in the transport community. In this review, the authors discuss the main trends impacting city logistics such as population growth, urbanization, e-commerce growth, time flexibility, the sharing economy and sustainability. They also mention digital connectivity, big data, automation, and automotive technology as the main advances in technology that help to advance city logistics. In another work, different aspects of city logistics (i.e., the importance of reducing emissions) and research opportunities for future research are detailed by Taniguchi and Thompson (2018a).

Sustainability, and particularly emissions, are the most discussed topics in city logistics. Several authors have particularly focused on the environmental issues. For example, Yao et al. (2019) studied the effect of collaboration in city logistics from the aspects of profit and CO_2 emissions. The authors showed that collaboration in city logistics could improve transportation costs and help reduce emissions at the same time. In another study, Groß et al. (2019) proposed a city logistics VRP with uncertain travel times. The authors proposed a robust VRP for city logistics with satellite locations and tested their methodology with a case study in Stuttgart, Germany. In another case study, Wang et al. (2016) proposed a network minimal cost flow problem for last-mile delivery problems in city logistics and used their approach for Singapore and Beijing data sets. Their results showed that their proposed methodology could support real-time last-mile delivery optimization on a large-scale network. In another study, Sampaio et al. (2019) illustrated the characterizing features of crowd city logistics and discussed future research directions.

Xu et al. (2015) studied intermodal transportation auctions for the B2B e-commerce logistics problem. The authors proposed efficient auctions to minimize the sum of intermodal service costs and transaction costs (i.e., logistics chain cost). Later, Eshtehadi et al. (2017) studied demand and travel time uncertainty in green transport planning by proposing several robust optimization techniques; soft worst case, hard worst case and chance constraints.

2.2. VRP with multi-compartment vehicles

The presence of capacity constraints is an important factor for routing problems and is generally studied under Capacitated VRP (CVRP) (Golden et al., 2008). The main assumption of the CVRP is that a vehicle has a limited weight (or volume) capacity to ship goods from an origin to a destination location. In multi-compartment VRPs, a vehicle can have more than one compartment and all compartments might have different features (i.e., weight capacity, volume capacity, temperature-controlled environment etc.). This makes the VRP even more challenging and practical at the same time, especially for city logistics.

The multi-compartment VRP (VRPMC or MCVRP) was first studied as a special variant of VRP by Christofides et al. (1979). The authors highlighted several examples for a delivery vehicle which has two distinct compartments for food transportation, namely refrigerated and non-refrigerated compartments, and a tanker which transports different types of liquid products in separate compartments. In 2008,

El Fallahi et al. (2008) studied a transportation problem to deliver various products to customers by a fleet of identical vehicles, each with limited capacity. The authors used memetic and tabu search algorithms for the solution of the problem. In 2010, Muyldermans and Pang (2010) investigated the advantages of collaborative collection by multi-compartment vehicles and proposed a Guided Local Search metaheuristic with various local search operators.

From the environmental perspective, Tassou et al. (2009) investigated road food transport refrigeration compartments and estimated their environmental impacts. They also investigated alternative technologies for reducing energy consumption and emissions. Recently, Rai and Tassou (2017) compared the use of a newer refrigeration system instead of a diesel engine for temperature control. They considered two types of products, three delivery schedules and two types of vehicles, and analyzed the emissions and cost of each technology. In 2018, Stellingwerf et al. (2018a) introduced a temperature-controlled load-dependent VRP (TCDVRP) model to optimize vehicle routes and consider refrigeration emissions in the VRP domain. The authors used their model for small numerical instances and compared the results of three models. In a related study, Stellingwerf et al. (2018b) studied the environmental and economic benefits of cooperation between different stake holders in a supply chain. The authors tested their approach on a case study of cooperation between a number of supermarket chains in the Netherlands, and analyzed the data of the case study to quantify both the economic and environmental benefits of implementing cooperation.

In another study, Silvestrin and Ritt (2017) proposed an iterated tabu search algorithm for the MCVRP. The authors also studied the effect of split demands. Ostermeier and Hübner (2018) proposed a model and Large Neighborhood Search (LNS) algorithm for MCVRPs considering mixed fleets involving single- and multi-compartment vehicles. The authors considered different fixed and operational costs for each vehicle and tested their methodology on a real-life case. Their computational results showed that a mixed fleet could reduce costs by up to 30%. Later, Alinaghian and Shokouhi (2018) introduced a mathematical model and hybrid algorithm composed of adaptive LNS and variable neighborhood search (VNS) for multi-depot multi-compartment VRPs.

3. Problem definition

The investigated problem can be formulated on a complete directed graph $G = (\mathcal{N}, \mathcal{V})$ with $\mathcal{N} = \{0, 1, 2, .., n\}$ as the set of nodes with node 0 considered as the only depot. Set $\mathcal{V} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$ shows all arcs and the distance from node i to node j is presented as d_{ij}. Set $\mathcal{N}_c = \{\mathcal{N} \cup \{0\}\}$ is used to separate the customer set from the depot.

Each customer order includes a set of products which might need different transportation requirements to be transported (i.e., multi-compartment) (Tassou et al., 2010). Set $C = \{1, 2, .., c\}$ represents different vehicle compartments. The products can be categorized as refrigerated or non-refrigerated products. For the refrigerated products, a certain level of temperature should be maintained during the whole journey. For example, whereas shelf-stable (ambient) products can be safely stored at room temperature in a vehicle compartment, chilled products require the temperature to be below 8 °C and frozen products require a minimum storage temperature of -18 °C or colder at all times. In our research, we set $c = 3$, which means only three separate compartments (e.g., ambient, chilled, frozen) is considered. All type of products can be carried in one of these three compartments.

In our problem setting, the vehicle space consists of a number of compartments with individual and independent temperature values (e.g., –18 °C to +18 °C) which provide logistical flexibility for transporting various types of products. We also assume that each compartment requires energy to maintain the temperature for the total number of delivery crates inside a compartment. This means that the temperature-related costs increase with the number of crates carried inside the compartment. The number of multi-compartment homogeneous vehicles (i.e., city van) is a deterministic parameter and a set of vehicles is denoted by $K = \{1, 2, .., m\}$. Each customer order is defined based on its demand (number of crate(s)) in each compartment c) ${q_i}^c$ and weight (payload) ${p_i}^c$. The volume capacity of a vehicle for

each type of compartment c is denoted as Q^c (i.e., available space in each compartment). Moreover, we assume hard time windows for satisfying customer deliveries. We also note that split delivery is not considered in our problem setting.

In this city logistics application, customer profile is defined based on order arrival time and preferred delivery service. The first group of customers must order before 24:00 on the previous day (i.e., for Tuesday delivery the order should be received on Monday). These customers can pick, on the following day, any 1–hour time slot or any 4–hour period (i.e, Period 1, Period 2, or Period 3) or do not declare their preferences and are served based on the generated plan. The second group includes orders coming on the execution day between 24:01 and 12:00 noon. These customers can be served in any possible planning period with 1–hour time slot or 4–hour period preferences. The last group requests (i.e., same day customers) are the ones which arrive on the execution day and wish to be considered in the first planning period. These customers can also select any 1–hour time slot in the next available planning period. The planning cut-off times of these periods are defined as 06:00 for the first period, 12:00 for the second period and 18:00 for the third time period. Two hours difference from the dispatch time of vehicles ensures sufficient time for planning of the vehicle routes and preparation of the orders (i.e., loading crates to vehicles).

To calculate daily operational costs, we consider the fixed vehicle costs and the variable costs such as fuel consumption, wage of driver(s) and temperature running costs for keeping the temperature at desired levels. An instantaneous fuel consumption rate is calculated with the CMEM introduced by Barth et al. (2005). More specifically, CMEM follows, to some extent, the basic engine model of Ross (1994). The CMEM is a microscopic model which requires a set of parameters related to vehicle characteristics, environment and traffic conditions. Especially, vehicle-related parameters can easily be adjusted for different types of vehicles. For this reason, we have used the CMEM model in our research. Alternatively, up-to-date macroscopic models could also be used for a better estimation of energy consumption, and emissions in particular. Such models can be found in the review of (Demir et al., 2011) and (Demir et al., 2014b). We now briefly present the model as the following:

$$Fuel = \lambda \left(k_e N_e V_e + W\gamma\alpha \, speed(v) + \gamma\alpha \, load(f) \, speed(v) + \beta\gamma \, speed(v)^3 \right) distance(d)/speed(v), \tag{1}$$

where τ,θ, and W denote acceleration, road gradient, and curb weight. Moreover, $\lambda = \xi/(\kappa\psi)$ and $\gamma = 1/(1,000 n_{tf}\eta)$ are defined as constants. Furthermore, $\alpha = \tau + g\sin\theta + gC_r\cos\theta$ and $\beta = 0.5 C_d \rho A$ are defined as vehicle-specific constants. The fuel and CO_2equivalent emissions-related costs can be calculated as $Total\,Cost = Fuel * f_c$, where f_c is the sum of fuel and CO_2e emissions costs. Finally, we note that $speed(v)$ is the speed of a vehicle, $distance(d)$ is the distance, and finally $load(f)$ is the total payload carried. All parameters are provided in Table 1.

We note that some of the values used in Table 1 are from the literature and the others are calculated for a chosen specific city van which is applicable to our problem setting.

3.1. Mathematical formulation

We now introduce a mixed-integer linear programming model for the investigated problem. Since the fuel consumption function introduced in the previous section is non-linear, we discretize the speed variable by using a speed level r transformation to lineralize the fuel and travel time function as done in Bektaş and Laporte (2011). Sets and parameters are defined in Table 2 whereas decision variables are provided in Table 3.

The mathematical model of the investigated problem is shown below:

Table 1
A list of parameters used in the fuel consumption model and the objective function:

Symbol	Meaning	Chosen value	Reference
W	Empty vehicle weight (kilograms)	2,300	The car manual[a]
P	Payload (kilograms)	1,200	The car manual
ξ	Fuel rate	1	Demir et al. (2011)
k_e	Diesel engine friction factor (kJ/rev/liter)	0.23	Demir et al. (2011)
N_e	Diesel engine speed (rev/second)	35	Demir et al. (2011)
V_e	Diesel engine displacement (liters)	3	Demir et al. (2011)
g	Gravitational constant (meter/second^2)	9.81	Bektaş and Laporte (2011)
C_d	Aerodynamic drag coefficient	0.32	Bektaş and Laporte (2011)
ρ	Air density coefficient (kilogram/meter^3)	1.2041	Bektaş and Laporte (2011)
A	The total frontal surface (meter^2)	5	The car manual
C_r	Rolling resistance coefficient	0.01	Demir et al. (2011)
n_{tf}	Drive train efficiency	0.4	Demir et al. (2014a)
η	Diesel engine efficiency constant	0.9	Demir et al. (2011)
κ	Heating value (kJ/gram)	45	Demir et al. (2014a)
ψ	Conversion factor (gram/second to liter/second)	737	Demir et al. (2014a)
v^l	Lowest speed limit (meter/second)	11.1	Demir et al. (2014a)
v^u	Upper speed limit (meter/second)	22.22	Demir et al. (2014a)
f_c	Fuel cost per liter (pounds)	1.2	Approximate value
f_d	Driver wage per second (pounds)	0.0022	Approximate value
f_f	Fixed cost of vehicle (pounds)	20	Approximate value
tc^1	Temperature cost to keep a product between 16°C and 20°C (per crate, pounds)	0.01	Approximate value
tc^2	Temperature cost to keep a product between 0°C and 8°C (per crate, pounds)	0.05	Approximate value
tc^3	Temperature cost to keep a product below –18°C (per crate, pounds)	0.1	Approximate value

[a]Vehicle-related parameters are taken from the car manual of a city van.

