Publié par :
Published by:
Publicación de la:

Faculté des sciences de l'administration
2325, rue de la Terrasse
Pavillon Palasis-Prince, Université Laval
Québec (Québec) Canada G1V 0A6
Tél. Ph. Tel. : (418) 656-3644
Télec. Fax : (418) 656-7047

Disponible sur Internet :
Available on Internet
Disponible por Internet :

http://www4.fsa.ulaval.ca/la-recherche/publications/documents-de-travail/

DOCUMENT DE TRAVAIL 2020-004

Time-Dependent Fleet Size and Mix
Location Routing Problem

Carise E. SCHMIDT
Arinei C. L. SILVA
Maryam DARVISH
Leandro C. COELHO

Document de travail également publié par le Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, sous le numéro CIRRELT-2020-13

Mai 2020

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
Bibliothèque et Archives Canada, 2020

ISBN 978-2-89524-504-9 (PDF)

Time-Dependent Fleet Size and Mix Location Routing Problem

Carise E. Schmidt[1], **Arinei C. L. Silva**[2], **Maryam Darvish**[3,*], **Leandro C. Coelho**[3]

[1] Instituto Federal de Santa Catarina (IFSC), Brazil
[2] PPGMNE, Universidade Federal do Paraná, Brazil
[3] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Operations and Decision Systems, 2325, rue de la Terrasse, Université Laval, Québec, Canada, G1V 0A6

**Corresponding author: maryam.darvish@cirrelt.ca*

ABSTRACT

In this paper, we consider an extension of the classical location routing and the time-dependent vehicle routing problems to introduce the time-dependent fleet size and mix location routing problem. We develop the mathematical model for this new challenging problem, along with several generic and problem specific valid inequalities. A new powerful metaheuristic is proposed to solve different sized instances of the problem which are generated from the real traffic data. Our metaheuristic is assessed on a special case of the problem from the literature and obtains many new best-known solutions. For the newly introduced problem, we provide good solutions and tight bounds. Our Computational results demonstrate the importance of using a powerful algorithm to solve complex optimization problems.

Keywords: Urban logistics, integrated logistics, vehicle routing, location, time-dependent.

Acknowledgments: This project was partly funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant 2019-00094. This support is greatly acknowledged. We thank Hamza Heni and Khaled Belhassine for their help on creating the instances.

1. Introduction

City logistics is faced with several major challenges, among which congestion and traffic are of great concern. Finding an efficient and effective way to transport goods in urban areas has became the definition of the city logistics itself [26]. This efficient and effective distribution within cities is not only related to cost reduction, but it also aims to alleviate and avoid congestion and, consequently, greenhouse gas emissions (GHG). After all, research on green freight transportation highlights the effect of congestion on fuel consumption and GHG emissions [7].

Besides distribution decisions, there are several other decisions to be considered in a city logistics context that have a direct impact on the distribution costs and GHG emissions. In this sense, facility location, routing, and fleet composition are known as the main decisions to be optimized [19]. Research and practice show that all these decisions are interdependent and, therefore, must be jointly optimized. Several integrated supply chain optimization problems have been introduced in the literature that combine decisions from different levels. Strategic level decisions such as facility location are combined with the vehicle routing problem (VRP), leading to the Location Routing Problem (LRP) (see Drexl and Schneider [9]). At the same time, tactical level decisions for the composition of the fleet, also known as the fleet size and mix, are considered in several LRP studies [18, 19]. These problems are very applicable to real world situations in which a fleet of heterogeneous vehicles is available to conduct distribution.

In this paper, we study the time-dependent fleet size and mix location routing problem (TD-FSMLRP). Our main contribution lies in considering that routing costs depend on the time of the day, i.e., a time-dependent routing cost. In routing problems, this cost is mainly based on the distances, arguing that fuel expenses rise as longer distances are traveled. In reality, the fuel consumption and, consequently, the routing cost depend on the distance traveled, the time spent in traffic, and the speed, among other factors. Using a real traffic database, we estimate the routing costs considering the travel time.

In addition to introducing the TD-FSMLRP, the contributions of this work are as follows. We formally define and model the problem and solve it with both exact and approximate approaches. Several instances of the problem are developed using real traffic data from Quebec

City. These instances are used to show the effectiveness of the proposed solution algorithms. As they are generated from real data, they also help us in gaining insights on the impacts of location and routing decisions on city logistics issues.

The remainder of this paper is as follows. In Section 2, we provide an overview of the studies related to the TD-FSMLRP. In Section 3, we present the formal description of the problem and its mathematical formulation. Our proposed heuristic is described in Section 4. This is followed by the computational experiments in Section 5. Finally, we draw the conclusions of our study in Section 6.

2. Literature review

This paper is closely related to two areas of research: fleet size and mix LRPs and time-dependent routing problems. In this section, we provide a brief review of the recent state-of-the-art in these domains.

Since its introduction by Von Boventer [34], LRPs have been broadly studied in the literature [9, 30]. Particularly, this problem is very applicable within the context of city logistics. In their recent work, Schneider and Löffler [31] use a large composite neighborhood algorithm to solve large scale LRP instances. As the LRP has become an area of interest in the past ten years, several new variants have been introduced and studied in the literature: multi-commodity LRP [2], flexible two-echelon LRP [6], electric LRP [27, 28], and green LRP [10]. Fleet size and mix LRPs are also among the variants being studied [18]. More generally, the idea of sourcing from different locations and pooling inventory to satisfy a given demand with minimum cost has been shown to be effective [33].

First introduced by Golden et al. [14], fleet size and mix vehicle routing is also a very well established class of routing problems (see Renaud and Boctor [23], Lahyani et al. [21]). This problem differs from the heterogeneous VRP as the fleet is considered to be unlimited [19]. Arguing that in many companies a fleet of different capacities is available, Salhi and Fraser [25] incorporate heterogeneous vehicles in the LRP context. Wu et al. [35] propose a decomposition method to solve the multi-depot LRP with a limited fleet of heterogeneous vehicles. In the LRP

proposed by Ambrosino et al. [1], a heterogeneous fleet of vehicles is considered where location, fleet assignment, and routing decisions need to be made. The problem is solved by a two-phase heuristic approach. I Koç et al. [18], time windows are added to the fleet size and mix LRP where a number of vehicles with different capacities and costs are available to serve customers from a set of potential depots. Later, Koç et al. [19] study the impact of depot location, fleet size and mix, and routing decisions on fuel consumption, emissions, and operational costs, by solving the problem with an adaptive large neighborhood search metaheuristic.

The common assumption in most fleet size and mix location routing studies is the constant traveling time. Whereas, in reality, the cost varies with respect to the traffic behavior. With unpredictable incidents aside, the traveling time (or speed) can be defined as a function of the time of the day. Malandraki and Daskin [22] introduce and model the time-dependent VRP (TD-VRP). For a comprehensive review of the literature, see Gendreau et al. [12]. Despite the recent interest in time-dependent routing research, studies on the variants of TD-VRP are very limited and mainly restricted to cases with time windows (e.g., Figliozzi [11], Taş et al. [32], Heni et al. [16]). In the green vehicle routing and scheduling problem studied in Xiao and Konak [36], heterogeneous vehicles and time-varying traffic congestion are considered. The only paper that studies the time-dependent location routing problem (TD-LRP) is the one of Schmidt et al. [29] where a limited fleet of homogeneous vehicles is considered and a single depot must be selected.

The evident lack of the literature in studying TD-LRP with a heterogeneous fleet of vehicles and multiple depots has inspired this paper, in which fleet optimization is incorporated.

3. Problem description and formulation

In this section, we formally describe the TD-FSMLRP and present its mathematical formulation. The TD-FSMLRP is defined on a directed graph $G = (N, A)$, where N represents the node set and A is the set of arcs. Let N_d be the set of all the potential depots and N_c be the set of customers. We also consider a set of dummy nodes called terminals, denoted by N_t, to be used by each type of vehicle as they return to the depot, such that $N_d \cap N_c = \emptyset$, $N_d \cap N_t = \emptyset$ and $N_c \cap N_t = \emptyset$.

Let K be the set of $|K|$ type of vehicles, each with a limited capacity Q_k. Each terminal is linked to only one type of vehicle, i.e., for each potential depot $i \in N_d$, we define $\delta_k(i)$ as a unique subset of terminals linked to vehicles of type $k \in K$, that is, $\delta_k(i) \subseteq N_t$. Therefore, we have a terminal $v \in \delta_k(i)$ for each type of vehicle from the fleet, in each potential depot. This indicates that each vehicle has to return to the depot it belongs to. Let also A_{dc}, A_{cc}, A_{ct} be arc sets such that each arc (i, j) is given from the Cartesian products as $A_{dc} = N_d \times N_c$; $A_{cc} = N_c \times N_c$, $i \neq j$, and $A_{ct} = \cup_{i \in N_d}\{N_c \times \delta(i)\}$ such that $A = A_{dc} \cup A_{cc} \cup A_{ct}$.

The graph is time-dependent, meaning that as the traffic condition changes, the time it takes to traverse any arc (i, j) also changes. We define H as the set of time intervals, where an interval is a period of time over which traffic pattern is constant. There is a deterministic travel time t_{ij}^h associated to each arc $(i, j) \in A$ and each interval $h \in H$.

We consider a single period (i.e., a day) divided into $m+1$ intervals, where each time interval $h \in H = \{0, 1, .., h, .., m\}$ has the same length of $\overline{T}$ seconds. Therefore, $[h\overline{T}, (h+1)\overline{T} - \varepsilon]$ represents the time interval associated with h, where ε is a very small positive number representing the smallest time unit, i.e., one second.

The demand and the service time associated to each customer $i \in N_c$ are denoted by q_i and s_i, respectively. The fixed cost for each vehicle of type $k \in K$ is denoted by F_k. Let W_i be the capacity of depot $i \in N_d$ and O_i its opening cost. Finally, C is the coefficient used to convert the travel time into its cost equivalent.

