

# Optimization

A Journal of Mathematical Programming and Operations Research

ISSN: 0233-1934 (Print) 1029-4945 (Online) Journal homepage: [www.tandfonline.com/journals/gopt20](http://www.tandfonline.com/journals/gopt20)

## The time-dependent multi-depot fleet size and mix green vehicle routing problem: improved adaptive large neighbourhood search

Mahdi Alinaghian, Maryam Jamshidian & Erfan Babaee Tirkolaee

To cite this article: Mahdi Alinaghian, Maryam Jamshidian & Erfan Babaee Tirkolaee (2022) The time-dependent multi-depot fleet size and mix green vehicle routing problem: improved adaptive large neighbourhood search, Optimization, 71:11, 3165-3193, DOI: [10.1080/02331934.2021.2010078](https://doi.org/10.1080/02331934.2021.2010078)

To link to this article: <https://doi.org/10.1080/02331934.2021.2010078>



Published online: 19 Dec 2021.



Submit your article to this journal [↗](#)



Article views: 629



View related articles [↗](#)




View Crossmark data [↗](#)



Citing articles: 9 View citing articles [↗](#)



# The time-dependent multi-depot fleet size and mix green vehicle routing problem: improved adaptive large neighbourhood search

Mahdi Alinaghian<sup>a</sup>, Maryam Jamshidian<sup>a</sup> and Erfan Babaee Tirkolaee <sup>b</sup>

<sup>a</sup>Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran;

<sup>b</sup>Department of Industrial Engineering, Istinye University, Istanbul, Turkey

## ABSTRACT

This study presents a mathematical model for the multi-depot Time-Dependent Fleet Size and Mix Green Vehicle Routing Problem (TD-FSMGVRP). The objective function of the developed model is to minimize the total cost including vehicles' fixed cost, drivers' cost, fuel costs, and costs of Greenhouse Gas (GHG) emission. Fleet composition, load, vehicle speed, road slope, and traffic are considered as factors affecting the produced pollution. Considering the NP-Hard complexity of this problem, an Improved Adaptive Large Neighbourhood Search (IALNS) algorithm is designed to treat the problem efficiently. The performance of the proposed algorithm is enhanced using the Taguchi design method. Finally, Adaptive Large Neighbourhood Search (ALNS) algorithm and Variable Neighbourhood Search (VNS) algorithm are considered as two well-known algorithms to test the efficiency of the IALNS algorithm using benchmark problems. Furthermore, a statistical test is conducted to efficiently provide the required comparisons for large-sized problems. It is revealed that the proposed IALNS has a superior performance and can appropriately tackle the problem. Finally, the impacts of the proposed model on cost-saving are evaluated using the proposed IALNS algorithm.

## ARTICLE HISTORY

Received 20 June 2021


Accepted 12 November 2021

## KEYWORDS

Time-dependent vehicle routing problem; multiple depots; green vehicle routing problem; improved adaptive large neighbourhood search; variable neighborhood search

## 1. Introduction

The rapid rate of urbanization in the recent decades has led to a massively amplified demand for transportation and has consequently imposed many problems including traffic congestion, air pollution and increased pollutant emission on large cities and their administrations. Recently, green logistics has been of great interest to companies and organizations. This concept is concerned with the analysis of environmental impacts of various distribution strategies, energy consumption reduction, fuel consumption, and pollutant emission [1]. Vehicle

**CONTACT** Erfan Babaee Tirkolaee  [erfan.babaee@istinye.edu.tr](mailto:erfan.babaee@istinye.edu.tr)

This article has been republished with minor changes. These changes do not impact the academic content of the article.

Routing Problem (VRP) which falls into the category of NP-Hard problems, plays a central role in logistics and product distribution [2]. The volume of pollution produced by a fleet of vehicles is influenced by vehicle characteristics such as speed and maximum load as well as fleet composition and size, and also environmental factors such as road slope and traffic congestion. Vehicles with optimal speed produce less environmental pollutants. The choice of vehicle type affects the rate of fuel consumption, as smaller vehicles consume less fuel than larger ones. However, using a large vehicle in place of two smaller vehicles may reduce the overall fuel consumption. Demir et al. [3] have stated that the difference between the fuel consumptions of medium- and heavy-duty vehicles travelling a distance of 100 km with a given load may reach 14 litres.