Table 2
Sets and the rest of parameters used in the model:

Notation	Description
$\mathcal{N} = \{0, 1, 2, .., n\}$	Set of nodes which include all customer locations and the depot
$\mathcal{V}$	Set of arcs $\{(i, j) : i, j \in \mathcal{N}, i \neq j\}$
$\mathcal{N}_c$	Set of customer nodes $\{\mathcal{N} \cup \{0\}\}$
$\mathcal{T} = \{1, 2, 3\}$	Time periods
$\mathcal{R} = \{0, 1, 2, .., R\}$	Speed level $r \in \mathcal{R}$
d_{ij}	Distance between nodes i and j
$K = \{1, 2, .., m\}$	Set of homogeneous vehicles
$C = \{1, 2, 3\}$	Compartment types
$q_i{}^c$	For customer i, the required number of crates which need to be transported in compartment c
$p_i{}^c$	For customer i, the weight of products which require to be transported in compartment c
Q^c	Available space in compartment c
$s^t \in \{08{:}00, 14{:}00, 20{:}00\}$	The start (dispatch) time of each distribution period
$p^t \in \{06{:}00, 12{:}00, 18{:}00\}$	Cut-off planning time
$[a_i, b_i]$	Customer time window
$\tilde{}$	Uncertainty parameter
$\bar{}$	Nominal value
$\hat{}$	Uncertain part of uncertain parameter
$\tilde{t}_i$	Uncertain service time for each customer $i \in \mathcal{N}_c$
$\bar{t}_i$	Customer service time with nominal value
$\hat{t}_i$	Uncertain part of service time
$\hat{tt}_{ij}$	Uncertain travel time on an arc (i, j)
$[\bar{t}_i - \hat{t}_i, \bar{t}_i + \hat{t}_i]$	Uncertainty interval
m^t	Available number of vehicles at time period t

$$\text{minimize} \quad \sum_{(i,j)\in\mathcal{N}} f_c k_e N_e V_e \lambda d_{ij} \sum_{r\in\mathcal{R}} z_{ij}^r / \bar{v}^r + \sum_{(i,j)\in\mathcal{N}} f_c W \gamma \lambda \alpha_{ij} d_{ij} \sum_{t\in\mathcal{T}} x_{ij}^t \tag{2}$$

$$+ \sum_{(i,j)\in\mathcal{N}} f_c \gamma \lambda \alpha_{ij} d_{ij} e_{ij} + \sum_{(i,j)\in\mathcal{N}} f_c \beta \gamma \lambda d_{ij} \sum_{r\in\mathcal{R}} z_{ij}^r (\bar{v}^r)^2 + \sum_{(i,j)\in\mathcal{N}} \sum_{c\in\mathcal{C}} tc^c f_{ij}^c \tag{3}$$

$$+ \sum_{i\in\mathcal{N}} f_d s_i + \sum_{t\in\mathcal{T}} u^t f_f \qquad 6 \tag{4}$$

Table 3
Decision variables used in the model:

Notation	Description
x_{ij}^t	Binary variable - 1 if an arc is traveled by a vehicle, and 0 otherwise
f_{ij}^c	Continuous variable - the total number of crates in compartment on arc
e_{ij}	Continuous variable - the total payload on arc
y_j	Continuous variable - the start time at node $j \in \mathcal{N}_c$
uv^t	Integer variable - the number of vehicles used at time period t
z_{ij}^r	Binary variable - a speed level r defined on each arc (i, j)

subject to

$$\sum_{i\in\mathcal{N}} x_{0i}^t \le m^t \qquad \forall t \in \mathcal{T} \tag{5}$$

$$\sum_{i\in\mathcal{N}} x_{0i}^t = uv^t \qquad \forall t \in \mathcal{T} \tag{6}$$

$$\sum_{j\in\mathcal{N}}\sum_{t\in\mathcal{T}} x_{ij}^t = 1 \qquad \forall i \in \mathcal{N}_c \tag{7}$$

$$\sum_{j\in\mathcal{N}}\sum_{t\in\mathcal{T}} x_{ji}^t = 1 \qquad \forall i \in \mathcal{N}_c \tag{8}$$

$$\sum_{j\in\mathcal{N}}\sum_{t\in\mathcal{T}} x_{ij}^t - \sum_{j\in\mathcal{N}}\sum_{t\in\mathcal{T}} x_{ji}^t = 0 \qquad \forall i \in \mathcal{N}_c \tag{9}$$

$$\sum_{j\in\mathcal{N}} f_{ji}^c - \sum_{j\in\mathcal{N}} f_{ij}^c = q_i^c \qquad \forall i \in \mathcal{N}_c, \forall c \in C \tag{10}$$

$$q_j^c \sum_{t\in\mathcal{T}} x_{ij}^t \le f_{ij}^c \le (Q^c - q_i^c)\sum_{t\in\mathcal{T}} x_{ij}^t \qquad \forall (i,j) \in \mathcal{V}, \forall c \in C \tag{11}$$

$$\sum_{j\in\mathcal{N}} e_{ji} - \sum_{j\in\mathcal{N}} e_{ij} = \sum_{c\in C} p_i^c \qquad \forall i \in \mathcal{N}_c \tag{12}$$

$$\sum_{c\in C} p_c^j \sum_{t\in\mathcal{T}} x_{ij}^t \le e_{ij} \le (P - \sum_{c\in C} p_i^c)\sum_{t\in\mathcal{T}} x_{ij}^t \qquad \forall (i,j) \in \mathcal{V} \tag{13}$$

$$y_0 + \sum_{t\in\mathcal{T}} x_{0j}^t(\bar{s}^t - \bar{s}^0) - y_j + t_0 + \sum_{r\in\mathcal{R}} d_{0j} z_{0j}^r/\bar{v}^r \le K_{0j}(1 - \sum_{t\in\mathcal{T}} x_{0j}^t) \qquad \forall j \in \mathcal{N}_c \tag{14}$$

$$y_i - y_j + t_i + \sum_{r\in\mathcal{R}} d_{0j} z_{ij}^r/\bar{v}^r \le K_{ij}(1 - \sum_{t\in\mathcal{T}} x_{ij}^t) \qquad \forall i \in \mathcal{N}, \forall j \in \mathcal{N}_c \tag{15}$$

$$y_j - s_j - \sum_{t\in\mathcal{T}} x_{j0}^t \bar{s}^t + t_j + \sum_{r\in\mathcal{R}} d_{j0} z_{j0}^r/\bar{v}^r \le L(1 - \sum_{t\in\mathcal{T}} x_{j0}^t) \qquad \forall j \in \mathcal{N}_c \tag{16}$$

$$a_j \le y_j \le b_j \qquad \forall j \in \mathcal{N}_c \tag{17}$$

$$\sum_{r\in\mathcal{R}} z_{ij}^r - \sum_{t\in\mathcal{T}} x_{ij}^t = 0 \qquad \forall (i,j) \in \mathcal{V} \tag{18}$$

$$x_{ij}^t \in \{0,1\} \qquad \forall (i,j) \in \mathcal{V}, \forall t \in \mathcal{T} \tag{19}$$

$$z_{ij}^r \in \{0,1\} \qquad \forall (i,j) \in \mathcal{V}, \forall r \in \mathcal{R} \tag{20}$$

$$f_{ij}^c \ge 0 \qquad \forall (i,j) \in \mathcal{V}, \forall c \in C \tag{21}$$

$$e_{ij} \ge 0 \qquad \forall (i,j) \in \mathcal{V} \tag{22}$$

$$y_i, s_i \ge 0 \qquad \forall i \in \mathcal{N}_c \tag{23}$$

$$uv^t \ge \{0,1\} \qquad \forall t \in \mathcal{T}. \tag{24}$$

The objective cost (2)–(4) consists of fixed and variable (operational) vehicle costs. Fixed costs include the vehicle capital cost, insurance cost, excise duty cost and maintenance cost and represented by f_f. Variable costs are related to fuel, driver wage and energy costs to keep the temperature at a preset value during the whole journey. The objective function (2)–(3) is defined for fuel consumption. The first part of objective (4) defines the total wage of drivers. The second component is for the total energy costs required for multiple compartments. The last component of the objective function defines the fixed vehicle costs. Constraints (5) state that the number of vehicles used in a solution cannot be more than the available number of vehicles in each time period t. Constraints (6) define the number of vehicles used in each time period t. Constraints (7)–(9) are the assignment constraints required for visiting each and every customer in the right time period. Constraints (10) and (13) are used for arc flows. These constraints ensure that compartment capacity in terms of crates and weight is not violated. Constraints (14) and (15), where $K_{ij} = \max\{0, b_i + t_i + d_{ij}/v^l - a_j\}$, impose time window preferences. These constrains accumulate the total travel time from the depot to the last customer of a route. Constraints (16) defines the total driving time. These constraints are used to calculate the total traveling time for the whole journey. The lower and upper bounds of each time window are ensured in constraints (17). Constraints (18) limit to a single speed level on an arc in each time period t and $z^r_{ij} = 1$ if $x^t_{ij} = 1$. Constraints (19)–(24) are used to describe non-negativity conditions.