We define our formulation based on the following binary variables: x_{ij}^h indicate whether arc (i, j) is traversed by a vehicle during interval h; z_{ij} take value of 1 if arc (i, j) is traversed by a vehicle; w_i are used to determine if depot i is selected or not; y_i^h take value of 1 if a route leaves from customer $i \in N_c$ in time interval $h \in H$; and finally, g_{ij}^h take value of 1 if a route leaves from depot $i \in N_d$ toward customer $j \in N_c$ in interval $h \in H$. We also define the following continuous variables: a_i represent the departure time from customer $i \in N_c$; b_{ij} represent the departure time from depot $i \in N_d$ or the arrival time to terminal $j \in N_t$; u_i represent a bound on the accumulated deliveries to all customers already visited before departing from customer $i \in N_c$. The minimum value for these variables is the accumulated deliveries and the maximum value is the capacity of the vehicle used for the delivery. Therefore, if the total

demand delivered by a vehicle is less than the vehicle capacity, this variable may not represent the accumulated demand delivered, as the difference between the vehicle capacity and the total demand of that route represents a slack. Variables l_{ij} are used in the same sense, as they represent the accumulated demand delivered by a vehicle upon its arrival to terminal $j \in N_t$. As before, if the total capacity of the vehicle is not used, there is a slack equal to the unused capacity. Finally, continuous variables f_i take the value of the index of the depot to which the node $i \in N$ is linked.

Table 1 summarizes the notation used in our model.

Table 1: Notation used in the model

Parameters	
C	Routing coefficient to covert time to cost
F_k	Fixed cost of each vehicle of type $k \in K$
$m+1$	Number of intervals in the planning horizon
O_i	Opening cost of depot $i \in N_d$
q_i	Demand of each customer $i \in N_c$
Q_k	Capacity of each vehicle of type $k \in K$
s_i	Service time of each customer $i \in N_c$
t_{ij}^h	Travel time of arc $(i,j) \in A$ in interval h
$\overline{T}$	The length of each interval
W_i	Capacity of depot $i \in N_d$
Sets	
N_c	Set of customers
N_d	Set of potential depots
N_t	Set of terminals
K	Set of vehicle types
H	Set of time intervals
A	Set of arcs
$\delta_k(i)$	Unique subset of terminals linked with each vehicle type $k \in K$ for each potential depot $i \in N_d$
Variables	
a_i	Departure time from customer $i \in N_c$
b_{ij}	Departure time from depot $i \in N_d$ or arrival time to terminal $j \in N_t$ where $(i,j) \in A \setminus A_{cc}$
f_i	The index of the depot to which node $i \in N_d$ is linked.
g_{ij}^h	If a route exists between depot $i \in N_d$ and customer $j \in N_c$ in interval $h \in H$
l_{ij}	A value between the accumulated demand delivered arriving to terminal $j \in N_t$ and the capacity of the vehicle
u_i	A value between the accumulated demand delivered departing from customer $i \in N_c$ and the capacity of the vehicle
w_j	If depot j is selected
x_{ij}^h	If arc (i,j) is traversed by a vehicle in interval h
y_i^h	If a route leaves from customer $i \in N_c$ in time interval $h \in H$
z_{ij}	If arc (i,j) is traversed by a vehicle
Indices	
h	Time interval
i, j, v	Nodes

The mathematical formulation is as follows:

$$\min \sum_{(i,j)\in A} \sum_{h\in H} Ct_{ij}^h x_{ij}^h + \sum_{i\in N_d} O_i w_i + \sum_{k\in K} \sum_{j\in N_c} \sum_{\substack{v\in\delta_k(i)\\ i\in N_d}} F_k z_{jv} \tag{1}$$

subject to:

$$\sum_{i\in(N_c\setminus\{j\})\cup N_d} z_{ij} = 1, \quad \forall j \in N_c \tag{2}$$

$$\sum_{j\in(N_c\setminus\{i\})\cup N_t} z_{ij} = 1, \quad \forall i \in N_c \tag{3}$$

$$\sum_{j\in N_c} z_{ij} \leq |N_c| w_i, \quad \forall i \in N_d \tag{4}$$

$$\sum_{j\in N_c} z_{ij} = \sum_{j\in N_c} \sum_{k\in K} \sum_{v\in\delta_k(i)} z_{jv}, \quad \forall i \in N_d \tag{5}$$

$$u_i - u_j + \max_{k\in K}\{Q_k\} z_{ij} \leq \max_{k\in K}\{Q_k\} - q_j, \quad \forall i,j \in N_c, i \neq j \tag{6}$$

$$q_i \leq u_i \leq \max_{k\in K}\{Q_k\}, \quad \forall i \in N_c \tag{7}$$

$$u_j - \max_{k\in K}\{Q_k\}(1 - z_{jv}) \leq l_{jv} \leq u_j, \quad \forall j \in N_c, \forall v \in N_t \tag{8}$$

$$l_{ij} \leq Q_k z_{ij}, \quad \forall i \in N_c, \forall j \in \delta_k(v), \forall k \in K, \forall v \in N_d \tag{9}$$

$$\sum_{j\in N_c} \sum_{\substack{v\in\delta_k(i)\\ k\in K}} l_{jv} \leq W_i, \forall i \in N_d \tag{10}$$

$$\sum_{h\in H} x_{ij}^h = z_{ij}, \quad \forall (i,j) \in A \tag{11}$$

$$x_{ij}^h \leq y_i^h, \quad \forall (i,j) \in A \setminus A_{dc}, \forall h \in H \tag{12}$$

$$x_{ij}^h \leq g_{ij}^h, \quad \forall (i,j) \in A_{dc}, \forall h \in H \tag{13}$$

$$\sum_{h\in H} g_{ij}^h = z_{ij}, \quad \forall (i,j) \in A_{dc} \tag{14}$$

$$\sum_{h\in H} y_i^h = 1, \quad \forall i \in N_c \tag{15}$$

$$b_{ij} + s_j + t_{ij}^h - 2\overline{T}|H|(1 - x_{ij}^h) \leq a_j \leq b_{ij} + s_j + t_{ij}^h + \overline{T}|H|(1 - x_{ij}^h), \quad \forall i \in N_d, \forall j \in N_c, \forall h \in H \tag{16}$$

$$a_i + s_j + t_{ij}^h - 2\overline{T}|H|(1 - x_{ij}^h) \leq a_j \leq a_i + s_j + t_{ij}^h + \overline{T}|H|(1 - x_{ij}^h), \quad \forall i,j \in N_c, i \neq j, \forall h \in H \tag{17}$$

$$a_j + t_{jv}^h - 2\overline{T}|H|(1 - x_{jv}^h) \leq b_{jv} \leq a_j + t_{jv}^h + \overline{T}|H|(1 - x_{jv}^h), \quad \forall j \in N_c, \forall v \in N_t, \forall h \in H \tag{18}$$

$$b_{ij} \leq (\overline{T}|H| - \varepsilon)z_{ij}, \quad \forall (i,j) \in A \setminus A_{cc} \tag{19}$$

$$\sum_{h \in H} h\overline{T}y_i^h \leq a_i \leq \sum_{h \in H} h\overline{T}y_i^h + (\overline{T} - \varepsilon), \quad i \in N_c \tag{20}$$

$$\sum_{h \in H} h\overline{T}g_{ij}^h \leq b_{ij} \leq \sum_{h \in H} h\overline{T}g_{ij}^h + (\overline{T} - \varepsilon), \quad \forall (i,j) \in A_{dc} \tag{21}$$

$$f_i = i, \quad \forall i \in N_d \tag{22}$$

$$f_j = f_i, \quad \forall j \in \delta_k(i), \forall k \in K, \forall i \in N_d \tag{23}$$

$$f_i + |N_d|(z_{ij} - 1) \leq f_j \leq f_i + |N_d|(1 - z_{ij}), \quad \forall (i,j) \in A \tag{24}$$

$$w_i \in \{0,1\}, \quad \forall i \in N_d \tag{25}$$

$$z_{ij} \in \{0,1\}, \quad \forall (i,j) \in A \tag{26}$$

$$y_i^h \in \{0,1\}, \quad \forall i \in N_c, \forall h \in H \tag{27}$$

$$g_{ij}^h \in \{0,1\}, \quad \forall (i,j) \in A_{dc}, \forall h \in H \tag{28}$$

$$x_{ij}^h \in \{0,1\}, \quad \forall (i,j) \in A, \forall h \in H \tag{29}$$

$$a_i, u_i \in \mathbb{R}_+, \quad \forall i \in N_c \tag{30}$$

$$b_{ij} \in \mathbb{R}_+, \quad \forall (i,j) \in A \setminus A_{cc} \tag{31}$$

$$l_{ij} \in \mathbb{R}_+, \quad \forall (i,j) \in A_{ct} \tag{32}$$

$$f_i \in \mathbb{R}_+, \quad \forall i \in N. \tag{33}$$

The objective function (1) minimizes the sum of routing costs, opening costs of the depots, and fixed vehicle costs. Constraints (2) and (3) are assignment constraints. They assure that each customer is visited exactly once. Constraints (4) enforce that vehicles leave only from the selected depots. The vehicle flow between each depot and its terminals is balanced by constraints (5). Constraints (6) and (7) are the extensions of the Miller-Tucker-Zemlin subtour elimination, originally proposed by Kulkarni and Bhave [20]. Constraints (8) control the accumulated demand delivered and constraints (9) restrict the vehicle capacity. The total

demand assigned to a depot cannot exceed its capacity as imposed by constraints (10). Each arc (i, j) is visited in a single interval as imposed by constraints (11). Constraints (12) and (13) enforce that if an arc (i, j) is traversed by a vehicle in time interval h, then h is the time interval considered in the departure from the origin i. Constraints (14) and (15) guarantee that only one time interval h is associated with the departure from node i. We control the departure time from the first customer of each route using constraints (16). Similarly, constraints (17) control the departure time from all the other customers. The same control is applied for the arrival time to the terminals with constraints (18). Constraints (19) ensure that the vehicle performs its route within the planning horizon. The departure time from each node i is linked to subsequent time intervals by (20) and (21). Constraints (22)–(24) ensure that each vehicle returns to its original depot. Finally, constraints (25)–(33) enforce integrality and non-negativity conditions on the variables.

3.1. Removing integrality constraints for the x variables

Note that constraints (12) and (27), (13) and (28) imply $x_{ij}^h \leq 1$. Moreover, as stated in the next theorem, the non-negativity of the variables x_{ij}^h already guarantees their integrality, that is, when we assume the non-negativity of the variables x_{ij}^h, constraints (29) automatically become redundant.