In urban road networks, travel time usually depends on the time of departure, and speed variations are triggered either by hourly, daily, and weekly patterns of traffic or unexpected events such as accidents. Thus, giving due consideration to the dependence of travel time on the time of departure can profoundly affect the quality of solutions. Despite the high practicality of the Time-Dependent Vehicle Routing Problem (TDVRP), they have only been rarely addressed in the literature. First-In-First-Out (FIFO) is the most important feature of TDVRP. This feature is denoted as follows: when a vehicle departs node  $i$  at time  $t_i$ , its time of arrival at the destination node is always less (sooner) than the case where it departs node  $i$  at time  $t'_i > t_i$  [4]. Most of the preceding research works that have provided a mathematical model for TDVRP or its real applications have not considered this feature. In some cases, companies use several depots with separately optimized tours to distribute their products. However, an integrated model for single-depot problems may result in more ideal solutions.

This study provides a novel mathematical model for Time-Dependent Fleet Size and Mix Green Vehicle Routing Problem (TD-FSMGVRP). The Objective function of the proposed model tries to minimize the vehicles' fixed costs, drivers' costs, fuel costs, as well as Greenhouse Gas (GHG) emissions. In the proposed model, GHG emissions and fuel consumption are calculated by the use of Comprehensive Modal Emission Model (CMEM) [5]. FIFO feature is also incorporated into the provided model. Despite the applicability of this problem, to the best of our knowledge, no such model has been constructed with these assumptions. Taking into account the NP-Hard complexity of the problem, a meta-heuristic algorithm based on Adaptive Large Neighbourhood Search (ALNS) entitled Improved Adaptive Large Neighbourhood Search (IALNS) is proposed as the main solution technique. The efficiency of the IALNS is enhanced using the Taguchi design method to tune its parameters. Then, the performance of the IALNS algorithm is assessed against the exact method, basic ALNS, and Variable Neighbourhood Search (VNS) algorithm using benchmark instance problems. Using a statistical test, it is revealed that the proposed IALNS algorithm is the best tool to tackle the complexity of the problem. Finally, the impacts of the model on cost-saving are evaluated.

The rest of the paper is structured as follows. Section 2 reviews the literature of the problem briefly. Section 3 introduces the problem and its resulting mathematical model. The applied solution algorithms are described in Section 4. Section 5 presented the computational results, and finally, Section 6 concludes the study and discussed future research directions.

## 2. Survey of the literature

The earliest notable study in VRP literature is the work of Dantzig and Ramser [6], who investigated a large-scale TSP and provided a solution for this problem. Clark and Wright [7] were the first authors who considered more than one vehicle in the problem formulation. Therefore, that work can also be regarded as the first well-known VRP study. Malandraki et al. [8] studied TDVRP and TDTSP problems and developed Mixed-Integer Linear Programming (MILP) model with time windows, vehicle capacity and waiting time at customer location. They calculated the travel time by the use of stepwise functions and described the travel time between two customers as a function of the distance between them and of time at which travel takes place. The nearest neighbour random (NNR) heuristic algorithm was also proposed by these authors. They then employed the cutting plane heuristic to solve small 10–25 node problems. Hill and Benton [9] examined a TDVRP problem with no time windows and provided a time-dependent travel velocity model. These authors also suggested a simple competitive heuristic to solve the problem. Lack of due attention to FIFO feature is the major weakness of the above-mentioned models. Ichoua et al. [10] built a travel velocity model for the TDVRP problem with a FIFO feature which minimized the total weighted sum of time and delay. They divided the day into three time periods in which velocity was changing based on traffic congestion. The model was assessed by Tabu Search (TS) algorithm in static and dynamic environments. Haghani and Jung [11] took into account the travel time as a continuous function and suggested a Genetic Algorithm (GA) to tackle the TDVRP problem. This algorithm was validated by an exact algorithm for problem instances up to 30 customers. Chen et al. [12] employed an MILP model to solve the TDVRP problem by a heuristic algorithm which included some route improvement methods. The main weakness of this work was that they did not take the FIFO feature into account. In Malandraki et al. [8], the travel time for each arc is a function of departure time, but Balseiro et al. [13] studied the TDVRP problem with a step-wise travel time distribution and FIFO feature. They hybridized Ant Colony Algorithm (ACO) and insertion heuristic to treat the TDVRP problem with time windows (TDVRPW). For the Capacitated Vehicle Routing Problem (CVRP), ACO found unfeasible solutions with non-routed customers in the final stages, but this issue was resolved through combining this algorithm with the insertion heuristic at the last steps. Furthermore, several local searches were used to optimize the solutions.