3.1.1. Robust modeling formulation

In order to consider time uncertainty of the investigated problem, we also provide the required changes with the well-known Hard Worst case approach proposed by Ben-Tal and Nemirovski (1998). Travel time uncertainty is caused by many different factors such as traffic, weather situations, accident or vehicle defect. Similarly, service time uncertainty can be caused by wasted time finding an address, or delayed customer response. In response to these uncertainties, we adapt the following constraints (instead of constraints (15)–(17)) as below:

$$y_0 + \sum_{t\in\mathcal{T}} x^t_{0j}(\bar{s}^t - \bar{s}^0) - y_j + \bar{t}_0 + \hat{t}_0 + \hat{t}t_{ij} + \sum_{r\in\mathcal{R}} d_{0j} z^r_{0j}/\bar{v}^r \le K_{0j}(1 - \sum_{t\in\mathcal{T}} x^t_{0j}) \qquad \forall j \in \mathcal{N}_c \tag{25}$$

$$y_i - y_j + \bar{t}_i + \hat{t}_i + \hat{t}t_{ij} + \sum_{r\in\mathcal{R}} d_{0j} z^r_{ij}/\bar{v}^r \le K_{ij}(1 - \sum_{t\in\mathcal{T}} x^t_{ij}) \qquad \forall i \in \mathcal{N}, \forall j \in \mathcal{N}_c \tag{26}$$

$$y_j - s_j - \sum_{t\in\mathcal{T}} x^t_{j0}\bar{s}^t + \hat{t}t_{ij} + \bar{t}_j + \hat{t}_j + \sum_{r\in\mathcal{R}} d_{j0} z^r_{j0}/\bar{v}^r \le L(1 - \sum_{t\in\mathcal{T}} x^t_{j0}) \qquad \forall j \in \mathcal{N}_c \tag{27}$$

Our problem is NP-hard and only small-sized instances can be solved to optimality. In order to solve larger-sized problem instances, we have proposed a metaheuristic algorithm.

4. Solution methodology

In this section, we present the solution methodology, namely ALNS algorithm. ALNS algorithm was introduced by Ropke and Pisinger (2006) and has been widely used for many variants of the VRP. ALNS includes many destroy and repair neighborhood structures (operators) to obtain a good-quality solution and is a relatively fast algorithm, compared to other well-known metaheuristics algorithms.

4.1. A brief description of ALNS algorithm

To solve the investigated problem, we propose the enhanced ALNS algorithm proposed by Demir et al. (2012). The steps of ALNS algorithm are provided in Algorithm 1.

Some notations used in the algorithm are discussed first. Variable S^{best} is used for the best solution. This solution is also the main output of the algorithm. Variable $S^{current}$ refers to the current solution used at the start of an iteration. Variable S^{new} is used for a temporary solution obtained at the end of an iteration. The cost of a solution S is represented by $cost(S)$. A new (feasible) solution S^{new} is accepted if

$cost(S^{new}) < cost(S^{current})$, and can be accepted with a probability of $e^{-(cost(S^{new})-cost(S^{current}))/\Delta}$ if $cost(S^{new}) > cost(S^{current})$, where Δ is the temperature.

The initial temperature is set at $cost(S_{initial})P_{initial}$ where $cost(S_{initial})$ is the first feasible objective function value obtained with the Clarke and Wright (C&W) algorithm (Clarke and Wright, 1964). We note that parameter $P_{initial}$ is an initialization constant used at the beginning of the algorithm. The current temperature at each iteration is gradually reduced as $ch\Delta$, where $0 < ch < 1$ is a constant as in Simulated Annealing (SA) algorithm. We note that SA is widely used in ALNS algorithms (Ropke and Pisinger, 2006). We also refer interested readers to Aarts et al. (2005) for an overview of SA algorithms.

We have modified the C&W algorithm as follows: First, we calculate the savings as $s(i, j) = d_{0i}+d_{0j}-d_{ij}$ for every pair (i, j) of locations. Then, we rank the calculated savings $s(i, j)$ and list them in descending order of magnitude. For the savings $s(i, j)$, we include an arc (i, j) in a route if no side constraints are violated through the inclusion of arc (i, j). We also look at the following conditions: (i) The two delivery locations are not already on the same route, (ii) nodes are not interior to their routes, meaning that both nodes can directly be connected to the depot, (iii) the capacity of each compartment and time window constraints are not violated by the merged route. Finally, if the savings list $s(i, j)$ has not been emptied, we return to the previous step and continue with the next single largest saving $s(i, j)$. If the list is empty, the C&W algorithm stops.

Algorithm 1: An enhanced ALNS algorithm

Input : Instance data; The parameters of ALNS, including sets of removal (RO) and insertion (IO) operators, variables Δ and temperature T
Output: S^{best}

Create a feasible initial solution using C&W savings heuristic (Clarke and Wright, 1964)
Initialize probability for Π_r^t, $r \in RO$
Initialize probability for Π_i^t, $i \in IO$
Set the values of Δ and iteration number T
Set the initial values of variables $S^{current} = S^{best} = S^{initial}$
repeat
  Apply a removal operator $r^* \in RO$, considering Π_r^t
  S^{new} becomes the new partial solution after applying operator r^* to $S^{current}$
  Apply an insertion operator $i^* \in IO$, considering Π_i^t
  S^{new} is the new temporary but feasible solution
  if *$cost(S^{new}) < cost(S^{current})$* **then**
    $S^{current} = S_{new}$
    Apply a local search to improve the current solution
  else
    Let $v = e^{-(cost(S^{new})-cost(S^{current}))/\Delta}$
  Pick a random number $\epsilon \in [0, 1]$
  if $\epsilon < v$ **then**
    $S^{current} = S^{new}$
    Apply a local search to improve the current solution
  if *$cost(S^{current}) < cost(S^{best})$* **then**
    $S^{best} = S^{new}$
  Update the temperature $\Delta \leftarrow ch\,\Delta$
  Calculate new scores for removal operators
  Calculate new scores for insertion operators $T \leftarrow T + 1$
until *The algorithm has been concluded.*

The removal and insertion operators are chosen based on a roulette-wheel procedure. At the start of ALNS algorithm, all operators are assigned with the same probability. A probability of an operator r at iteration t is represented as Π_r^t. The probability of each operator r is updated as $\Pi_r^{t+1} = \Pi_r^t\,(1-r^p) + r^p\,\pi_i/\omega_i$, where r^p is a constant, π_i is the score and ω_i is the number of times that the operator was called in the last N_w iterations. If a new best solution is obtained, the score is raised by σ_1. If the new solution is better

than the current one, the total score is raised by σ_2. If the new solution is not better than the current one but it is still accepted, the total score is raised by σ_3.

In order to improve the solution quality at each iteration, we have applied inter 2-opt algorithm on the current solution. This is done due to the fact that ALNS requires further intensification around the solution found at each iteration. At the end of the algorithm, the algorithm stops when the maximum number of iterations (i.e.,25K) has been reached.

4.2. Neighborhood structures

This section discusses the ten implemented removal operators. In order to reflect the characteristics of the investigated problem, these operators are either adapted from the literature (see e.g., Ropke and Pisinger (2006), Shaw (1998), Demir et al. (2012)), or newly proposed.

4.2.1. A list of removal operators

1. **Random (R1)**: This operator randomly selects s nodes (customers) and removes them from the current solution. Any customer node can be chosen randomly by this operator. Making a random selection of nodes helps diversify the solution search space and might lead to better solutions in the subsequent iterations of the algorithm.
2. **Worst fuel consumption (R2)**: The R2 operator removes customers with higher fuel consumption costs. The total fuel cost is calculated as the sum of fuel costs from the previous customer to a subsequent customer. For example, the operator selects node $j^* = \underset{j \in N}{argmax}\{|f_{ij} + f_{jk}|\}$ each time. This operator helps by removing the distant customer or the one with a higher demand. Eventually, this operator leads to a lower amount of fuel consumption in the solution.
3. **Worst time (R3)**: This operator calculates the time difference between the start time from the depot and a lower bound of the time window a_j. The operator selects the node with the largest difference in each iteration. This operator is especially important from a cost perspective for temperature-controlled compartments. This is due to the fact that total travel time will increase the wage of driver and the costs required to keep the temperature at desired levels for compartments. Mathematically, the operator calculates $j^* = \underset{j \in N}{argmax}\{|y_j - a_j|\}$.
4. **Complete route (R4)**: This standard operator is used to empty the complete route(s). The operator randomly selects a route from the set of routes in the solution. The operator continues selecting a node j from the chosen route until all nodes are removed. Depending on the number of nodes in the chosen route, this operator can empty more than one route.
5. **Shaw (R5)**: This operator is similar to the mechanism proposed by Shaw (1998). The operator removes related customers. It removes a node based on a formula $j^* = \underset{j \in N}{argmax}\{\Phi_1 f_{ij} + \Phi_2|a_i - a_j| + \Phi_3 l_{ij} + \Phi_4|q_i^c - q_j^c|\}$, where Φ_1–Φ_4 are weights and l_{ij} is equal to 1 if $i, j \in \mathcal{N}$ are included in the same route.
6. **Proximity (R6)**: This operator is similar to the previous one where $\Phi_1 = 1$, and other Φ values are zero. More specifically, it removes a node that requires a similar amount of fuel to be visited.
7. **Time (R7)**: This operator is also similar to $R5$ with $\Phi_2 = 1$, and other Φ values being equal to zero.
8. **Demand (R8)**: The last removal operator is also similar to $R5$ with $\Phi_4 = 1$ and other Φ values being equal to zero.
9. **Compartment (R9)**: This operator removes customers that have the highest demand in the most restricted compartments in all routes. We note that the type of compartment that is restricted in each route may be different, and this operator calculates this value for each node based on its route capacity.
10. **Time slot (R10)**: This operator removes customers with the largest time window period (l_i–e_i). If the time window period is equal for some nodes, the node with the smallest number of crates is chosen.

4.2.2. A list of insertion operators

We now present six insertion operators suggested for the ALNS algorithm. During the insertion process, a feasibility of the solution must be ensured. For example, the capacity of each compartment and time windows of a customer must be ensured. At the end of this stage, all customers must be visited and all constraints must be satisfied.

1. **Greedy (basic) (I1)**: The I1 operator sequentially reinserts a removed node in the best position in terms of fuel consumption. The cost of insertion is calculated as $f_i = f_{ji} + f_{ik} - f_{jk}$ for $j = 1, ..., n$ and $i = 1, ..., n$.
2. **Regret (I2)**: The I2 operator considers a regret criterion defined in the literature. Let Δf_i define the change in objective value by reinserting a node i into its best and second-best position with respect to fuel consumption f_i. Let $i^* = \underset{i \in L}{argmax} \{\Delta f_{i2} - \Delta f_{i1}\}$, where Δf_{i1} is the minimum fuel consumption increase and Δf_{i2} refers to the second-lowest fuel consumption increase.
3. **Greedy with noise (I3)**: The I3 operator considers a degree of freedom in deciding the best location for a removed node. The new cost is calculated with the following formula for node i: $C = C + \bar{d}\,\mu\,\epsilon$ where $\bar{d}$ is the maximum distance, μ is the noise constant, and ϵ is the random number.
4. **Regret with noise (I4)**: This operator is similar to the previous operator but considers the same noise function in the $I2$ operator instead.
5. **Period (I5)**: The $I5$ operator considers the time windows' characteristics. It finds the best time period for each node and tries to come up with a better delivery plan, considering the time flexibility given by a customer.
6. **Flexibility (I6)**: This operator calculates the flexibility score for each node and reinserts the removed nodes by considering the order of lower scores. We define three indexes for this operator as $Score = W1 * wf_1 + W2 * wf_2 + W3 * wf_3$, where wf_1 is the number of possible feasible insertions divided by total number of insertions in current routes. wf_2 is the time window divided by 16, and wf_3 is the one divided by the total number of crates in the current route. We also note that $W1 + W2 + W3 = 1$.