Theorem 1. *In the model* (1)–(33)*, integrality conditions* (29) *on variables x can be removed. Notably, if*

$$x_{ij}^h \geq 0, \quad \forall (i, j) \in A, \forall h \in H, \tag{34}$$

then $x_{ij}^h \in \{0, 1\}, \forall (i, j) \in A, \forall h \in H$.

Proof*: Consider* $(i, j) \in A$ *and an arbitrary* $h^* \in H$*. If* $g_{ij}^{h^*} = 0$*, where* $(i, j) \in A_{dc}$*, then* $x_{ij}^{h^*} = 0$ *by* (13) *and* (34)*. Otherwise, by* (28)*, we have* $g_{ij}^{h^*} = 1$*. Therefore, from* (14)*,* (26) *and* (28)*, we have that* $g_{ij}^h = 0$*, thus* $x_{ij}^h = 0, \forall h \neq h^*$*. Similarly, if* $y_i^{h^*} = 0$*, where* $i \in N_c$*, then* $x_{ij}^{h^*} = 0$*,* $\forall (i, j) \in A \setminus A_{dc}$ *by* (12) *and* (34)*. Otherwise, by* (27) *we have* $y_i^{h^*} = 1$*. Therefore, from* (15) *and* (27)*, we have that* $y_i^h = 0$*, thus* $x_{ij}^h = 0, \forall h \neq h^*$*. Thus, using* (11) *and* (27)*, we obtain* $x_{ij}^{h^*} = \sum_{h \in H} x_{ij}^h = z_{ij} \in \{0, 1\}$*. Therefore,* $x_{ij}^h = 0$*,* $\forall (i, j) \in A, \forall h \in H$.

3.2. Valid inequalities from the literature

As suggested in Kara et al. [17] and applied for homogeneous fleet in Schmidt et al. [29], constraints (6) can be lifted as in (35):

$$u_i - u_j + \max_{k \in K}\{Q_k\} z_{ij} + \left(\max_{k \in K}\{Q_k\} - q_i - q_j\right) z_{ji} \leq \max_{k \in K}\{Q_k\} - q_j, \quad \forall i, j \in N_c, i \neq j. \quad (35)$$

The problem can be further reduced in size by removing some variables associated with the departure interval of the vehicles. Based on Schmidt et al. [29], we can remove several arc traversal variables. We consider two cases as presented by constraints (36) and (37). Any arc (i, j) can be removed for the interval h, if the sum of the shortest time to traverse any incoming arc (from the depot or any other customers) to customer i and the service time required in i exceeds the upper bound of that interval, imposed by constraints (36). This logic is also true if the time to reach the closest customer to any depot in interval h and the service time required for the customer i exceeds the the upper bound of interval h, as imposed by constraints (37).

$$x_{ij}^h = 0 \quad \forall i \in N_c \mid \left(\min_{a \in (N_c \setminus \{i\}) \cup N_d} \{t_{ai}^h\} + s_i\right) \geq (h+1)\overline{T}, \quad \forall j \in N_c \cup N_t, j \neq i, \forall h \in H \quad (36)$$

$$x_{ij}^h = 0 \quad \forall i \in N_c \mid \left(\min_{\substack{a \in N_d \\ b \in N_c}} \{t_{ab}^h\} + s_i\right) \geq (h+1)\overline{T}, \quad \forall j \in N_c \cup N_t, j \neq i, \forall h \in H. \quad (37)$$

Similarly, Schmidt et al. [29] remove some variables related to the terminal nodes. The cases include when the traveling time of an arc to the depot in the last time interval (m) or when the sum of the shortest arrival time to a customer, its service time, and the shortest time it takes to reach a terminal from this customer exceeds the length of the interval:

$$x_{ij}^m = 0, \quad \forall i \in N_c, \forall j \in N_t \mid t_{ij}^m \geq \overline{T} \quad (38)$$

$$x_{ij}^m = 0, \quad \forall j \in N_c \mid \left(\min_{a \in N_c \setminus \{j\}} \{t_{aj}^m\} + s_j + \min_{b \in N_t} \{t_{jb}^m\}\right) \geq \overline{T}, \quad \forall i \in N_c, i \neq j. \quad (39)$$

Finally, we improve the routing part of the model by forbidding subtours of sizes two and three:

$$z_{ij} + z_{ji} \leq 1, \quad \forall i, j \in N_c, i \neq j \quad (40)$$

$$z_{ij} + z_{ji} + z_{iv} + z_{vi} + z_{jv} + z_{vj} \leq 2, \quad \forall i, j, v \in N_c, i \neq j, i \neq v, j \neq v. \tag{41}$$

3.3. New and problem-specific valid inequalities

First, we establish lower bounds for each type of cost from the objective function. By valid inequalities (42), we set a lower bound on the depot opening costs. Let o' be the minimum cost required to open depots. It is obtained as a solution of a variable cost and size bin packing problem [5], taking into account the demand of customers, the capacity of the depots and their opening fixed costs.

$$\sum_{i \in N_d} O_i w_i \geq o'. \tag{42}$$

Furthermore, a lower bound can be established on the fixed cost of using vehicles as in (43). Let f' be the minimum cost required to serve all customers, considering the capacity of different types of vehicles available and the demand of customers. It is also obtained as a solution of a variable cost and size bin packing problem.

$$\sum_{k \in K} \sum_{j \in N_c} \sum_{\substack{v \in \delta_k(i) \\ i \in N_d}} F_k z_{jv} \geq f'. \tag{43}$$

We also set a lower bound on the routing costs. Let k' be the minimum number of vehicles required to meet all demands considering the maximum vehicle capacity of $\max_{k \in K}\{Q_k\}$, also obtained as a solution of a bin packing problem.

First considering all intervals, we identify the minimum cost arc leaving each depot to reach every customer $s_{di} = \min_{h \in H}\{t^h_{di}\}$, $\forall d \in N_d, \forall i \in N_c$. Second, we do the same for each arc returning from every customer i to every depot d as $r_{id} = \min_{h \in H}\{t^h_{id}\}$. Then, we arrange all the $|N_c|$ values obtained for each depot d for s_{di} and r_{id} in an increasing order, defining s^n_{di} and r^n_{id} as the s_{di} and r_{id} values in the n^{th} ordered position. Moreover, for each $d \in N_d$, let us define $f^n_d = s^n_{di} + r^n_{id}$. Once again, we arrange all the values obtained for f^n_d in an increasing order, where f^n is the value in the n^{th} position. We then establish that at least the first k' values of the vector f^n will be considered as routing costs to leave and return to any depot. Finally, we set $g_c = \min_{\substack{a \in N_c \setminus \{c\} \\ h \in H}}\{t^h_{ca}\}, \forall c \in N_d \cup N_c$. We also arrange g_c in an increasing order. Therefore, g^n is defined as the value in the n^{th} ordering position. This term allows us to establish a lower

bound on routing costs for reaching customers.

$$\sum_{(i,j)\in A}\sum_{h\in H} Ct_{ij}^{h}x_{ij}^{h} \geq \sum_{n=1}^{k'} Cf^{n} + \sum_{n=1}^{|N_c|-k'} Cg^{n}. \tag{44}$$

The minimum number of vehicles (with the maximum capacity) k' is set as the lower bound for all departures from all depots.

$$\sum_{i\in N_d}\sum_{j\in N_c}\sum_{h\in H} x_{ij}^{h} \geq k' \tag{45}$$

$$\sum_{i\in N_d}\sum_{j\in N_c} z_{ij} \geq k'. \tag{46}$$

Finally, we can establish that at least one vehicle must leave from every selected depot:

$$\sum_{i\in N_d}\sum_{j\in N_c}\sum_{h\in H} g_{ij}^{h} \geq w_i. \tag{47}$$

4. Metaheuristic algorithm

The general structure of our proposed metaheuristic is based on evolutionary search methods. It is inspired by ideas proposed in Koç et al. [19], but adjusted for a time-dependent problem. The choice of the solution algorithm is mainly based on its success in dealing with location decision of depots as tactical/strategical level decisions in a multi-depot context (e.g., Canca and Barrena [3] and Zhou et al. [37]).

The algorithm encompasses three main phases: initialization, evolution, and intensification. In the first phase, we generate an initial population of solutions. The second phase deals with creating new solutions and selecting promising ones among them. The final phase comprises an additional process applied to improve the quality of the selected solutions. An overview is presented in Algorithm 1 and detailed as follows.

The initialization consists of creating initial solutions by assigning customers to depots and then constructing routes to serve them. Then, by using different operators, we populate the initial

solution set until it reaches η individuals. In the evolution phase, we perform crossover, mutation, and select the surviving population. When the number of iterations without improvement (δ) in the current population is reached, this phase ends and the intensification phase begins. Exchange operators, based on the demand and service time, are applied and we update the surviving population. When no further improvements are possible in any of surviving individuals, this step terminates. The final step is to apply an improvement heuristic. To do so, we select the best ϑ solutions and, if possible, try to improve the departure time of each vehicle and the sequence of the customers being served on the route. After this improvement phase, the solution cost is updated and the best solution is saved.

Algorithm 1 General structure of the metaheuristic algorithm

```
Generation of initial solutions.                                  //Initialization: Section 4.1
while initial population ≤ η do
    Increase the population by using initial solutions.
end while
while the number of iterations without improvement < δ do         //Evolution: Section 4.2
    Select a set of parents from the current generation.
    Apply crossover operators.
    Apply directed mutation operators.
    Update the surviving population.
end while
while there is an improvement in any current solution do          //Intensification: Section 4.3
    Apply exchange operators.
    Update the surviving population.
end while
for each one of the ϑ best solutions do
    Apply an improvement heuristic.
    Update the solution cost.
end for
Return the best solution.
```

4.1. Initialization phase

The initialization of our algorithm consists of creating a set of initial solutions by assigning customers to depots, followed by route construction.

The assignment is based on the travel time between each depot and a customer. As the travel time varies for each interval $h \in H$, different assignments can be generated. Therefore, we generate four time matrices based on: (i) the traveling time in each interval $h \in H$, (ii) maximum times, (iii) minimum times, and (iv) average travel times.