The notable studies on the subject of Multi-Depot Vehicle Routing Problems (MDVRPs) are briefly mentioned below. Jin et al. [14] solved the MDVRP as a zero-one programming problem and presented two solution techniques. The first one divides the problem into two sub-problems of allocation and independent routing, and treats it using a two-stage procedure. The second method integrates the allocation and independent routing sub-problems. In the end, it was concluded that the integrated method arrives at better solutions than the two-stage procedure. Tsirimpas et al. [15] addressed a multi-product MDVRP assuming that customer sequence is predetermined and solved the problem with a dynamic programming algorithm tailored for optimal routing policy. Liu et al. [16] solved MDVRP with a heterogeneous fleet of vehicles and developed a mathematical model for their assumptions. The objective function of this model sought to minimize the travel of empty vehicles. They also utilized a greedy algorithm to tackle the problem. Gulczynski et al. [17] presented a Split Delivery Vehicle Routing Problem (SDVRP) and solved it by an integer programming heuristic. The aim of the model was to minimize the distance travelled by vehicles. This minimization was realized using the concept of split delivery. Salhi et al. [18] studied the MDVRP with a heterogeneous fleet of vehicles based on a given number of vehicles at each depot. In their model, vehicle load was measured for any delivery and empty vehicles were guaranteed to return to the depot. They solved the problem using a VNS algorithm with new features and compared the results with 26 instances provided by the literature, and reported that this method managed to obtain better solutions for 23 instances.

The main studies on the subject of GVRP are as follows. Kuo [4] added travel distance, load weight and travel velocity to the fuel consumption model of TDVRP, generalized it for TDVRP, and used Simulated Annealing (SA) algorithm to find the solution. Xiao et al. [19] formulated the fuel consumption model by adding the fuel consumption rate (FCR)-as a function of vehicle load- to the CVRP model and then reconstructed the CVRP model in order to minimize the fuel consumption. They took into consideration both travel distance and time as the factors affecting the rate of fuel consumption. Moreover, FCR was regarded to be a load-dependent linear function. Madden et al. [20] examined the TDVRP problem with traffic congestion and reported 7% reduction in CO<sub>2</sub> emission after routing based on different time-dependent velocities. However, their model mainly focused on travel time minimization rather than emissions minimization. Bektas and Laporte [21] developed a PRP model with/without time windows, whose objective function was to minimize the cost corresponding to CO<sub>2</sub> emission, fuel consumption and driver cost. Their model assumed that the minimum velocity is 40 km/h, which is inconsistent with real traffic congestions. Bektas and Koc [22] studied a pollution-routing problem with a heterogeneous fleet of vehicles to minimize the routing cost, fuel cost and pollutants emission. Recently, Karakostas et al. [23] designed Adaptive Variable Neighbourhood Search (AVNS) algorithms to solve the fleet size and mix location inventory routing problem

considering pollution minimization as well as Just-in-Time replenishment policy and Capacity Planning. They evaluated the performance of the algorithms against CPLEX in small-scale problems.

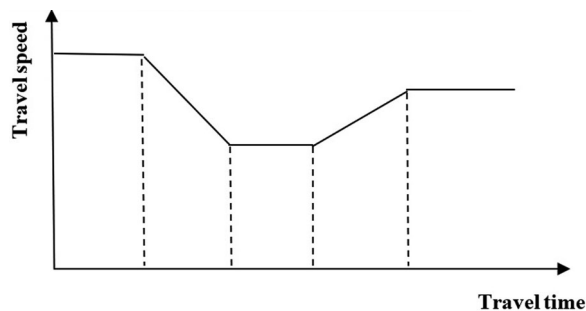
The present study provides a novel MILP model and efficient algorithm for the multi-depot TD-FSMGVRP. Despite the practicality of assumptions constituting this problem, to the best of the authors' knowledge, the present work is the first research on this particular subject.

### 3. Problem statement

This study offers a novel mathematical model for the multi-depot TD-FSMGVRP. The developed model also incorporates the assumption of varying traffic congestion over the transportation network. This means that vehicle speed and travel time depend on the time of departure. FIFO is a realistic feature of TDVRP, and is described as follows: when a vehicle departs node  $i$  (source node) at time  $t_i$ , it always arrives sooner at the destination node than the case where it departs node  $i$  at time  $t'_i > t_i$ . In this study, the FIFO feature is incorporated using a piecewise linear function for calculating the speed (or time) of travel over the arc. It has been proven that the FIFO feature can be emulated by a piecewise linear function with a slope greater than or equal to 1. The application of such a function also causes the speed variation over the time horizon to be uniform, which provides a more realistic condition. This uniformity feature cannot be achieved by conventional stepwise functions. An example of a piecewise linear function is shown in Figure 1.