5. Computational study

In this section, we present the computational results of experiments on both benchmark and newly generated instances.

5.1. Data setting and problem characteristics

To analyze the features of the investigated problem, we have generated eight sets of ten instances with different characteristics. The instances range from ten to 200 nodes (excluding depot). Each instance represents randomly selected postcodes located in London (United Kingdom) and is denoted by *LonA–B*, where *A* is the total number of customers in an instance and *B* is the number of an instance (i.e., from 01 to 10). Time period preferences, demands and time slot preferences are randomly generated for each customer. We assume that customers' orders are received by the planning system independently and the algorithm only considers the orders that had arrived by the cut-off time in each period.

In the generated instances, customers are categorized according to their preferences within their order time frame. There are three time periods used as delivery time periods. These include the first period (08:00–12:00), the second period (14:00–18:00), and the third period (20:00–24:00). The cut-off times for these time periods are 06:00 for the first period deliveries, 12:00 noon for the second period deliveries, and 18:00 for the third period deliveries. It is assumed that each customer who ordered before 24:00 in the evening is promised delivery in the following day. These customers have three options: (*i*) selecting a 1–hour time slot in any time period, (*ii*) selecting a 4–hour time period (i.e., Period 1, Period 2, or Period 3), and (*iii*) or do not specify any time slots and will be served anytime during the day.

The second customer group can place a delivery order by 12:00 noon on the planning day. We assume that these customers can either select a 1–hour time slot or a 4-hour period, considering the availability of time periods to be served. The third customer group is promised fast delivery; these customers can order by 4pm in the afternoon and are assigned either a 1–hour time slot or do not provide a preference.

In our newly generated instances, we assume that 60 percent of the customers have opted for the following day, 30 percent for the same day and 10 percent for fast delivery, as shown in Table 4. This table also shows time windows (delivery slot) preference probabilities used (i.e., 1-hour time slot, time period) for each customer type.

Table 4
Data characteristics - I: Customers' time window preferences:

Customer type	Ratio (%)	1-hour time slot preference (%)	Time period preference (%)	No preference (%)
Type I	60	60	30	10
Type II	30	80	0	20
Type III	10	90	0	10

Another important feature of our study is the use of multi-compartment vehicles. This requirement is an important factor in city logistics and many logistics companies tend to convert their vehicles to be equipped with flexible or fixed multi-compartment features. In this regard, we generated our instances based on three categories widely used in practice. This classification is especially valid for perishable product transportation, such as food, meat and poultry, flower, and dairy industries. In this research, we refer to these compartments as normal temperature-, low temperature- and very low temperature-controlled compartments. Each compartment has a specific volume and weight capacity. Also, the maximum payload that a vehicle can carry in all compartments is set to 1,200 kilograms. Customers' order demands are shown in Table 5.

Table 5
Data characteristics - II: Customers' demands:

Customer type	The number of delivery crates per customer					
	Normal		Low		Very low	
	Lower bound	Upper bound	Lower bound	Upper bound	Lower bound	Upper bound
Type I	0	4	0	3	0	2
Type II	0	3	0	2	0	2
Type III	0	2	0	2	0	2

As shown in table, each customer can order a different number of products from each of three categories defined by the LSP. We assume that these orders are carried in crates, ranging from zero to four crates (maximum). All orders are carried within crates from the depot to each customer location. The compartments' capacities for normal, low and very low temperatures are set to 30, 30 and 20 delivery crates. We also assume that each crate weighs between five and 15 kilograms and the weight of each crate is randomly decided within the same interval. Service times of the customers are decided based on a weighted sum of fixed (i.e., 5 minutes) and variable times (30 seconds for each crate). We use the fuel consumption parameters provided in Table 1. Moreover, the algorithm is implemented with C programming language and run on a personal computer with a Core i5 4200u 1.68GHz CPU and 8GB of RAM.

5.2. Analysis of the parameters used in the enhanced ALNS metaheuristic algorithm

The implementation of ALNS algorithm requires 15 parameters and they are provided in Table 6. Parameters are divided into three groups as done in Demir et al. (2012) and Franceschetti et al. (2017). Group (*i*) presents the parameters that control the roulette wheel and the selection of the operators. Group (*ii*) presents the parameters that control ALNS algorithm. This is especially important for the diversification of the solution space. Lastly, group (*iii*) provides all parameters used in operators. We note that we have focused on certain parameters used in ALNS algorithm. These investigated operators have big impact either on the solution quality or solution times.

Table 6
A list of parameters used in ALNS algorithm :

Group	Symbol	Meaning	Chosen value
(i)	r^p	Roulette wheel parameter	0.1
	N_w	Number of iterations needed for the roulette wheel mechanism	100
	σ_1	Score when a new best solution is found	10
	σ_2	Score when a new solution is accepted (better than a current solution)	5
	σ_3	Score when a new solution is accepted (worse than a current solution)	1
(ii)	N	Total number of iterations	25,000
	$P_{initial}$	Starting temperature parameter	100
	ch	Cooling rate	0.999
(iii)	γ	The upper bound for nodes to be removed	$\lfloor \log_{10}(N) \rfloor$
	$\underline{\gamma}$	The lower bound for nodes to be removed	$\lfloor \log_{1.4}(N) \rfloor$
	ϕ_1	First Shaw parameter (i.e., fuel consumption)	0.5 or 1
	ϕ_2	Second Shaw parameter (i.e., time windows)	0.25 or 1
	ϕ_3	Third Shaw parameter (i.e., being on the same route)	0.15
	ϕ_4	Fourth Shaw parameter (i.e., demand)	0.25 or 1
	μ	Noise parameter	0.1

To have the best set of values for the parameters used in ALNS, we conducted parameter-tuning as a starting point. All initial experiments are completed on a tuning set including 16 instances out of 80 (i.e., Lon10–01 & 02, Lon15–01 & 02, Lon20–01 & 02, Lon25–01 & 02, Lon50–01 & 02, Lon75–01 & 02 Lon100–01 & 02, and Lon200–01 & 02). For each parameter, we investigated multiple sets of values for each tested instance and applied our ALNS algorithm to the tuning set of instances three times. The outcome of the analysis is discussed in the following subsections. In order to guarantee comparison fairness for the same instance, we have started with the same initial solution obtained with the C&W savings algorithm in all runs.

It is important to note that one of the biggest advantages of ALNS is to allow diversification of the solutions over all iterations. We have observed that the algorithm was able to find the best solution after 22,000 iterations due to the effective use of simulated annealing search framework.

5.3. The roulette wheel-related parameters

For the Group (i) parameters, i.e., σ_1, σ_2 and σ_3, we have conducted numerical tests on the tuning set instances. We considered the following four combination for $(\sigma_1, \sigma_2, \sigma_3)$: (*i*) $(10, 5, 1)$, (*ii*) $(10, 1, 5)$, (*iii*) $(10, 10, 10)$ and (*iv*) $(1, 5, 10)$. The comparison of sigma values is provided in Table 7 below. $Average(l)$ gives average objective function value out of three runs for each combination. $Dev(l)$ shows the percentage deviation with regards to the smallest average solution. This value is calculated as follows $Dev(l) = (Average(l) - Average(l))/Avg(Best(l)$, where Avg is the smallest average value found out of all runs in all combinations.

In Table 7, for each instance, we calculated the deviation between the average of all runs and the average found in the current set of parameters. We can see that combination $(10, 5, 1)$ performs the best

Table 7
Parametric analysis for the roulette wheel $(\sigma_1, \sigma_2, \sigma_3)$:

Instance	$(\sigma_1, \sigma_2, \sigma_3) = (10, 5, 1)$		$(\sigma_1, \sigma_2, \sigma_3) = (10, 1, 5)$		$(\sigma_1, \sigma_2, \sigma_3) = (10, 10, 10)$		$(\sigma_1, \sigma_2, \sigma_3) = (1, 5, 10)$		Avg
	Average	*Dev*	*Average*	*Dev*	*Average*	*Dev*	*Average*	*Dev*	
Lon10–01	155.5	0.6	155.3	0.5	154.5	0.0	155.0	0.3	154.5
Lon10–02	181.9	0.1	182.1	0.2	182.4	0.3	181.8	0.0	181.8
Lon15–01	167.8	0.2	167.7	0.2	167.4	0.0	167.9	0.3	167.4
Lon15–02	157.9	0.0	157.9	0.1	158.1	0.2	157.8	0.0	157.8
Lon20–01	200.2	0.2	200.4	0.3	199.8	0.0	200.3	0.3	199.8
Lon20–02	196.0	0.0	196.4	0.2	196.6	0.3	196.5	0.3	196.0
Lon25–01	234.9	0.3	234.7	0.2	234.1	0.0	234.4	0.1	234.1
Lon25–02	247.2	0.2	246.6	0.0	247.3	0.3	246.7	0.0	246.6
Lon50–01	371.7	0.0	372.7	0.3	372.5	0.2	373.4	0.5	371.7
Lon50–02	368.3	0.1	368.3	0.1	367.9	0.0	368.4	0.1	367.9
Lon75–01	487.1	0.0	487.6	0.1	487.2	0.0	488.5	0.3	487.1
Lon75–02	492.5	0.0	492.3	0.0	495.4	0.6	494.9	0.5	492.3
Lon100–01	633.9	0.0	635.9	0.3	635.9	0.3	636.3	0.4	633.9
Lon100–02	635.8	0.0	640.7	0.8	641.0	0.8	641.1	0.8	635.8
Lon200–01	1,114.5	0.0	1,133.8	1.7	1,118.3	0.3	1,130.7	1.5	1,114.5
Lon200–02	1,143.5	0.0	1,147.0	0.3	1,146.0	0.2	1,146.6	0.3	1,143.5
Average		0.1		0.3		0.2		0.4	

and combination $(1, 5, 10)$ performs the worst. According to this analysis, we have used $\sigma_1 = 10, \sigma_2 = 5, \sigma_3 = 1$ for the rest of our numerical experiments.

In order to tune other roulette wheel parameters r^p and N_w, we ran experiments on the tuning set instances using four different combination values: (*i*) (0.1, 10), (*ii*) (0.1, 100), (*iii*) (0.2, 10), and (*iv*) (0.2, 100). We can note that if the r^p value is high, it shows that the current performance is more valuable than its past performance, leading to more diverse solutions at N_w iterations. Table 8 compares these four combination values. We have similar columns as in the previous table.

The results show that it is better to use a combination of $(0.1, 100)$ of r^p, N_w values, since they usually lead to the lowest total cost.