Given any of these four time matrices, we assign each customer $i \in N_c$ to its closest depot. This is performed based on the shortest travel time between the depot and every customer, for both departure and return. Then, we list the customers in a descending order of travel time. From the top of this list, we assign customers to their closest depot, considering the depot capacity constraints. This procedure continues until all customers are assigned. At this point, we try to improve the assignment, by closing depots. We first close one depot and try to assign its associated customers to all the other ones. The same procedure is applied for closing two depots.

After the allocation phase, we start the routes construction, for which only the largest capacity vehicles are used. Aiming to diversify the solutions, we apply two heuristics as follows.

In the first heuristics, the routes are constructed sequentially, i.e., once the capacity of the vehicle is reached for the first route, the second route can be started. For each depot, we randomly select a customer and insert it at the end of the current route. These selection and insertion processes are repeated until the route becomes infeasible, either due to exceeding the planning horizon or violating the vehicle capacity. When this occurs, a new route starts and we repeat the process until all customers are assigned to a route. This operation is repeated for all open depots and continues until the complete solution is generated. This process is repeated for each of the four time matrices.

The second heuristic uses the Clarke and Wright [4] algorithm to generate routes. However, instead of using a distance matrix, we use the four travel time matrices, as previously described. Note that the selected time matrix has an effect on the assignment of customers to depots, and now again, it affects the route generation.

In what we call the intra-route improvement step, a permutation procedure is applied, sequentially and iteratively. Once all the routes are generated, we apply an improvement procedure using either the full route permutation or partial permutations. In the full route permutation, all routes containing up to ν_1 customers are explored, and we determine the best sequence of customers by enumerating all permutations. Partial permutation, however, is applied to a subgroup of ν_2 customers. The position of these customers in this subgroup is modified until the best sequence is obtained. Then, the position of the first customer in the subgroup is fixed and

the same procedure is applied to the next ν_2 remaining customers. This process continues until all possible improvements for the entire route is taken place. A numerical example is presented in Figure 1. Assume that we have a route with customers 1, 2, 3, 4 and 5. With $\nu_2 = 3$, we first create a subgroup with customers 1, 2 and 3. After checking all 6 possibilities for their sequence, we identify $3-1-2$ as the optimal one for the subgroup. Now, we fix the position of the first customer in the subgroup, i.e., customer 3, and generate another subgroup with customers 1, 2 and 4. Again, we try all the possibilities for creating an optimal route within this subgroup. We find $4-2-1$ as the optimal route for our new subgroup. As before, we fix the position of the first customer in our subgroup, i.e., customer 4, and continue making a new subgroup as $2-1-5$ and find $5-1-2$ as the optimal route for this subgroup. Now, we create a new subgroup as $1-2-3$ which turns out to be the optimal route for this subgroup. The last step, is to generate a subgroup with customer 2 and the first two customers of the route, customers 3 and 4. All possibilities are checked and the optimal sequence is found as $4-2-3$. A full round is complete, therefore, now we add the complete solution to the initial population set.

Initial route	**1**	**2**	**3**	4	5
Partial optimal 1	**3**	**1**	**2**	4	5
	3	**1**	**2**	**4**	5
Partial optimal 2	3	**4**	**2**	**1**	5
	3	4	**2**	**1**	**5**
Partial optimal 3	3	4	**5**	**1**	**2**
	3	4	5	**1**	**2**
Partial optimal 4	3	4	5	**1**	2
	3	**4**	5	1	**2**
Local optimal	2	3	5	1	4

Figure 1: A numerical example for the intra-route partial permutations improvement step

In the second phase of the initialization algorithm, new solutions are generated to increase the

size of the initial population. For this purpose, large neighborhood search operators are applied to the initial solutions, with the selection of a removal and an insertion operator:

Removal operators:

- **Depot swap:** Applied by Hemmelmayr et al. [15] and Koç et al. [19], this operator randomly selects an open depot and closes it and then the operator also opens a closed depot. All customers assigned to the closed depot are then kept in a removal list L_r.
- **Depot opening:** This operator is an adaptation of the one used by Hemmelmayr et al. [15] and Koç et al. [19]. It randomly opens a closed depot. Then n' customers are removed from the current solution and added to the list L_r. The removal criterion is based on the shortest average travel time between the customer and the newly opened depot. To calculate this time, we consider the time to traverse the respective arcs, in both directions and for every time interval.
- **Random removal:** Used by Ropke and Pisinger [24] and Koç et al. [19], this operator randomly selects n' customers and adds them into the removal list L_r.
- **Depot time exchange:** It is based on the first removal operator, but differs from it in the criterion applied to choose the new open depot. This operator was originally proposed by Koç et al. [19], but we adapted it to consider the shortest average travel time between the closed and the open depot. As before, the calculation considers the arc that connects both depots in both directions and for every time interval.

Insertion operators:

- **Greedy insertion:** it is an adaptation of the operator applied by Ropke and Pisinger [24]. A customer is randomly selected from the list L_r, its insertion is tested into all possible positions for each route. The cost is also evaluated for new routes from already opened depots. We respect the capacity constraints of depots and vehicles so as to maintain feasibility.
- **Improved greedy insert:** This operator is based on the previous greedy operator, but differs from it since the insertion does not take place randomly. We test the position

that minimizes the insertion cost for all customers, as in Koç et al. [19]. Always ensuring feasibility, the insertion is conducted on existing and new routes. Then, the customer with the lowest insertion cost is assigned. After each insertion, the remaining non-assigned customers are reevaluated.

- **Regret insertion:** it is also an adaptation of the operator applied by Ropke and Pisinger [24] and it is performed to avoid a customer being assigned to a bad position in a route. This operator is more complicated than the previous ones, and requires a higher number of operations. For each customer, the regret is calculated based on the cost difference between the insertion in its best and second best positions. Then, the customer with the highest regret is selected. All possibilities are tested on existing and new routes, as long as they remain feasible. After an insertion, all customers from the removal list are reevaluated.

The number n' of customers removed is randomly chosen from a discrete uniform distribution from the interval $[n_l^{initial}, n_u^{initial}]$ that we calculate as percentages of the total number of customers in the instance. To create each new solution, we randomly select one of the solutions generated in the first phase of the initialization. The complete solution is added to the initial population. This process is repeated until the initial population reaches the size of η.

4.2. Evolution

In this phase, we improve and generate new solutions by using crossover and mutation operators. The goal is to diversify the search and improve the quality of the solutions created at each iteration.

To generate solutions by the crossover operator we apply the *Partially Mapped Crossover* [13]. Initially, two solutions (parents) are randomly selected, P_1 and P_2. Two-cut off points indicating the number of genes to be crossed are determined, which refers to the number of customers to be removed from the sequence. This number is randomly selected from a discrete uniform distribution from the interval $[n_l^{cross}, n_u^{cross}]$, where n_l^{cross} and n_u^{cross} are lower and upper bounds calculated as percentages of the total number of customers of the instance. Once the position of the first cut-off point is determined, the genes that are between the two points are crossed for

offspring generation. It is probable that during the genetic material exchange, chromosomes end up having repeated genes. In this case, all those outside the cutoff region are replaced with those on the same *locus*, but in another chromosome. Having generated the sequence of customers for each child solution, following the configurations of their parents with respect to the order of the sequence and the number of customers at the respective routes, these children are placed in routes. Finally, we check the feasibility of every solution and discard all the infeasible ones.

Mutation involves creating new children solutions by copying another solution to change one or more of the current chromosome genes. In order to improve the quality of each new generation, we use directed mutation, in form of an improvement heuristic. The procedure consists of two steps: a removal followed by an insertion. In the first step, n' customers are iteratively allocated to a removal list L_r. Again, n' is randomly selected from a discrete uniform distribution from the interval $[n_l^{mutation}, n_u^{mutation}]$, where $n_l^{mutation}$ and $n_u^{mutation}$ are calculated as percentages of the number of customers. During the insertion step, the removed customers are relocated in the incomplete solution. Both operators are randomly selected from the following ones.

Removal operators:

- **Neighborhood removal:** This operator is inspired by Ropke and Pisinger [24], Demir et al. [8], Koç et al. [19]. The general idea is to remove the n' customers that are "extreme" with respect to the travel time. We identify these customers by calculating the sum of the time required to arrive into each customer coming from its previous node and the time required to go from this customer to the subsequent node in the route. For each customer, we take into consideration the corresponding interval h associated to each inbound and outbound arc.

- **Worst travel time removal:** This operator removes the n' customers that are "extreme" with respect to the route insertion cost. By inserting a customer into a route, we change the costs associated with the new arc added to the route as well as the costs of all the subsequent arcs. Therefore, here, we define the insertion cost as the difference between the total execution time of a route with and without adding each customer. A distance-based version of this operator can be found in Ropke and Pisinger [24], Demir et al. [8],

and Koç et al. [19].

- **Depot removal - efficiency:** Proposed by Koç et al. [19], the aim of this operator is to calculate the utilization efficiency of each open depot, which is expressed by the ratio of the total demand allocated to the depot over its capacity. The depot with the least efficiency rate is removed from the solution and its customers are placed in the removal list L_r.
- **Vehicle removal:** Similar to the previous one, this operator computes the utilization efficiency of each vehicle. This value is expressed by the ratio of the total demand associated with a vehicle to its capacity. All customers associated with the least efficient vehicle are placed in the removal list L_r.

Insertion operators:

- **Greedy insertion – lowest cost:** This operator iteratively assigns customers to routes, in a position that minimizes the insertion cost. In this operator, before adding customer to routes, we try to improve the existing solution. Given a solution, we check the possibility of serving the routes with a small size vehicle (i.e., lower fixed cost) or of allocating the routes to an already open depot with no customers. Then for every customer in the L_r list, we compute its insertion cost, including additional fixed and opening costs. At each iteration, we identify the customer with the lowest insertion cost and add it to the route. This process continues until all customers from the list are added to a route.
- **Greedy insertion – highest cost:** This operator is similar to the previous one. The only difference is that here we insert the customer with the highest cost into a route.

The evolution and selection procedures are described next.