On the other hand, in some cases, companies may use multiple depots with separately optimized tours to distribute their products. Here, a multi-depot TDVRP can provide more ideal solutions by following an integrated approach.

The results of a study by Demir et al. [5] on macroscopic and microscopic models have shown that estimates of comprehensive modal emission model (CMEM) are closer to reality, so this model is more suitable for applications such as the one intended in this study. Therefore in this study, fuel consumption is



**Figure 1.** Piecewise linear function used in travel speed calculations.

calculated by the CMEM model, in which emission, speed, vehicle load and road slope are studied.

The United States Federal Highway Administration (USFHA) <https://www.sciencedirect.com/science/article/pii/S0191261514001623-b0115> has classified vehicles into three main categories based on the Gross Vehicle Weight Rating (GVWR): light-duty ( $h = 1$ ), medium-duty ( $h = 2$ ), and heavy-duty ( $h = 3$ ) [24]. With regards to the rules of CMEM model, the fuel consumption rate of vehicle  $h$  can be obtained from Equation (1):

$$FR^h = \xi (k^h N^h V^h + P^h / \eta) / \kappa, \quad (1)$$

where  $\xi$  represents the fuel-to-air mass ratio,  $K$  denotes the engine friction factor,  $N$  is the vehicle speed,  $V$  denotes the engine displacement,  $\eta$  and  $\kappa$  are constant and denote the diesel engine efficiency and heating value of diesel fuel, respectively. Moreover,  $P$  is the engine's real-time output power (kW) and is expressed as

$$P^h = P_{tract}^h / \eta_{tf}^h + P_{acc}, \quad (2)$$

where  $\eta_{tf}$  denotes the vehicle's drive train efficiency,  $P_{acc}$  is the power drawn by accessories such as air conditioner, etc. Moreover,  $P_{tract}$  denotes the tractive power of vehicle wheels (kW) and can be obtained from the following equation:

$$P_{tract}^h = (M^h a + M^h g \sin \theta + 0.5 C_d^h \rho A v^2 + M^h g C_r \cos \theta) v / 1000, \quad (3)$$

where  $M$  is the vehicle weight, composed of two parts: empty vehicle weight ( $W$ ) and load weight ( $f$ ). Here,  $\alpha$  denotes vehicle acceleration ( $\text{m/s}^2$ ). Furthermore,  $v$ ,  $\theta$  and  $g$  are vehicle speed (m/s), road slope and gravitational constant, respectively. Finally,  $C_d$  and  $C_r$  denote coefficients related to air and rolling resistance, respectively, and  $\rho$  and  $A$  stand for air density ( $\text{kg/m}^3$ ) and vehicle frontal surface area ( $\text{m}^2$ ).

For the arc ( $i, j$ ) of length  $d$ ,  $v$  is the speed at which a vehicle travels over that arc. When all variables in Equation (1) except speed are supposed to be constant, fuel consumption over the arc can be computed as

$$F^h = k^h N^h V^h \lambda d_{/v} + P^h \lambda \gamma^h d_{/v}, \quad (4)$$

such that  $\lambda$  and  $\gamma$  are expressed as follows.

$$\lambda = \xi / \kappa \psi, \quad (5)$$

$$\gamma^h = 1 / 1000 n_{tf}^h \eta, \quad (6)$$

where  $\Psi$  is a factor for converting fuel consumption rate from gram per second to litres per second. Moreover,  $\alpha$  and  $\beta$  can be calculated from:

$$\alpha = a + g \sin \theta + g C_r \cos \theta, \quad (7)$$

$$\beta^h = 0.5 C_d^h \rho A^h. \quad (8)$$

Indices related to arc  $(i, j)$  will be placed on speed, distance, vehicle load and  $\alpha$  of that arc. Equation (4) can be rewritten as Equation (9) [24]:

$$F^h = \lambda(k^h N^h V^h d_v + M^h \gamma^h \alpha d + \beta^h \gamma^h v^2 d), \quad (9)$$

where  $k^h N^h V^h d_v$  represents the engine module as a linear expression in the travel time. Moreover,  $M^h \gamma^h \alpha d$  stands for the weight module and  $\beta^h \gamma^h v^2 d$  denotes the speed module. The speed module is quadratic in speed.

Parameters common among all vehicles and vehicle-specific parameters are presented in Tables 1 and 2, which have been also employed by Bektaş and Laporte [21], Koç et al., [22] and Demir et al. [24]. For detailed information, please see Barth and Boriboonsomsin et al. [25].