5.4. The upper and lower bounds for the removable nodes

We now tune the parameters related to the number of nodes to be removed at each iteration. This interval is estimated with a logarithmic function as suggested by Franceschetti et al. (2017). We now give Table 9 for different values on the tuning set. Similar to previous tables, we present *Average*(l) and *Dev*(l) columns for each combination l.

As shown in Table 9, the best combination for the logarithmic bound function is $(log10, log1.4)$ to decide the number of nodes to be removed at each iteration. We have used this combination for the rest of the experiments.

5.5. Analysis of the operators

This section shows the performance of operators both in terms of speed and the quality of solutions. For each instance, we ran ALNS three times as in previous experiments. The first column in Tables 10 and 11 reports the name of the operators. The first part (top) of each table presents the percentage of iterations that an operator has been used. The second part (bottom) of each table reports the normalized CPU times.

From Tables 10 and 11, we see that operators are utilized in very similar proportions. This is because our implementation is intended to promote diversification more than intensification at further iterations

Table 8
Tuning of (r^p) and (N_w) parameters:

Instance	$(r^p, N_w) =$ (0.1, 10)		$(r^p, N_w) =$ (0.1, 100)		$(r^p, N_w) =$ (0.2, 10)		$(r^p, N_w) =$ (0.2, 100)		Avg
	Average	*Dev*	*Average*	*Dev*	*Average*	*Dev*	*Average*	*Dev*	
Lon10-01	154.5	0.0	154.5	0.0	154.5	0.0	154.5	0.0	154.5
Lon10–02	181.8	0.0	181.8	0.0	181.8	0.0	181.8	0.0	181.8
Lon15–01	167.1	0.0	167.1	0.0	167.1	0.0	167.1	0.0	167.1
Lon15–02	157.8	0.0	157.8	0.0	157.8	0.0	157.8	0.0	157.8
Lon20–01	199.5	0.0	199.5	0.0	199.5	0.0	199.5	0.0	199.5
Lon20–02	195.7	0.0	195.7	0.0	196.0	0.2	196.0	0.2	195.7
Lon25–01	234.0	0.0	234.0	0.0	234.0	0.0	234.0	0.0	234.0
Lon25–02	246.4	0.0	246.4	0.0	246.4	0.0	246.4	0.0	246.4
Lon50–01	371.1	0.0	371.4	0.1	371.3	0.0	372.7	0.4	371.1
Lon50–02	368.1	0.1	367.9	0.0	367.9	0.0	367.9	0.0	367.9
Lon75–01	486.3	0.0	486.4	0.0	487.0	0.2	486.9	0.1	486.3
Lon75–02	493.0	0.2	492.1	0.1	491.8	0.0	493.5	0.3	491.8
Lon100–01	637.8	0.6	633.8	0.0	635.2	0.2	637.8	0.6	633.8
Lon100–02	637.0	0.3	635.2	0.0	637.9	0.4	637.8	0.4	635.2
Lon200–01	1,123.2	0.9	1,113.6	0.1	1,117.2	0.4	1,124.0	1.0	1,112.9
Lon200–02	1,143.5	0.5	1,140.7	0.3	1,147.2	0.8	1,137.6	0.0	1137.6
Average		0.23		0.15		0.21		0.28	

Table 9
Tuning of the number of nodes to be removed at each iteration:

Instance	$(min, max) =$ ($log10, log1.4$)		$(min, max) =$ ($log10, log3$)		$(min, max) =$ ($log15, log1.4$)		$(min, max) =$ ($log15, log3$)		Avg
	Average	*Dev*	*Average*	*Dev*	*Average*	*Dev*	*Average*	*Dev*	
Lon10–01	154.5	0.0	154.5	0.0	154.5	0.0	154.5	0.0	154.5
Lon10–02	181.8	0.0	181.8	0.0	181.8	0.0	181.8	0.0	181.8
Lon15–01	167.1	0.0	167.1	0.0	167.1	0.0	167.1	0.0	167.1
Lon15–02	157.8	0.0	158.3	0.3	157.8	0.0	158.3	0.3	157.8
Lon20–01	199.5	0.0	199.5	0.0	199.5	0.0	200.2	0.3	199.5
Lon20–02	195.7	0.0	195.7	0.0	196.0	0.2	196.4	0.4	195.7
Lon25–01	234.0	0.0	234.2	0.1	234.0	0.0	234.9	0.4	234.0
Lon25–02	246.4	0.0	246.4	0.0	246.4	0.0	246.4	0.0	246.4
Lon50–01	371.4	0.0	372.6	0.3	371.4	0.0	371.9	0.1	371.4
Lon50–02	367.9	0.0	368.0	0.0	367.9	0.0	368.2	0.1	367.9
Lon75–01	486.4	0.0	487.0	0.1	487.0	0.1	487.9	0.3	486.4
Lon75–02	492.1	0.0	496.7	0.9	492.4	0.1	497.9	1.2	492.1
Lon100–01	633.8	0.0	643.9	1.6	634.7	0.1	637.8	0.6	633.8
Lon100–02	635.2	0.0	636.6	0.2	642.3	1.1	636.5	0.2	635.2
Lon200–01	1,113.6	0.2	1,127.3	1.5	1,110.9	0.0	1,121.5	1.0	1,110.9
Lon200-02	1,140.7	0.1	1,150.6	1.0	1,147.2	0.7	1,139.4	0.0	1,139.4
Average		0.0		0.6		0.3		0.5	

of the ALNS. In terms of solution quality, it can be seen that the best-performing removing operator is R3 followed by the R5 and R8 operators. Relatively few improvements have been achieved with the R4 removal operator. Among insertion operators, the best and the second-best performing operators are I3 and I1, respectively. The least-performing insertion operator is I5 operator, as expected.

Table 10
Descriptive statistics for the ten removal operators:

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
	The number of iterations (percentage (%) of 25K)									
Lon10	10.66	8.56	11.20	6.67	10.82	10.27	10.57	10.42	10.56	10.28
Lon15	10.66	8.67	10.77	6.87	12.16	9.72	11.29	10.78	9.96	9.13
Lon20	10.70	11.72	9.63	4.86	10.68	9.72	9.98	11.00	10.79	10.93
Lon25	9.99	10.08	10.38	7.83	10.29	10.10	10.37	10.75	10.07	10.15
Lon50	10.50	10.32	10.84	7.28	10.40	10.05	9.68	9.70	10.15	11.08
Lon75	10.48	9.75	10.27	7.54	10.92	10.33	10.04	10.55	9.90	10.23
Lon100	10.82	10.28	11.48	4.95	10.44	10.39	10.06	11.05	11.03	9.50
Lon200	10.56	10.33	12.08	4.93	10.92	10.07	10.40	10.84	9.15	10.73
Average	10.55	9.96	10.83	6.36	10.83	10.08	10.30	10.64	10.20	10.25
	CPU times required by each operator (seconds)									
Lon10	0.002	0.002	0.003	0.001	0.013	0.001	0.003	0.003	0.002	0.003
Lon15	0.002	0.003	0.007	0.003	0.016	0.002	0.007	0.004	0.002	0.001
Lon20	0.001	0.004	0.008	0.001	0.015	0.001	0.005	0.003	0.004	0.006
Lon25	0.005	0.007	0.020	0.003	0.044	0.004	0.011	0.014	0.009	0.007
Lon50	0.003	0.007	0.015	0.002	0.033	0.003	0.011	0.009	0.006	0.003
Lon75	0.011	0.022	0.051	0.010	0.185	0.028	0.045	0.047	0.027	0.028
Lon100	0.017	0.025	0.070	0.006	0.227	0.024	0.051	0.058	0.032	0.040
Lon200	0.027	0.062	0.214	0.021	0.897	0.056	0.131	0.131	0.065	0.071
Average	0.008	0.016	0.048	0.006	0.179	0.015	0.033	0.034	0.018	0.020

Table 11
Descriptive statistics for the six insertion operators:

	I1	I2	I3	I4	I5	I6
	The number of iterations (percentage (%) of 25K)					
Lon10	17.40	17.39	17.80	16.43	15.14	15.86
Lon15	15.81	15.77	19.68	19.23	15.39	14.13
Lon20	16.56	16.52	16.81	16.66	16.47	16.99
Lon25	16.94	15.76	16.67	16.71	17.10	16.82
Lon50	17.12	17.15	17.41	17.27	13.82	17.23
Lon75	17.08	18.13	15.57	15.41	16.32	17.49
Lon100	17.42	15.28	16.01	16.17	17.36	17.75
Lon200	17.14	18.48	16.19	13.25	17.50	17.44
Average	16.94	16.81	17.02	16.39	16.14	16.71
	CPU times required by each operator (seconds)					
Lon10	0.046	0.074	0.036	0.042	0.001	0.006
Lon15	0.068	0.095	0.058	0.058	0.002	0.006
Lon20	0.046	0.096	0.038	0.047	0.001	0.006
Lon25	0.177	0.355	0.195	0.208	0.006	0.018
Lon50	0.170	0.299	0.170	0.160	0.005	0.010
Lon75	0.856	1.694	0.867	0.831	0.037	0.084
Lon100	1.063	1.924	0.997	1.069	0.053	0.097
Lon200	4.571	7.616	4.032	4.242	0.081	0.529
Average	0.875	1.519	0.799	0.832	0.023	0.094

5.6. Comparison with the results of the mathematical model for small-sized benchmark instances

We now present the performance of our ALNS metaheuristic algorithm on the generated instances. We solve small- and medium-sized instances (i.e., 10, 15, 20 and 25 nodes) using CPLEX solver of IBM ILOG (2018).

Table 12 and 13 compares the algorithm with the mixed-integer linear programming (MILP) model. Column *CPLEX* presents the total cost obtained with the MILP formulation. We note that we have tried two speed level settings for the model. In the first setting, we have used a single speed level to reduce the complexity (i.e., a lower number of decision variables). In the second setting, we have used nine speed levels starting from 11.11 m/s to 22.22 m/s. Since the highest speed level is normally the chosen speed, the objective function values obtained with both settings do not hugely differ. Column *ALNS* presents the best solution found by our metaheuristic algorithm. Column *ALNS with SOP* presents the best solution found by our metaheuristic algorithm with the speed optimization algorithm proposed by Demir et al. (2014b). The columns *GAP* present the differences in objective function values. These values are obtained with the formula of $[Cost(S^{ALNS}_{best}) - Cost(S^{CPLEX})]/Cost(S^{ALNS}_{best})$. The solver time limit for the solution is set to 7,200 seconds in CPLEX. We note that the negative value occurs due to the lineralization of speed variables.