From the initial population $\Gamma_{initial}$, which has a size η, we randomly select two solutions, P_1 and P_2 and apply the crossover operator. If at least one of the two created children is feasible and new, its cost is computed and the solution is added to the current generation $\Gamma_{generation}$. We repeat this process until ω new individuals are created. Then, the mutation procedure is used. It is iteratively applied to the initial population solutions $\Gamma_{initial}$. We randomly select one removal and one insertion operator and apply them on an individual from the list. If this procedure

creates a new solution, it is added to the current generation $\Gamma_{generation}$. This procedure is applied either to the entire initial population or until η new individuals are created. Then, we combine $\Gamma_{initial}$ and $\Gamma_{generation}$ which will make the current population Γ. This population is ordered by cost, from the lowest to the highest. Therefore, the current population Γ can contain up to 2η ordered solutions, from which a set $\Gamma_{parents}$ is selected to constitute the next generation and play the role of parents in the next iteration of the algorithm.

The parents for the next generation are selected based on the following criterion. Given the current population Γ, we select the χ_1 best solutions, the χ_3 worst, and among the remaining solutions, we also randomly select χ_2 solutions, so that $\chi_1 + \chi_2 + \chi_3 = \eta$. We then apply the crossover and mutation operators to this population, as previously described.

Finally, to select the surviving population, we only save the η best solutions and the rest is discarded. To the current population Γ, we add the new generation $\Gamma_{generation}$. Once again, the individuals are ordered based on their costs and we repeat the process of selecting parents and creating a new generation. We continue this repetitive procedure until the stopping criterion is reached. The process terminates when there is no further improvement to any of the σ best solutions for δ consecutive iterations.

The evolution phase is described in Algorithm 2. At the end of this phase, the surviving population is reordered, and the best solutions are saved for the next phase.

4.3. Intensification

The last phase of our metaheuristic aims to intensify the search for the remaining solutions. This phase is divided into two main steps. The first one is called intra-route exchanges and includes an iterative process, similar to the one applied in the evolution phase. It involves two operators: one associated with customer demands and the other with the service time. The second step encompasses a heuristic that combines the best departure time for each vehicle with the most appropriate customer sequence for that moment. Note that we do not change the customer set that is served by the vehicle, but only its sequence.

Next, we elaborate on the operators for the first step and explain how they are applied.

Algorithm 2 Evolution

1: $\Gamma_{parents} \leftarrow \emptyset$; $\Gamma_{generation} \leftarrow \emptyset$
2: $\Gamma \leftarrow \Gamma_{parents} \leftarrow \Gamma_{initial}$ `//Use the initial population as the first generation`
3: **while** there is no improvement on the σ best solutions for δ iterations **do** `//Crossover`
4: $\tau \leftarrow 0$
5: **while** the number of feasible and new solutions $\leq \omega$ **do**
6: Randomly select P_1 and P_2 from $\Gamma_{parents}$ and apply the crossover operator.
7: **for** each new solution **do**
8: Improve routes with the intra-route improvement algorithm.
9: Add the solution to $\Gamma_{generation}$.
10: $\tau = \tau + 1$
11: **if** $\tau = \omega$ **then**
12: Go to line 16.
13: **end if**
14: **end for**
15: **end while**
16: **for** each solution in $\Gamma_{parents}$ **do** `//Mutation`
17: Randomly apply a removal and an insertion operator and apply.
18: **if** the generated solution is new **then**
19: Improve routes by the intra-route improvement algorithm.
20: Add the solution to $\Gamma_{generation}$.
21: **end if**
22: **end for**
23: Order Γ by the lowest cost and keep the η best solutions. `//Surviving population`
24: $\Gamma \leftarrow \Gamma_{generation}$; $\Gamma_{generation} \leftarrow \emptyset$.
25: Update $\Gamma_{parents}$.
26: **end while**

- **Demand-based exchange:** it aims to exchange customers with similar demands from different routes.
- **Service time exchange:** this operator is similar to the previous one except that customers are chosen based on similarity in their service time.

First, we apply the demand-based exchange operator to the surviving population Γ from the previous phase. The newly generated solutions are added to the population as new individuals. We order them based on their cost, and only keep the η best solutions in the population. To this population, we now apply the service time exchange operator and similarly, we only keep the η best solutions. This sequence of procedures is performed repeatedly until no further improvement can be reached.

When the first step finishes, we apply a departure time improvement procedure with the ordering sequence of customers. It is applied to the best ϑ individuals, and works as follows. Each route from a solution is handled individually. We start from the customer sequence saved during its generation, which considers the departure time $t = 0$. The total route duration is calculated as the difference between the moment the vehicle leaves the depot and the moment it returns to its terminal. The process is iterative, and at each iteration, we increase the departure time by Δ_t units. If the time interval $h \in H$ associated with some customer visits is changed, we apply the intra-route improvement algorithm. After this, if the route duration decreases, we replace the original customer sequence with the new one and update the departure time. We repeat this process until the departure time increase of Δ_t units causes the route to become infeasible, obtaining then the best departure time and customer sequence. After applying this heuristic to the best ϑ solutions, we identify the best solution, which is the output of our algorithm. The intensification phase is described in Algorithm 3. It is the final phase of our metaheuristic algorithm.

5. Computational experiments

In this section, we provide details of our instances, the setting and parameters of the algorithms and extensive results along with an elaborated analysis. The algorithms are coded in C++ and

Algorithm 3 Intensification

while number of consecutive iterations without improvement ≥ 0 **do** `//Intra-route exchanges`
  for each solution in Γ **do**
    Apply the demand-based operator
    if the solution is new **then**
      Improve routes by the intra-route improvement algorithm.
      Insert the solution in Γ.
    end if
  end for
  for each solution in Γ **do**
    Apply the service time-based operator
    if the solution is new **then**
      Improve routes by the intra-route improvement algorithm.
      Insert the solution in Γ.
    end if
  end for
  Order individuals in Γ by their cost and retain the best η solutions.
end while
for each ϑ solution from the best ones **do** `//Departure time improvement`
  for each route from the solution **do**
    $\tau_0 \leftarrow$ route execution time, take into account the departure time $t_s = 0$.
    $Seq \leftarrow$ Route customers sequence
    while $(t_s + \Delta_t + \sum_{\forall i \in N_c} s_i) \leq |H|\overline{T}$ **do**
      $t_s = t_s + \Delta_t$.
      Apply the intra-route improvement algorithm, take into account the departure time t_s.
      $\tau_1 =$ route execution time taking into account the departure time t_s.
      if $\tau_1 < \tau_0$ **then**
        $\tau_0 \leftarrow \tau_1$.
        $Seq \leftarrow$ Current route customers sequence.
      end if
    end while
    Update the route sequence to Seq and the time execution to τ_0.
  end for
  Update the complete solution cost.
end for
Return the best solution.

we use Gurobi Optimizer 8.1.1 as the mixed integer programming (MIP) solver, in its default settings. All computational experiments are conducted on an Intel Core i7 processor running at 3.4 GHz with 64 GB of RAM installed, with the Ubuntu Linux operating system. Two threads are used by the solver and a total time limit of 10800 seconds is imposed for each execution. Section 5.1 describes our instances and the results of detailed computational experiments are provided in Section 5.2.

- The initial population size η is defined as a function of the number customers in the instance: $\eta = 30\sqrt{|N_c|}$. This relation is defined to avoid overpopulating generations in large instances.

- The number of best solutions is a function of the initial population. Therefore, we set $\sigma = 0.1\eta$ for instances with 10 and 20 customers, $\sigma = 0.15\eta$ for instances with 50 customers, and $\sigma = 0.2\eta$ for all the other instances. In the intensification phase, we set it equal to $\vartheta = 0.05\eta$ in all configurations.

- For the intra-route improvement heuristic, we set ν_1 equal to 10, 8, 7, and 6, respectively, for instances with 10, 20, 50, and 80 or more customers; ν_2 equals to 3 for all configurations.

- For the $\Gamma_{parents}$ composition, we use $\chi_1 = 0.5\eta, \chi_2 = 0.4\eta$ and $\chi_3 = 0.1\eta$.

- The range defined for n', the number of customers removed, differs for each phase of the algorithm. In the initialization phase, it is set between $n_l^{initial} = 0.3|N_c|$ and $n_u^{initial} = 0.6|N_c|$, while in the evolution phase, the range is defined as $n_l^{cross} = 0.2|N_c|$ and $n_u^{cross} = 0.4|N_c|$. Finally, for the mutation operator, these bounds are defined by $n_l^{mutation} = 0.2|N_c|$ and $n_u^{mutation} = 0.5|N_c|$.

- The number of individuals created at each iteration is set to $\omega = 0.02\eta$.

We consider that all parameters have integer values. Hence, if necessary, we round the result to the nearest positive integer.

5.1. Instance generation

We modify the instances used in Schmidt et al. [29] to fit the fleet size and mix problem studied here. The instances are based on geographical information from the real road network and traffic of Quebec City. A planning horizon of 15 hours (from 6:00 to 21:00) is divided into three equal-length intervals of 3600, 5400, and 10800 seconds. These three intervals then differentiate the *large*, *medium*, or *small* instances. The number of customer equals to 10, 20, 50, 80, or 100, and the number of available depots for each instance is either 3 or 5. The demand and service time for each customer are random numbers chosen from $[50, 750]$ units and $[1000, 10800]$ seconds, respectively. We modify these instances as follows. We set the number of different types of vehicles in the fleet, $|K|$, to 3 for all instances. For the capacity Q_k of these types of vehicles, we consider 2000, 4000, and 6000 units, and the fixed cost, F_k, 1000, 1500, and 2000 monetary units, respectively. We randomly generate the depot capacity, W_i, from a discrete uniform distribution from the interval $[w_l^i, w_u^i]$. We define the lower and upper bounds as percentages of total customer demand, set at 50% and 85%, respectively. The opening cost of each potential depot is proportional to its capacity. We round all values to the nearest integer, if needed. For each unit of travel time, we set the cost of 1 monetary unit. All instances, solutions, and detailed results area available from `https://www.` [1].