Table 12
Comparison with CPLEX solver on 10- and 15-node instances:

Instance	CPLEX (One speed level) (£)	ALNS (£)	GAP (%)	CPLEX (Nine speed levels) (£)	ALNS with SOP (£)	GAP (%)
Lon10–01	155.10	155.10	–0.0001	154.52	154.52	–0.0004
Lon10–02	182.48	182.48	–0.0001	181.71	181.76	0.0278
Lon10–03	144.89	144.89	–0.0001	143.97	143.97	–0.0006
Lon10–04	164.81	164.81	–0.0001	164.01	164.11	0.0637
Lon10–05	158.14	158.14	–0.0001	157.10	157.10	–0.0007
Lon10–06	138.73	138.73	–0.0001	138.13	138.13	–0.0004
Lon10–07	163.90	163.90	–0.0001	162.52	162.52	–0.0008
Lon10–08	175.53	175.53	–0.0001	174.00	174.00	–0.0008
Lon10–09	150.61	150.61	–0.0001	149.51	149.50	–0.0100
Lon10–10	151.23	151.23	–0.0001	150.32	150.32	–0.0006
Lon15–01	168.00	168.00	–0.0001	167.05	167.12	0.0421
Lon15–02	158.65	158.65	–0.0001	157.82	157.80	–0.0161
Lon15–03	178.25	178.25	–0.0001	177.17	177.20	0.0185
Lon15–04	182.55	182.55	–0.0001	181.06	181.04	–0.0107
Lon15–05	176.67	176.67	–0.0001	175.77	175.76	–0.0052
Lon15–06	161.10	161.10	–0.0001	159.82	159.82	–0.0008
Lon15–07	160.84	160.84	–0.0001	159.84	159.85	0.0042
Lon15–08	182.33	182.33	–0.0001	181.24	181.24	–0.0006
Lon15–09	160.51	160.51	–0.0001	159.50	159.50	–0.0006
Lon15–10	174.14	174.14	–0.0001	172.73	172.72	–0.0046
Average	164.41	164.41	–0.0001	163.39	163.40	–0.0052

The results in Table 12 highlight that 10- and 15-node instances have small deviations between CPLEX and ALNS algorithms. The maximum deviation is 0.06%, which is the worst-case performance for the ALNS. The required CPU time for our algorithm is less than thirty seconds for 15-node instances. For 10-node instances, the CPU time required by CPLEX is less than one second. For 15-node instances, average CPU times with one and nine speed levels are 12 and 28 seconds, respectively. We note that it was not possible to get any optimal solution for 20- and 25-node instances with the solver regardless of chosen speed level setting. Therefore, we only present feasible solutions obtained at 7,200 seconds in Table 13.

Table 13
Comparison with CPLEX solver on 20- and 25-node instances:

Instance	CPLEX* (One speed level) (£)	ALNS (£)	GAP (%)	CPLEX* (Nine speed levels) (£)*	ALNS with SOP (£)	GAP (%)
Lon20–01	200.80	200.80	–0.0002	199.51	199.51	–0.0015
Lon20–02	197.34	197.34	–0.0002	195.69	195.69	–0.0035
Lon20–03	233.06	233.06	–0.0001	231.41	231.76	0.1494
Lon20–04	197.35	197.35	–0.0002	195.57	195.56	–0.0070
Lon20–05	185.36	185.36	–0.0001	183.80	183.80	–0.0008
Lon20–06	207.35	207.35	–0.0001	205.89	205.89	–0.0007
Lon20–07	200.21	200.21	–0.0001	198.29	198.29	–0.0009
Lon20–08	182.15	182.15	–0.0001	180.96	180.96	–0.0035
Lon20–09	185.58	185.58	–0.0002	184.53	184.50	–0.0138
Lon20–10	201.39	201.39	–0.0001	199.66	199.70	0.0221
Lon25–01	236.31	236.31	–0.0001	234.00	234.00	–0.0009
Lon25–02	247.73	247.73	–0.0001	248.49	246.36	–0.8573
Lon25–03	231.62	232.18	0.2422	231.17	231.65	0.2078
Lon25–04	246.34	246.34	–0.0001	245.63	245.62	–0.0062
Lon25–05	218.56	218.56	–0.0002	217.99	217.97	–0.0054
Lon25–06	211.49	211.49	–0.0001	210.00	209.99	–0.0054
Lon25–07	251.99	251.39	–0.2368	250.06	250.04	–0.0077
Lon25–08	213.46	213.46	–0.0002	211.89	211.86	–0.0156
Lon25–09	228.11	228.11	–0.0001	228.77	227.05	–0.7518
Lon25–10	284.47	267.35	–6.4035	286.75	265.89	–7.2730
Average	218.03	217.18	–0.32	217.00	215.80	–0.43

* could not be solved to optimality within 7,200 seconds.

5.7. Results of the algorithm on benchmark instances

This section provides the results on benchmark instances of Chen and Shi (2019). These instances are generated based on well-known Solomon's VRPTW instances Solomon (1987). In order to assess the quality of our algorithm, we have solved sixteen instances, which come in two sets random and randomly clustered with 100 and 2000 nodes. Chen and Shi (2019) provided two metaheuristic algorithms for multi-compartment VRPTW, namely the conventional particle swarm optimization (PSO) and the hybrid particle swarm optimization (HPSO).

Table 14 compares the results of the *PSO*, *HPSO* and *ALNS* algorithms. The comparisons are made with regards to the best objective function values obtained through five runs of the ALNS algorithm with 25,000 iterations. We present, for 16 instances, the value of the solutions found with PSO and HPSO algorithms under column "PSO" and "HPSO", respectively. The column "ALNS" provides the results obtained with our ALNS algorithm. The CPU time with the ALNS algorithm is also presented in the last column of Table 14. The columns titled "Gap PSO (%)" and "Gap HPSO (%)" present the percentage deviation of PSO and HPSO algorithms from ALNS, respectively. The column titled "Gap Best (%)" shows the percentage deviation of best obtained value from the ALNS algorithm.

As can be seen from Table 14, the enhanced ALNS algorithm performs very well on the MCVRPTW instances of Chen and Shi (2019). For the majority of the instances, the ALNS is able to obtain the best known solution (i.e., nine out of 16). For the rest of the instances, the percentage deviations are no greater than 0.45%.

5.8. Analysis on travel and service time uncertainty

This section investigates the service and travel time uncertainty on each cost component (i.e., fuel cost, driver wage, and temperature cost) as provided in Table 15. The first group provides the obtained results with zero uncertainty level. The second group "10% uncertainty level" provides the details when

Table 14
Results of the ALNS algorithm on MCVRPTW benchmark instances:

	PSO	HPSO	ALNS	Best	Gap PSO (%)	Gap HPSO (%)	Gap Best (%)	CPU ALNS (seconds)
R101	1,660.63	1,692.29	**1,657.62**	**1,657.62**	–0.18	–2.09	0.00	83.16
R102	1,526.03	1,541.15	**1,520.54**	**1,520.54**	–0.36	–1.36	0.00	92.71
R103	1,288.22	1,255.60	**1,243.33**	**1,243.33**	–3.61	–0.99	0.00	90.90
R104	1,052.25	**1,036.32**	1,049.27	**1,036.32**	–0.28	1.23	1.23	91.41
R201	1,175.71	1,191.36	**1,156.91**	**1,156.91**	–1.63	–2.98	0.00	143.78
R202	1,106.84	1,084.14	**1,074.70**	**1,074.70**	–2.99	–0.88	0.00	89.58
R203	921.28	**902.62**	915.40	**902.62**	–0.64	1.40	1.40	89.68
R204	**782.75**	797.09	791.90	**782.75**	1.16	–0.66	1.16	102.17
RC101	1,711.29	1,702.69	**1,676.69**	**1,676.69**	–2.06	–1.55	0.00	68.91
RC102	1,572.96	1,581.29	**1,559.88**	**1,559.88**	–0.84	–1.37	0.00	60.50
RC103	**1,363.25**	1,376.89	1,369.01	**1,363.25**	0.42	–0.58	0.42	51.57
RC104	1,244.43	1,229.81	**1,220.92**	**1,220.92**	–1.93	–0.73	0.00	62.11
RC201	1,311.41	**1,282.35**	1,300.51	**1,282.35**	–0.84	1.40	1.40	124.54
RC202	1,163.48	**1,108.01**	1,125.17	**1,108.01**	–3.40	1.53	1.53	132.68
RC203	980.71	1,014.43	954.05	**954.05**	–2.79	–6.33	0.00	78.50
RC204	854.84	**834.73**	**834.73**	**834.73**	–2.41	0.00	0.00	50.60
Average					–1.40	–0.87	0.45	

the uncertainty level is 10%, and the last group “30% uncertainty level" gives the results when uncertainty level is set to 30%. For all three groups, the columns provide details on total objective function value, fuel, and temperature costs (all in pounds).

Table 15
Travel and service time uncertainty for 100-node instances:

Instance	Zero uncertainty level			10% uncertainty level			30% uncertainty level		
	Sol. value (£)	Fuel cost (£)	Temp. cost (£)	Sol. Value (£)	Fuel cost (£)	Temp. cost (£)	Sol. value (£)	Fuel cost (£)	Temp. cost (£)
Lon100–01	633.78	109.25	122.59	696.63	128.17	112.08	801.13	154.51	88.39
Lon100–02	635.25	99.96	127.15	681.39	108.62	114.81	790.23	153.81	105.00
Lon100–03	627.50	106.41	117.54	696.41	117.91	94.07	800.11	147.58	84.73
Lon100–04	656.88	111.70	105.19	685.40	125.68	108.59	786.90	151.86	98.61
Lon100–05	638.56	103.93	113.53	673.78	115.60	105.93	770.14	149.66	104.49
Lon100–06	678.00	112.54	114.63	716.99	126.75	114.15	825.53	160.07	96.43
Lon100–07	630.94	97.55	128.99	667.07	107.21	120.18	726.28	131.57	117.15
Lon100–08	622.13	105.93	122.84	674.18	112.66	110.28	756.95	135.12	101.15
Lon100–09	635.33	106.07	117.50	693.01	117.83	97.75	762.57	149.46	96.51
Lon100–10	621.08	107.96	117.90	663.53	124.21	119.22	761.93	142.06	96.51
Average	637.95	106.13	118.79	684.84	118.46	109.71	778.18	147.57	98.90

Table 15 highlights that the service and travel time uncertainty increases the total costs and fuel consumption. However, temperature costs decrease with the increase in uncertainty level. This can be explained with the lower number of visits per each vehicle used in a solution. Shorter travel time leads to a more efficient use of compartments, compared to deterministic solutions (i.e., zero uncertainty level).

5.9. Analysis on the use of multiple compartments

This section investigates the effect of multiple compartments on each cost component (i.e., fuel, driver, and temperature costs). Table 16 presents the analysis on multi-compartment as explained in Section

3. It is noted that these values are average values of all 100-node instances. The column “Sol. Value" provides the best results of three runs (in pounds). The third “CPU time" and fourth “Dist." columns provide the CPU time and total traveled distance values, respectively. The fifth column “Fuel cons." provides the calculated fuel consumption in liters. We note that the total CO_2equivalent emissions can easily be calculated using the estimated amount of fuel and the corresponding conversion factor (i.e., 3.13kg CO_2e emissions per one liter of diesel fuel). The next three columns “Vehicle cost", “Driver wage" and “Temp. cost" give the results of total vehicle cost, total driver wage and estimated temperature cost for each instance. We also list the number of customers (and the number of vehicles used) in the table.