5.2. Computational results

We now present the results of our extensive computational experiments. We start our analysis by showing in Section 5.2.1 the results from the basic mathematical model from Section 3. Then, we compare these results with the ones in which all valid inequalities are added. In Section 5.2.2, we provide detailed results using our metaheuristic presented in Section 4. Finally, we evaluate the performance of the metaheuristic algorithm to solve also a special case without departure time optimization, a single depot, homogeneous and limited fleet, and without fixed costs, the TD-LRP [29], in Section 5.2.3.

[1] omitted for double blind review

5.2.1. Results of the mathematical model

We now present the average results obtained by solving the proposed mathematical formulation of Section 3, when provided with a pool of initial solutions as follow. We save the top ten solutions obtained at the initialization phase as input (initial solution) for the MIP model presented in Section 3. However, we provide these solutions to the solver only partially: we only provide the solver with the the arcs traversed, and not the departure time (variables z_{ij}).

Table 2 presents the results for 3 and 5 potential depots. On the first two columns of this table, we provide information about the instance. Then, for each different number of depots, we report information about the best initial solution (Best IS), upper bound (UB), lower bound (LB), gap calculated as $100(UB - LB)/UB$, execution time, and finally the improvement over the initial solution.

The results on Table 2 show that, typically, the solver is not able to significantly improve the initial solutions. Even after 3 hours, the improvements are merely marginal. These improvements are 0.68% for instances with 3 and 0.67% in cases with 5 potential depots. This table also shows a very large difference between the upper and lower bounds, even in instances with a few customers, with an average gap of over 98% for both sets with 3 and 5 depots. Detailed results (available online) reveal that without an initial solution the solver is not even able to obtain any feasible solution for most instances.

In what follows, we analyze the average results from our mathematical model when it is fed the pool of initial solutions and under the presence of all valid inequalities. Table 3 presents the results.

The results show that providing valid inequalities does not have any significant effect on the upper bounds, leading to mostly the same solutions compared to the ones reported in Table 2. An improvement of only 1.25% is obtained for 3-depots instances, whereas it is even lower (0.22%) for 5-depots instances. However, as expected, adding valid inequalities significantly improves the lower bounds and consequently reduces the gap. As indicated on the LB gap (%) in Table 3, the percentage difference between these lower bounds and those presented in Table 2 is, on average, greater than 7000% for both depot configurations. On columns Gap

Table 2: Average results for the TD-FSMLRP model with initial solutions and without valid inequalities

Size	$\|N_c\|$	3 depots						5 depots					
		Best IS	Upper Bound	Lower Bound	Gap (%)	Time (s)	Improvement over IS (%)	Best IS	Upper Bound	Lower Bound	Gap (%)	Time (s)	Improvement over IS (%)
Small	10	19472.20	19470.16	245.74	98.74	10800	0.01	18881.36	18730.10	252.16	98.65	10800	0.80
	20	21528.62	21059.10	296.43	98.59	10800	2.18	20989.08	20591.74	281.64	98.63	10800	1.89
	50	30971.66	30093.24	434.93	98.55	10800	2.84	28523.28	28514.50	429.31	98.49	10800	0.03
	80	42254.04	42251.80	499.26	98.82	10800	0.01	33435.64	33222.34	516.38	98.45	10800	0.64
	100	46364.52	46357.64	593.43	98.72	10800	0.01	40630.02	40627.98	526.95	98.70	10800	0.01
Average		32118.21	31846.39	413.96	98.68	10800	1.01	28491.88	28337.33	401.29	98.59	10800	0.67
Medium	10	19386.58	19219.82	235.41	98.78	10800	0.86	17521.32	17353.52	247.14	98.58	10800	0.96
	20	20612.26	20342.42	297.73	98.54	10800	1.31	20387.44	19593.20	282.33	98.56	10800	3.90
	50	30746.82	30744.38	429.67	98.60	10800	0.01	29686.72	29682.10	421.98	98.58	10800	0.02
	80	40148.22	40139.98	494.12	98.77	10800	0.02	35680.82	35677.32	511.59	98.57	10800	0.01
	100	44787.70	44785.66	590.08	98.68	10800	0.00	40410.62	40302.64	518.80	98.71	10800	0.27
Average		31136.32	31046.45	409.40	98.67	10800	0.44	28737.38	28521.76	396.37	98.60	10800	1.03
Large	10	19933.54	19933.22	220.96	98.89	10800	0.00	18047.92	17912.66	229.14	98.72	10800	0.75
	20	21267.84	20657.16	294.04	98.58	10800	2.87	20158.48	19996.26	277.99	98.61	10800	0.80
	50	32559.74	32557.36	426.08	98.69	10800	0.01	28507.60	28502.62	423.89	98.51	10800	0.02
	80	43036.14	43035.78	484.74	98.87	10800	0.00	37869.68	37868.94	505.73	98.66	10800	0.00
	100	48460.52	48460.10	587.97	98.79	10800	0.00	37098.88	37094.86	517.88	98.60	10800	0.01
Average		33051.56	32928.72	402.76	98.76	10800	0.58	28336.51	28275.07	390.92	98.62	10800	0.32
Global average		32102.03	31940.52	408.71	98.71	10800	0.68	28521.92	28378.05	396.19	98.60	10800	0.67

Table 3: Average results for TD-FSMLRP model with initial solutions and valid inequalities

Size	$\|N_c\|$	3 depots						5 depots					
		Upper Bound	Lower Bound	Gap (%)	Time (s)	Over the solutions without VIs UB gap (%)	Over the solutions without VIs LB gap (%)	Upper Bound	Lower Bound	Gap (%)	Time (s)	Over the solutions without VIs UB gap (%)	Over the solutions without VIs LB gap (%)
Small	10	19470.12	19468.58	0.01	271	0.00	7823.08	18728.92	18727.14	0.01	1640	0.01	7327.49
	20	21048.04	20764.90	1.35	10800	0.05	7000.47	20573.02	20084.04	2.38	10800	0.09	7204.61
	50	28424.90	25441.34	10.50	10800	5.54	6435.44	28298.42	24680.06	12.79	10800	0.76	6491.55
	80	42251.12	30108.24	28.74	10800	0.00	8362.79	33331.76	27684.38	16.94	10800	-0.33	6354.94
	100	46261.06	33675.34	27.21	10800	0.21	7695.54	40438.78	31703.36	21.60	10800	0.47	7574.11
Average		31491.05	25891.68	13.56	8694	1.16	7463.46	28274.18	24575.80	10.74	8968	0.20	6990.54
Medium	10	19210.94	19202.00	0.05	4021	0.05	8060.51	17345.62	17036.62	1.78	8414	0.05	6918.43
	20	20496.26	19480.18	4.96	10800	-0.76	6784.29	19477.52	18849.30	3.23	10800	0.59	6798.88
	50	28827.64	26244.48	8.96	10800	6.23	6609.24	29127.76	24341.82	16.43	10800	1.87	6802.60
	80	40138.74	29414.42	26.72	10800	0.00	8023.28	35678.50	28886.62	19.04	10800	0.00	6874.01
	100	44786.80	33375.44	25.48	10800	0.00	7490.00	40401.04	32330.12	19.98	10800	-0.24	7687.35
Average		30692.08	25543.30	13.23	9444.16	1.10	7393.47	28406.09	24288.90	12.09	10323	0.45	7016.25
Large	10	19929.04	19927.04	0.01	2193	0.02	8919.39	17904.84	17893.76	0.06	4995	0.04	7714.04
	20	20632.06	19857.84	3.75	10800	0.12	6916.84	19987.68	19309.78	3.39	10800	0.04	7090.12
	50	30250.62	25987.70	14.09	10800	7.09	6999.74	28502.94	24272.92	14.84	10800	0.00	6624.19
	80	43034.34	29491.00	31.47	10800	0.00	8777.84	37857.86	29679.78	21.60	10800	0.03	7385.78
	100	48360.74	34329.74	29.01	10800.00	0.21	8125.10	37095.92	31362.32	15.46	10800	0.00	7063.03
Average		32441.36	25918.66	15.67	9078.64	1.49	7947.78	28269.85	24503.71	11.07	9639	0.02	7175.43
Global average		31541.49	25784.55	14.15	9072.32	1.25	7601.57	28316.71	24456.13	11.30	9643.29	0.22	7060.74

(%), we can observe gap values equal to 14.15% and 11.30%, for instances with 3 and 5 depots, respectively. These results highlight the importance of the valid inequalities to improve lower bounds. Detailed results (available online) also show that for instances with 10 customers, the solver is able to prove optimality in 24 of the 30 test instances in an average execution time of less than 30 minutes. However, it is obvious that even by providing a pool of initial solutions and strengthening the mathematical model by a set of valid inequalities, the average gap still remains very high after three hours of execution for other instances. This reflects the complexity of the problem and the need to apply approximate methods.

5.2.2. Results of the proposed metaheuristic

Table 4 shows the average results from our metaheuristic algorithm, including the best solution obtained in each of the three phases. We present the best solution from initialization, evolution, and intensification phases, followed by the total execution time. Finally, on the last two columns for each depot configuration, we present the improvement of our metaheuristic over solutions from the mathematical model with initial solution and valid inequalities (results from Table 3).

Several interesting observations can be drawn from the analysis of the results in Table 4. First, we observe how each metaheuristic phase improves the quality of solutions. From the first to the second phases, it is possible to improve the average quality of the solutions by 6.15% and 10.66%, respectively for instances with 3 and 5 potential depots. From the second to the third phases, this improvement is less important as it represents 0.07% and 0.11%, respectively for 3-depots and 5-depots instances.

These results show that our metaheuristic approach is able to find, on average, better solutions than the MIP for all configurations. The improvements are very remarkable, 11.66% for the instances with 3 depots and 8.25% for the ones with 5 depots. Except for the instances with 10 customers in which the average obtained solution slightly worsens (less than 0.01%), we are able to notably improve the solutions. Compared with the solutions obtained by the mathematical formulation with valid inequalities, we can observe that the our method is able to improve the results by more than 27% in some cases (instances with 80 customers, 3 depots and 15 time intervals).

Table 4: Average results from metaheuristic algorithm

Size	$\|N_c\|$	3 depots						5 depots					
		Best solution for each phase			Total	Improvement over MIP		Best solution for each phase			Total	Improvement over MIP	
		Initial.	Evolution	Intensif.	time (s)	UB (%)	Time (%)	Initial.	Evolution	Intensif.	time (s)	UB (%)	Time (%)
Small	10	19476.10	19473.10	19470.98	1	0.00	99.66	18843.46	18732.13	18729.00	1	0.00	99.95
	20	21191.26	21040.53	21036.27	73	0.06	99.32	21080.99	20551.60	20550.00	61	0.11	99.43
	50	28177.99	26269.03	26258.13	626	7.62	94.20	28958.45	25590.05	25577.54	616	9.61	94.30
	80	34381.91	31635.18	31612.37	1308	25.18	87.89	34121.83	29276.20	29246.88	1197	12.26	88.91
	100	38746.15	35702.21	35663.62	3006	22.91	72.16	38149.60	33937.98	33886.33	2982	16.20	72.39
Average		28394.68	26824.01	26808.27	1003	11.15	90.65	28230.87	25617.59	25597.95	971	7.64	91.00
Medium	10	19232.28	19217.76	19212.24	1	0.00	99.98	17365.76	17353.68	17345.60	1	0.00	99.99
	20	20678.35	20180.91	20173.18	58	1.58	99.47	20055.76	19135.14	19130.09	111	1.78	98.97
	50	28271.79	26937.38	26916.69	622	6.63	94.24	30025.17	25161.38	25125.20	639	13.74	94.08
	80	33998.18	31159.46	31122.35	1571	22.46	85.45	35111.77	30445.73	30325.92	1626	15.00	84.95
	100	38673.21	35150.86	35109.44	3325	21.61	69.21	40676.74	34131.95	34103.65	6149	15.59	43.06
Average		28170.76	26529.27	26506.78	1116	10.45	89.67	28647.04	25245.58	25206.09	1705.16	9.22	84.21
Large	10	19936.55	19934.66	19929.82	1	0.00	99.96	18122.48	17911.40	17903.89	1	0.01	99.98
	20	20694.09	20363.83	20350.94	67	1.36	99.38	20567.17	19604.09	19595.24	98	1.96	99.09
	50	28211.13	26448.56	26427.48	547	12.64	94.94	28051.20	25489.62	25474.58	590	10.62	94.54
	80	35639.04	31197.58	31173.14	1904	27.56	82.37	36961.78	31411.72	31363.00	1074	17.16	90.06
	100	39819.06	36159.28	36113.08	3108	25.33	71.22	39750.69	33517.50	33475.33	3410	9.76	68.43
Average		28859.97	26820.78	26798.89	1125	13.38	89.57	28690.66	25586.87	25562.41	1034.46	7.90	90.42
Global average		28475.14	26724.69	26704.65	1081	11.66	89.96	28522.86	25483.35	25455.48	1237	8.25	88.54

Considering the time needed to achieve these improvements, the effectiveness of the proposed method is even better highlighted. The results from Table 4 show that the metaheuristic approach is able to provide better solutions considerably faster. For instances with 3 depots, the average execution time is 1081 seconds, whereas for cases with 5 potential depots it equals to 1237 seconds, against more than 9000 seconds on average for the MIP. These results highlight the importance of the metaheuristic approach in both aspects: to find high quality solutions in significantly lower execution times.

We can however combine the strengths of both methods: high quality solutions from our metaheuristic and tight lower bounds of the MIP, thanks to our valid inequalities. We present in Table 5 an aggregated gap by comparing these values.

Table 5: Average results of the metaheuristic solution and lower bound of the MIP

Size	$\|N_c\|$	3 depots			5 depots		
		Metaheuristic solution	Lower bound	Gap (%)	Metaheuristic solution	Lower bound	Gap (%)
Small	10	19470.98	19468.58	0.01	18729.00	18727.14	0.01
	20	21036.27	20764.90	1.29	20550.00	20084.04	2.27
	50	26258.13	25441.34	3.11	25577.54	24680.06	3.51
	80	31612.37	30108.24	4.76	29246.88	27684.38	5.34
	100	35663.62	33675.34	5.58	33886.33	31703.36	6.44
Average		26808.27	25891.68	2.95	25597.95	24575.80	3.51
Medium	10	19212.24	19202.00	0.05	17345.60	17036.62	1.78
	20	20173.18	19480.18	3.44	19130.09	18849.30	1.47
	50	26916.69	26244.48	2.50	25125.20	24341.82	3.12
	80	31122.35	29414.42	5.49	30325.92	28886.62	4.75
	100	35109.44	33375.44	4.94	34103.65	32330.12	5.20
Average		26506.78	25543.30	3.28	25206.09	24288.90	3.26
Large	10	19929.82	19927.04	0.01	17903.89	17893.76	0.06
	20	20350.94	19857.84	2.42	19595.24	19309.78	1.46
	50	26427.48	25987.70	1.66	25474.58	24272.92	4.72
	80	31173.14	29491.00	5.40	31363.00	29679.78	5.37
	100	36113.08	34329.74	4.94	33475.33	31362.32	6.31
Average		26798.89	25918.66	2.89	25562.41	24503.71	3.58
Global average		26704.65	25784.55	3.04	25455.48	24456.13	3.45

As presented, the resulting average gap for instances with 3 potential depots is only 3.04%, and it is only 3.45% for instances with 5 potential depots. Combining the information from good heuristic solutions and tight gaps, we have been able to provide several solutions with a gap of less than 0.01% for both sets with 3 and 5 depots.

5.2.3. Metaheuristic algorithm performance on the TD-LRP

We now evaluate the performance of our metaheuristic on the TD-LRP. To this end, we compare our results with the ones obtained by Schmidt et al. [29]. We consider two situations: first when all vehicles leave the depot at the beginning of the planning horizon, and second when the vehicles have flexibility with respect to the departure time. To adapt our proposed method for the first situation (i.e., fixed departure time), we remove the heuristic allowing the vehicles to depart at any time from the intensification phase. Thus, all routes start at the beginning of the time horizon. In the TD-LRP, the vehicle fleet is homogeneous, therefore, no fixed cost with respect to vehicles is considered and a single depot is chosen to serve all the customers. All these adaptations are incorporated to respect the original problem definition of the TD-LRP.

Tables 6 and 7 report the average results for instances with 3 and 5 potential depots, respectively. In each table, we compare the performance of our proposed algorithm against the one proposed specifically for the TD-LRP [29]. In the first two columns of each table, we provide information about the instances. Then we show their best solution and the execution time. The next four columns contain the average results of our proposed metaheuristic. We first present the best solution and the execution time without flexibility in the departure time and then, for the case with flexibility. Then, we present the improvement gained by our algorithm with fixed departure time consideration compared to the solutions from the literature, both in terms of solution quality and processing time. Finally, in the last column, we present the improvement in the objective function when our metaheuristic considers the flexibility time departure compared to the one without flexibility.

The results of Table 6 show the superiority of our metaheuristic approach. Better quality solutions are obtained for all instances, regardless of the number of customers and the size of the instance. The average improvement is 2.69% for small, 2.37% for medium, and 2.05% for large instances. This improvement can even go up to 5.45% for small instances with 10 customers. More detailed results (available online) also show that our metaheuristic is able to find an optimal solution for all instances with 10 customers and 3 depots. In addition, the required execution time is significantly smaller, providing an average reduction of 49%, when compared to the matheuristic approach of Schmidt et al. [29]. In addition, results of our

Table 6: Average results from metaheuristic approach of Section 4 to solve the TD-LRP [29] with and without flexibility for instances with 3 depots

Size	$\|N_c\|$	Matheuristic [29]		Metaheuristic of Section 4						
		Best solution	Time (s)	Best solution	Time (s)	Improvement (%) UB	Improvement (%) Time	Solution with departure flexibility	Time with departure flexibility (s)	Improvement with flexibility (%)
Small	10	7436.00	38	7030.40	8	5.45	79.50	6661.60	6	5.25
	20	13182.40	113	12860.80	64	2.44	43.31	12733.20	19	0.74
	50	23624.20	739	22955.20	380	2.83	48.63	22672.20	545	1.16
	80	33212.80	2710	32714.00	1083	1.50	60.05	32448.80	1113	1.38
	100	39759.80	5462	39267.40	2574	1.24	52.87	38671.60	2783	1.10
Average		23443.04	1813	22965.56	822	2.69	56.87	22637.48	893	1.93
Medium	10	7413.40	36	7252.60	15	2.17	58.86	6942.80	19	4.27
	20	13248.40	92	12742.80	49	3.82	46.37	12585.00	22	1.51
	50	23451.60	883	22710.80	485	3.16	45.01	22380.00	441	1.51
	80	34467.20	2512	34004.40	1188	1.34	52.71	33404.60	1294	1.90
	100	38704.20	4775	38170.20	1915	1.38	59.91	37169.20	2087	2.41
Average		23456.96	1660	22976.16	730	2.37	52.57	22496.32	772	2.32
Large	10	7378.00	34	7153.00	32	3.05	5.87	6848.00	37	4.26
	20	13328.80	89	12934.00	64	2.96	28.09	12550.60	19	3.21
	50	23261.60	845	22631.40	503	2.71	40.43	22272.60	532	1.49
	80	33240.80	2362	32983.80	815	0.77	65.48	32466.60	932	1.72
	100	38691.20	4630	38391.20	2412	0.78	47.91	37373.40	2137	2.50
Average		23180.08	1592	22818.68	765	2.05	37.56	22302.24	731	2.64
Global average		23360.03	1688	22920.13	772	2.37	49.00	22478.68	799	2.29

metaheuristic applied to the TD-LRP with flexible departure time show that it is possible to improve the travel time by 1.93% for small, 2.32% for medium, and 2.64% for large instances with 3 depots compared with the results for fixed departure time.

Similarly, the results from Table 7 show that our metaheuristic algorithm can produce better average solutions for all classes of instances. Improvements of 1.76%, 1.71% and 1.99% are obtained, for small, medium and large instances, respectively. The detailed results show that our metaheuristic is also able to find optimal solutions for 14 out of 15 instances with 10 customers. Moreover, for all the other instances our result is better than the one generated by the proposed matheuristic. With respect to the execution time, we also observe that our metaheuristic is noticeably faster, an average reduction of 59.40%. Furthermore, by relaxing the departure time of the vehicles from the depot, the solutions improve further. By using our metaheuristic to solve the TD-LRP with departure time flexibility, we obtain an improvement of 2.21% for small, 3.80% for medium, and 2.59% for large instances.