Table 16
Average results for all 100-node instances:

Compartment	Sol. value (£)	CPU time (sec)	Dist. (km)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
All compartments	637.95	28.02	963.91	106.13	168	245.03	118.79	26.4(2.3)	32.5(2.7)	41.1(3.4)
Normal	545.57	20.38	964.35	106.62	168	243.50	27.45	26.4(2.3)	32.5(2.7)	41.1(3.4)
Low	652.86	21.59	957.22	105.29	172	247.08	128.50	26.4(2.3)	32.5(2.8)	41.1(3.5)
Very Low	777.07	22.06	955.21	104.53	176	255.83	240.71	26.4(2.3)	32.5(2.9)	41.1(3.6)
Normal and Low	585.92	21.44	971.25	107.39	168	243.99	66.54	26.4(2.3)	32.5(2.7)	41.1(3.4)
Normal and Very Low	630.98	20.17	967.42	106.57	172	246.42	105.99	26.4(2.3)	32.5(2.8)	41.1(3.5)
Low and Very Low	689.60	24.11	968.03	106.83	170	249.11	163.67	26.4(2.3)	32.5(2.7)	41.1(3.5)

We note that for each category we only used the stated compartment(s), and other product demands are converted to the chosen compartment(s). Table 16 highlights the difference of having one or more compartments in a vehicle. It is evident that Low and Very Low compartments are more costly than Normal-temperature compartments. There is only a slight difference for the availability of multi compartments on the number of customers visited and the number of vehicles used in each period.

5.10. Analysis on customers' time window preferences

We now look at various time period settings for two 100-node instances (i.e., Lon100–01 and Lon100–02). We look at the original setting and five different variations of these two instances. We assume that all customers are the first type of customers in these variants and they keep similar preferences in each variant. First, we assume that all customers pick a 1–hour time slot from 8:00am in the morning to 24:00 in the evening (i.e., Lon100–01a and Lon100–02a). Second, we assume that all customers choose one of the time periods, 08:00–12:00, 14:00–18:00, or 20:00–24:00 (i.e., Lon100–01b and Lon100–02b). Third, we assume that all customers choose only the first period from 8:00am to 12:00 noon (i.e., Lon100–01c and Lon100–02c). Fourth, all customers choose a time period from 8:00am to 18:00pm (i.e., Lon100–01d and Lon100–02d). And finally, all customers can be served anytime between 8:00 and 24:00 (i.e., Lon100–01e and Lon100–02e). Similar to previous Tables 18–21, we provide similar information for each variant per instance in Table 17.

As seen in Table 17, we can observe that total costs increase with the flexibility in delivery time preferences. The highest cost is achieved if all customers choose a 1-hour time slot for delivery, and the cheapest cost is achieved when only a 4–hour time period is chosen. We also note that time flexibility increases fuel consumption and the total traveled distance. We have similar results in part (c,d,e) because all customers could be visited in four hours by the least number of vehicles. If there were more customers (especially with higher demands), this would result in using more vehicles.

5.11. Solving the medium- and large-sized instances with the ALNS algorithm

We now solve all sets of instances with our ALNS algorithm. We ran the ALNS algorithm three times. For medium- and large-sized instances, solving the MILP formulation is computationally expensive and

Table 17
Computational results for 100–node instances with different delivery time preferences:

Instance	Sol. value (£)	CPU time (sec)	Dist. (km)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
Lon100–01	633.78	28.17	982.39	109.25	160	241.94	122.59	27(2)	37(3)	36(3)
Lon100–02	635.25	29.26	917.74	99.96	160	248.14	127.15	27(2)	35(3)	38(3)
Lon100–01a	686.51	32.46	1,107.31	120.54	180	279.90	106.07	34(3)	31(3)	35(3)
Lon100–02a	684.41	29.24	957.61	104.90	200	268.87	110.64	35(3)	35(3)	30(4)
Lon100–01b	549.17	30.08	711.59	79.97	140	205.36	123.84	36(3)	31(2)	33(2)
Lon100–02b	546.25	29.27	668.08	74.88	160	200.14	111.23	37(3)	35(3)	28(2)
Lon100–01c	524.69	65.07	553.59	62.36	140	189.56	132.77	100(7)	0(0)	0(0)
Lon100–02c	506.44	41.80	501.67	56.65	120	183.50	146.29	100(6)	0(0)	0(0)
Lon100–01d	524.69	65.07	553.59	62.36	140	189.56	132.77	100(7)	0(0)	0(0)
Lon100–02d	506.44	41.80	501.67	56.65	120	183.50	146.29	100(6)	0(0)	0(0)
Lon100–01e	524.69	65.07	553.59	62.36	140	189.56	132.77	100(7)	0(0)	0(0)
Lon100–02e	506.44	41.80	501.67	56.65	120	183.50	146.29	100(6)	0(0)	0(0)

we, therefore, do not provide these values in the following tables. Table 18–21 provides the detailed results on four sets of instances (i.e., Lon–50, Lon–75, Lon–100 and Lon–200). The column “Sol. Value" provides the best values of three runs in pounds. The third column provides the “CPU time" information for each instance. The next column in these tables “Fuel cons." provides the estimated fuel consumption in liters. The next three columns “Vehicle cost", “Driver wage" and “Temp. cost" give the results of total vehicle cost, total driver wage and estimated temperature cost for each instance. Finally, the last three columns provide the number of customers served and the number of vehicles used in each corresponding time period (i.e., Period 1, Period 2, or Period 3).

Table 18
Results of the 50–node instances:

Instance	Sol. value (£)	CPU time (sec)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
Lon50–01	371.37	6.18	63.25	120	136.40	51.72	15(2)	8(2)	27(2)
Lon50–02	367.87	5.93	63.18	120	142.55	42.14	15(2)	13(2)	22(2)
Lon50–03	351.44	5.95	65.33	100	129.96	56.15	13(1)	16(2)	21(2)
Lon50–04	364.53	8.77	57.19	100	140.98	66.36	15(2)	13(1)	22(2)
Lon50–05	387.14	6.35	66.54	120	155.67	44.93	16(2)	16(2)	18(2)
Lon50–06	383.66	7.46	63.73	120	152.80	47.13	12(2)	23(2)	15(2)
Lon50–07	386.08	5.33	62.29	120	163.53	40.27	15(2)	15(2)	20(2)
Lon50–08	369.12	6.14	71.17	100	139.96	57.99	15(2)	15(1)	20(2)
Lon50–09	377.69	6.28	72.08	120	141.70	43.90	17(2)	13(2)	20(2)
Lon50–10	350.83	5.93	70.82	100	131.40	48.61	12(2)	13(1)	25(2)

Table 18 presents the detailed results for 50-node instances. The average solution time is quite small for solving a medium-size instance. This is especially important since we aim to provide quick and good-quality solutions for the sake of city logistics. We see that more customers are visited in the last period. Moreover, we observe that the second and third types of customers need to be visited next to the first type of customers. The maximum number of vehicles is found to be two for all instances in this

group.

Table 19
Results of the 75–node instances:

Instance	Sol. value (£)	CPU time (sec)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
Lon75–01	486.44	21.00	78.82	120	187.61	100.01	23(2)	26(2)	26(2)
Lon75–02	492.12	14.74	84.68	140	191.61	75.83	19(2)	24(2)	32(3)
Lon75–03	506.33	17.05	82.36	160	190.39	73.58	16(2)	24(3)	35(3)
Lon75–04	486.84	20.37	73.41	120	187.99	105.44	18(2)	30(2)	27(2)
Lon75–05	495.31	20.12	85.87	140	192.64	76.81	20(2)	21(2)	34(3)
Lon75–06	500.65	20.63	80.78	140	196.43	83.44	25(2)	19(2)	31(3)
Lon75–07	535.36	21.69	83.70	160	204.52	87.14	18(2)	20(3)	37(3)
Lon75–08	479.06	18.79	84.91	120	180.79	93.36	23(2)	26(2)	26(2)
Lon75–09	520.37	21.27	78.90	140	216.43	85.03	15(2)	31(3)	29(2)
Lon75–10	515.88	12.52	81.38	140	203.76	90.74	27(2)	23(3)	25(2)

Similar to Table 18, Table 19 lists the solution details for all 10 instances. The solution times are still quite small and applicable for city logistics. We also highlight that there is only one extra vehicle needed to serve an additional 25 customers. Since the total vehicle cost is part of the objective function, the algorithm also aims to reduce the number of vehicles. This provides an opportunity for LSPs to decide their fleet size for a given number of customers.

Table 20
Results of 100–node instances:

Instance	Sol. value (£)	CPU time (sec)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
Lon100–01	633.78	28.17	109.25	160	241.94	122.59	27(2)	37(3)	36(3)
Lon100–02	635.25	29.26	99.96	160	248.14	127.15	27(2)	35(3)	38(3)
Lon100–03	627.50	32.38	106.41	160	243.55	117.54	23(2)	37(3)	40(3)
Lon100–04	655.65	34.24	109.65	180	258.60	107.04	31(3)	33(3)	36(3)
Lon100–05	638.56	36.17	103.93	180	241.10	113.53	24(2)	30(3)	46(4)
Lon100–06	678.00	33.50	112.54	200	250.83	114.63	21(3)	31(3)	48(4)
Lon100–07	630.26	33.71	94.92	160	245.69	129.65	30(2)	34(3)	36(3)
Lon100–08	622.13	16.31	105.93	160	233.36	122.84	21(2)	30(2)	49(4)
Lon100–09	635.33	15.71	106.07	160	251.76	117.50	27(2)	31(2)	42(4)
Lon100–10	621.08	20.74	107.96	160	235.22	117.90	33(3)	27(2)	40(3)

Table 20 provides the results of 100-node instances. Average solution time for these instances have increased to 27.3 seconds. We can also observe that fuel consumption is not too high since city van has been used in the experiments. Based on this analysis, only four vehicles would be sufficient to serve the customers located in London. Another interesting finding of the study is that more customers are visited in the afternoon and evening periods than the morning one.

As can be seen from Tables 21, all instances up to 200 nodes can be solved within 1 minute and thirty one seconds. This is one of the biggest advantages of using ALNS metaheuristic algorithms for VRPs. There is clear evidence that temperature costs are an important part of the objective function and this can be explained by the fact that a certain level of compartment temperature must be kept even when the vehicle stops for a delivery. Shorter service times will minimize the temperature costs as expected. Again, the number of vehicles needed for each period does not differ dramatically for each instance in the same set.