In summary, the results show how our proposed metaheuristic algorithm adapted to the TD-LRP can produce better quality solutions and require substantially less processing time. Out

Table 7: Average results from metaheuristic approach of Section 4 to solve the TD-LRP [29] with and without flexibility for instances with 5 depots

Size	$\|N_c\|$	Matheuristic [29]		Metaheuristic of Section 4						
		Best solution	Time (s)	Best solution	Time (s)	Improvement (%) UB	Improvement (%) Time	Solution with departure flexibility	Time with departure flexibility (s)	Improvement with flexibility (%)
Small	10	7767.80	64	7604.00	63	2.11	1.70	7249.20	50	4.67
	20	12073.20	200	11771.80	73	2.50	63.72	11454.80	17	2.95
	50	23107.40	1354	22700.20	794	1.76	41.33	22530.00	572	0.84
	80	30723.00	4169	30213.60	897	1.66	78.49	29733.60	1308	1.50
	100	33290.60	9933	33031.80	2195	0.78	77.90	32464.00	2785	1.12
Average		21392.40	3144	21064.28	804	1.76	52.63	20686.32	946	2.21
Medium	10	7833.00	56	7788.40	14	0.57	74.86	6956.20	13	10.69
	20	12609.20	160	12176.20	64	3.43	60.04	11963.80	17	2.26
	50	22985.40	1543	22598.80	621	1.68	59.79	22171.00	574	1.94
	80	30604.20	3082	29890.00	1035	2.33	66.40	29444.00	959	1.49
	100	33065.00	9365	32892.80	2434	0.52	74.01	32372.20	1849	2.62
Average		21419.36	2841	21069.24	834	1.71	67.02	20581.44	682	3.80
Large	10	7479.80	56	7355.40	26	1.66	53.64	6624.60	32	10.96
	20	12376.60	157	11954.80	76	3.41	51.38	11458.40	17	4.75
	50	23406.00	1504	22923.60	849	2.06	43.53	22508.20	701	1.43
	80	31151.60	3820	30408.40	1125	2.39	70.55	29814.80	1414	1.84
	100	33185.20	9613	33037.00	2530	0.45	73.68	32422.40	2091	1.64
Average		21519.84	3030	21135.84	921	1.99	58.56	20565.68	851	2.59
Global average		21443.87	3005.10	21089.79	853	1.82	59.40	20611.15	827	2.87

of 150 instances, our method is able to provide 5 equal and 131 better solutions.

6. Conclusions

This paper studies the time-dependent location-routing problem with fleet size and mix decisions. Its main contribution is to extend the time-dependent literature to include more real-world features to this very practical problem. This paper also contributes to the integrated optimization literature as it presents the first mathematical formulation for the TD-FSMLRP. Using a commercial solver to solve instances that are generated from the real traffic data of Quebec city, we evaluate instances with up to 5 potential depots, 100 customers and 15 time intervals. We show that adding a pool of initial solutions and considering several problem-specific valid inequalities is extremely important to obtain and improve the lower bounds for the TD-FSMLRP. However, to achieve high quality solutions and to reduce the computational time, we propose a metaheuristic algorithm based on exploring a population of solutions. We have compared the performance of our proposed algorithm against the exact method. We have shown the importance of an approximate intricate approach to solve complex problems such as

the one studied in this paper. Our metaheuristic is able to find high quality solutions for large size instances, and also to reduce significantly the execution time. Comparing our solutions with the dual bounds, obtained with the help of our valid inequalities, shows a very tight gap even for very large instances. We have also evaluated our method on a special case of the problem from the literature, significantly improving the solution quality and run time.

Acknowledgments

"Information omitted for double blind review".

References

[1] D. Ambrosino, A. Sciomachen, and M.G. Scutellà. A heuristic based on multi-exchange techniques for a regional fleet assignment location-routing problem. *Computers & Operations Research*, 36 (2):442–460, 2009.

[2] M. Boccia, T.G. Crainic, A. Sforza, and C. Sterle. Multi-commodity location-routing: Flow intercepting formulation and branch-and-cut algorithm. *Computers & Operations Research*, 89: 94–112, 2018.

[3] D. Canca and E. Barrena. The integrated rolling stock circulation and depot location problem in railway rapid transit systems. *Transportation Research Part E: Logistics and Transportation Review*, 109:115–138, 2018.

[4] G. Clarke and J.W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581, 1964.

[5] T.G Crainic, G. Perboli, W. Rei, and R. Tadei. Efficient lower bounds and heuristics for the variable cost and size bin packing problem. *Computers & Operations Research*, 38(11):1474–1482, 2011.

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

[7] E. Demir, T. Bektaş, and G. Laporte. A comparative analysis of several vehicle emission models for road freight transportation. *Transportation Research Part D: Transport and Environment*, 16 (5):347–357, 2011.

[8] E. Demir, T. Bektaş, and G. Laporte. An adaptative large neighborhood search heuristic for the pollution-routing problem. *European Journal of Operational Research*, 223:346–359, 2012.

[9] M. Drexl and M. Schneider. A survey of variants and extensions of the location-routing problem. *European Journal of Operational Research*, 241(2):283–308, 2015.

[10] O. Dukkanci, B.Y. Kara, and T. Bektaş. The green location-routing problem. *Computers & Operations Research*, 105:187–202, 2019.

[11] M. A. Figliozzi. The time dependent vehicle routing problem with time windows: Benchmark problems, an efficient solution algorithm, and solution characteristics. *Transportation Research Part E: Logistics and Transportation Review*, 48(3):616–636, 2012.

[12] M. Gendreau, G. Ghiani, and E. Guerriero. Time-dependent routing problems: a review. *Computers & Operations Research*, 64:189–197, 2015.

[13] D.E. Goldberg and R. Lingle. Alleles, loci and the traveling salesman problem. In *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, pages 154–159, Pittsburg - United States, July 1985.

[14] B. Golden, A. Assad, L. Levy, and F. Gheysens. The fleet size and mix vehicle routing problem. *Computers & Operations Research*, 11(1):49–66, 1984.

[15] V.C. Hemmelmayr, J.-F. Cordeau, and T.G. Crainic. An adaptive large neighborhood search heuristic for the two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research*, 39:3215–3228, 2012.

[16] H. Heni, L.C. Coelho, and J. Renaud. Determining time-dependent minimum cost paths under several objectives. *Computers & Operations Research*, 105:102–117, 2019.

[17] İ. Kara, G. Laporte, and T. Bektaş. A note on the lifted Miller–Tucker–Zemlin subtour elimination constraints for the capacitated vehicle routing problem. *European Journal of Operational Research*, 158(3):793–795, 2004.

[18] Ç. Koç, T. Bektaş, O. Jabali, and G. Laporte. The fleet size and mix location-routing problem with time windows: formulations and a heuristic algorithm. *European Journal of Operational Research*, 248(1):33–53, Jan. 2016.

[19] Ç. Koç, T. Bektaş, O. Jabali, and G. Laporte. The impact of depot location, fleet composition and routing on emissions in city logistics. *Transportation Research Part B: Methodological*, 84: 81–102, 2016.

[20] R.V. Kulkarni and P.R. Bhave. Integer programming formulations of vehicle routing problems. *European Journal of Operational Research*, 20(1):58–67, 1985.

[21] R. Lahyani, L.C. Coelho, and J. Renaud. Alternative formulations and improved bound for the multi-depot fleet size and mix vehicle routing problem. *OR Spectrum*, 40:125–157, 2018.

[22] C. Malandraki and M.S. Daskin. Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. *Transportation Science*, 26(3):185–200, 1992.

[23] J. Renaud and F.F. Boctor. A sweep-based algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, 140(3):618–628, 2002.

[24] S. Ropke and D. Pisinger. An adaptative large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40:455–472, 2006.

[25] S. Salhi and M. Fraser. An intergrated heuristic approach for the combined location vehicle fleet mix problem. *Studies in Locational Analysis*, 8:3–22, 1996.

[26] M. Savelsbergh and T. Van Woensel. 50th anniversary invited articlecity logistics: Challenges and opportunities. *Transportation Science*, 50(2):579–590, 2016.

[27] M. Schiffer and G. Walther. The electric location routing problem with time windows and partial recharging. *European Journal of Operational Research*, 260(3):995–1013, 2017.

[28] M. Schiffer and G. Walther. Strategic planning of electric logistics fleet networks: a robust location-routing approach. *Omega*, 80:31–42, 2018.

[29] C.E. Schmidt, A.C.L. Silva, M. Darvish, and L.C. Coelho. The time-dependent location-routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 128:193–315, 2019.

[30] M. Schneider and M. Drexl. A survey of the standard location-routing problem. *Annals of Operations Research*, 259(1-2):389–414, 2017.

[31] M. Schneider and M. Löffler. Large composite neighborhoods for the capacitated location-routing problem. *Transportation Science*, 53(1):301–318, 2019.

[32] D. Taş, N. Dellaert, T. van Woensel, and T. De Kok. The time-dependent vehicle routing problem with soft time windows and stochastic travel times. *Transportation Research Part C: Emerging Technologies*, 48:66–83, 2014.

[33] G. Van der Heide, P. Buijs, K.J. Roodbergen, and I.F.A Vis. Dynamic shipments of inventories in shared warehouse and transportation networks. *Transportation Research Part E: Logistics and Transportation Review*, 118:240–257, 2018.

[34] E. Von Boventer. The relationship between transportation costs and location rent in transportation problems. *Journal of Regional Science*, 3(2):27–40, 1961.

[35] T.-H. Wu, C. Low, and J.-W. Bai. Heuristic solutions to multi-depot location-routing problems. *Computers & Operations Research*, 29(10):1393–1415, 2002.

[36] Y. Xiao and A. Konak. The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion. *Transportation Research Part E: Logistics and Transportation Review*, 88:146–166, 2016.

[37] L. Zhou, R. Baldacci, D. Vigo, and X. Wang. A multi-depot two-echelon vehicle routing problem with delivery options arising in the last mile distribution. *European Journal of Operational Research*, 265(2):765–778, 2018.