Table 21
Results of the 200–node instances:

Instance	Sol. value (£)	CPU time (sec)	Fuel cons. (L)	Vehicle cost (£)	Driver wage (£)	Temp. cost (£)	Period 1 C(V)	Period 2 C(V)	Period 3 C(V)
Lon200–01	1,113.58	46.41	154.13	260	422.11	277.34	61(4)	60(4)	79(5)
Lon200–02	1140.74	104.04	158.35	300	419.04	263.35	63(5)	56(4)	81(6)
Lon200–03	1154.39	79.77	160.39	300	425.71	268.29	65(5)	64(5)	71(6)
Lon200–04	1104.91	99.65	156.03	260	415.34	273.54	63(4)	66(4)	71(5)
Lon200–05	1163.19	91.37	163.04	300	449.32	250.84	58(4)	64(5)	78(6)
Lon200–06	1153.33	80.11	154.12	280	437.84	281.37	43(3)	70(5)	87(6)
Lon200–07	1155.47	103.03	164.05	280	431.31	280.11	54(4)	71(5)	75(5)
Lon200–08	1197.15	87.63	171.28	300	457.82	268.05	55(4)	67(5)	78(6)
Lon200–09	1121.88	92.48	152.66	280	430.11	259.11	52(4)	65(4)	83(5)
Lon200–10	1164.48	72.82	159.10	300	443.60	261.78	62(4)	68(5)	70(6)

6. Conclusions

We have used the enhanced ALNS algorithm for the solution of a real-life vehicle routing problem with multiple compartments. The proposed ALNS algorithm utilizes both new and existing removal and insertion operators from the literature. All operators are fast and provide good solution quality for the use of city logistics. This is especially important since the routing problem should be solved within minutes, as in real-life applications.

To numerically evaluate the proposed metaheuristic algorithm, we have created various sets of instances based on London geographic data. Additional analysis of various multiple compartments and customer time window preferences has also been conducted for further insights. Moreover, we have looked at service and travel time uncertainty to propose more robust solutions for the practical use of the algorithm. The computational results on all types of instances highlight that the ALNS algorithm is performing well in finding good-quality solutions on instances with up to 200 customers. From this research, three important conclusions are deducted as follows.

First, for the use of multi-compartment vehicles, the complexity of the routing problem increases. This does not come as a surprise. However, LSPs in urban areas need to carry different types of products within the same vehicle. This will require more advanced algorithms to operate flexible and modular vehicles. Among the costs defined in this research (i.e., fuel costs, driver wages, temperature-controlled compartment costs), driver wages and temperature costs increase due to the need for multiple stops on a route. Especially, driver wage is an important issue since customer preferences might lead to variable driver costs for different time periods. Second, another finding is that LSPs need to use robust models to consider travel and service time uncertainty. In urban areas, the speed of an vehicle is limited due to traffic congestion. This can lead to infeasible solutions since some customers cannot be visited on time. A little effort on incorporating robust modelling techniques into their planning systems can benefit LSPs. Third, customers' delivery preferences have big impact on operational costs. One option would be to guide customers with more environmentally-friendly delivery slots. This is somehow done in practice, but more efforts need to be put into showing customers with the actual impact of their delivery preferences on the environment.

In order to extend the current analysis, two possible research areas can be considered for the future. Logistics companies should consider their resources (i.e., vehicle fleet, driver portfolio) to offer last-mile solutions. It is important to tailor delivery services by relaxing such resources. For example, city logistics might offer crowdsourcing and the use of electric vehicles with modular compartments. One future research direction will be to consider heterogeneous vehicle types, including both diesel- and electric-powered vehicles. Especially, temperature-controlled compartments will be an interesting research direction for electric vehicles as we know that current electric battery technology has certain

limitations. Another future study would be on unavailability of customers during the delivery. This is a real challenge for LSPs and therefore algorithms should consider stochastic approach to characterize the behaviour of customers. We also aim to extend our research to look at this issue as a new research direction.

Acknowledgements

Thanks are due to the Editor-In-Chief, Area Editor, and to two anonymous referees for their invaluable comments and suggestions.

References

Aarts, E., Korst, J., Michiels, W., 2005. Simulated annealing, in: Search methodologies. Springer, pp. 187–210.

Alinaghian, M., Shokouhi, N., 2018. Multi-depot multi-compartment vehicle routing problem, solved by a hybrid adaptive large neighborhood search. Omega 76, 85–99.

Barth, M., Younglove, T., Scora, G., 2005. Development of a Heavy-Duty Diesel Modal Emissions and Fuel Consumption Model. Technical Report. UC Berkeley: California Partners for Advanced Transit and Highways (PATH).

Bektaş, T., Laporte, G., 2011. The pollution-routing problem. Transportation Research Part B: Methodological 45, 1232–50.

Ben-Tal, A., Nemirovski, A., 1998. Robust convex optimization. Mathematics of Operations Research 23, 769–805.

Chen, J., Shi, J., 2019. A multi-compartment vehicle routing problem with time windows for urban distribution–a comparison study on particle swarm optimization algorithms. Computers & Industrial Engineering 133, 95–106.

Christofides, N., Mingozzi, A., Toth, P., 1979. The vehicle routing problem. Wiley, Chichester.

Christopher, M., 2016. Logistics & supply chain management. Pearson UK.

Clarke, G., Wright, J., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–81.

Crainic, T.G., Ricciardi, N., Storchi, G., 2009. Models for evaluating and planning city logistics systems. Transportation Science 43, 432–54.

Dantzig, G.B., Ramser, J.H., 1959. The truck dispatching problem. Management Science 6, 80–91.

Demir, E., Bektaş, T., Laporte, G., 2011. A comparative analysis of several vehicle emission models for road freight transportation. Transportation Research Part D: Transport and Environment 6, 347–57.

Demir, E., Bektaş, T., Laporte, G., 2014a. The bi-objective pollution-routing problem. European Journal of Operational Research 232, 464–78.

Demir, E., T., B., G., L., 2012. An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. European Journal of Operational Research 223, 346–59.

Demir, E., T., B., G., L., 2014b. A review of recent research on green road freight transportation. European Journal of Operational Research 237, 775–93.

El Fallahi, A., Prins, C., Calvo, R.W., 2008. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. Computers & Operations Research 35, 1725–41.

Eshtehadi, R., Fathian, M., Demir, E., 2017. Robust solutions to the pollution-routing problem with demand and travel time uncertainty. Transportation Research Part D: Transport and Environment 51, 351–63.

Franceschetti, A., Demir, E., Honhon, D., Van Woensel, T., Laporte, G., Stobbe, M., 2017. A metaheuristic for the time-dependent pollution-routing problem. European Journal of Operational Research 259, 972–91.

Ghiani, G., Laporte, G., Musmanno, R., 2013. Introduction to logistics systems management. John Wiley & Sons.

Golden, B.L., Raghavan, S., Wasil, E., 2008. The vehicle routing problem: Latest advances and new challenges. Springer.

Groß, P.O., Ehmke, J.F., Mattfeld, D.C., 2019. Cost-efficient and reliable city logistics vehicle routing with satellite locations under travel time uncertainty. Transportation Research Procedia 37, 83–90.

Hemmelmayr, V.C., Cordeau, J.F., Crainic, T.G., 2012. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Computers & Operations Research 39, 3215–28.

IBM ILOG, 2018. Copyright ©International Business Machines Corporation 2018.

Insight, A., 2016. UK Sameday Delivery Market Insight Report 2016. Technical Report. Apex Insight.

Laporte, G., 2007. What you should know about the vehicle routing problem. Naval Research Logistics 54, 811–9.

Lee, H.L., Whang, S., 2001. Winning the last mile of e-commerce. MIT Sloan Management Review 42, 54–62.

Muyldermans, L., Pang, G., 2010. On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm. European Journal of Operational Research 206, 93–103.

Ostermeier, M., Hübner, A., 2018. Vehicle selection for a multi-compartment vehicle routing problem. European Journal of Operational Research 269, 682–94.

Rai, A., Tassou, S.A., 2017. Environmental impacts of vapour compression and cryogenic transport refrigeration technologies for temperature controlled food distribution. Energy Conversion and Management 150, 914–23.

Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science 40, 455–72.

Ross, M., 1994. Automobile fuel consumption and emissions: effects of vehicle and driving characteristics. Annual Review of Energy and the Environment 19, 75–112.

Sampaio, A., Savelsbergh, M., Veelenturf, L., Van Woensel, T., 2019. Crowd-based city logistics, in: Faulin, J., Grasman, S.E., Angel, A.J., Hirsch, P. (Eds.), Sustainable Transportation and Smart Logistics. Elsevier, Amsterdam. chapter 15, pp. 381–400.

Savelsbergh, M., Van Woensel, T., 2016. 50th anniversary invited article–city logistics: Challenges and opportunities. Transportation Science 50, 579–90.

Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems, in: Lecture Notes in Computer Science, Springer, Berlin. pp. 417–31.

Silvestrin, P.V., Ritt, M., 2017. An iterated tabu search for the multi-compartment vehicle routing problem. Computers & Operations Research 81, 192–202.

Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–65.

Stellingwerf, H.M., Kanellopoulos, A., van der Vorst, J.G., Bloemhof, J.M., 2018a. Reducing co 2 emissions in temperature-controlled road transportation using the ldvrp model. Transportation Research Part D: Transport and Environment 58, 80–93.

Stellingwerf, H.M., Laporte, G., Cruijssen, F.C., Kanellopoulos, A., Bloemhof, J.M., 2018b. Quantifying the environmental and economic benefits of cooperation: A case study in temperature-controlled food logistics. Transportation Research Part D: Transport and Environment 65, 178–93.

Taniguchi, E., Thompson, R.G., 2018a. City Logistics 1: New Opportunities and Challenges. John Wiley & Sons.

Taniguchi, E., Thompson, R.G., 2018b. City Logistics 2: Modeling and Planning Initiatives. volume 2. Wiley Online Library.

Tassou, S., De-Lille, G., Ge, Y., 2009. Food transport refrigeration–approaches to reduce energy consumption and environmental impacts of road transport. Applied Thermal Engineering 29, 1467–77.

Tassou, S., Lewis, J., Ge, Y., Hadawey, A., Chaer, I., 2010. A review of emerging technologies for food refrigeration applications. Applied Thermal Engineering 30, 263–76.

Toth, P., Vigo, D., 2014. Vehicle routing: problems, methods, and applications. volume 18. SIAM.

Wang, Y., Zhang, D., Liu, Q., Shen, F., Lee, L.H., 2016. Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions. Transportation Research Part E: Logistics and Transportation Review 93, 279–93.

Xu, S.X., Cheng, M., Huang, G.Q., 2015. Efficient intermodal transportation auctions for b2b e-commerce logistics with transaction costs. Transportation Research Part B: Methodological 80, 322–37.

Yao, X., Cheng, Y., Song, M., 2019. Assessment of collaboration in city logistics: From the aspects of profit and co2 emissions. International Journal of Logistics Research and Applications , 1–16.