The optimal speed of a medium-duty vehicle travelling a distance of 100 km is calculated by Equations (10) and (11). The optimal speed is the speed at which fuel cost is at a minimum. Equations (10) and (11) are achieved by equating the derivative of Equation (9) to zero [22]:

$$\frac{\partial(F(v))}{\partial(v)} = 0 \rightarrow \frac{-kNV\lambda d}{v^2} + 2\beta\gamma\lambda dv = 0. \quad (10)$$

**Table 1.** Vehicle common parameters.

Parameter	Description	Typical value
$\xi$	Fuel-to-air math ratio	1
$g$	Gravitational constant ( $\text{m/s}^2$ )	9.81
$\rho$	Air density ( $\text{kg/m}^3$ )	1.2041
$C_r$	Coefficient of rolling resistance	0.01
$\eta$	Efficiency parameter for diesel engines	0.01
$f_c$	Fuel and $\text{CO}_2$ emission cost	1.4
$f_d$	Driver wage	0.0022
$\kappa$	Heating value of typical diesel fuel ( $\text{kJ/gr}$ )	44
$\psi$	Conversion factor	737
$a$	Acceleration ( $\text{m/s}^2$ )	0

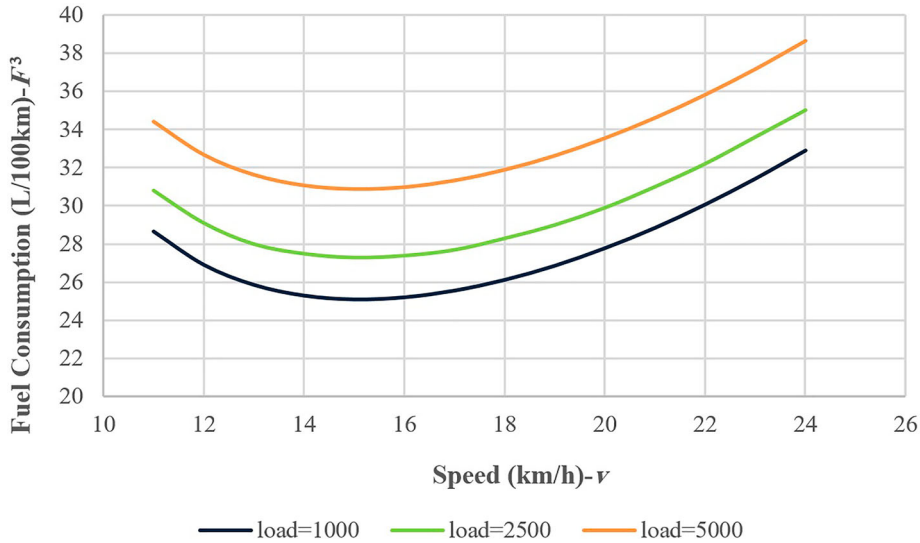
**Table 2.** Vehicle specific parameters.

Parameter	Description	Light duty ( $h = 1$ )	Medium duty ( $h = 2$ )	Heavy duty ( $h = 3$ )
$W^h$ (kg)	Curb weight	4672	6328	13,154
(kg) $Q_h$	Maximum payload	2585	5080	17,236
$f_h$	Vehicle fixed cost	41.68	59.9	93.92
$k^h$ (KJ / rev / litre)	Engine friction factor	0.25	0.2	0.15
$N^h$ (rev/s)	Engine speed	39	33	30.2
$V^h$ (lit)	Engine displacement	2.77	5	6.66
$C_d^h$	Coefficient of aerodynamics drag	0.6	0.6	0.7
$A^h$ ( $\text{m}^2$ )	Frontal surface area	9	9	9.8
$\eta_{tf}^h$	Vehicle drive train efficiency	0.4	0.45	0.5



**Table 3.** Optimal speed of each type of vehicle (m/s).

Vehicle type	Optimal speed
Light-duty (L)	13.83
Medium-duty (M)	15.27
Heavy-duty (H)	18.05

**Figure 2.** Effects of vehicle load on fuel consumption.

With regards to Equation (10), the optimal speed is written as

$$v^* = \sqrt[3]{\frac{kNV}{2\beta\gamma}}. \quad (11)$$

Using Equations (6), (8) and (11) and Tables 1 and 2, the optimal speed of each type of vehicle is presented in Table 3.

Figure 2 shows the fuel consumed by medium-duty vehicles hauling different masses of load over a distance of 100 km. As can be seen, as vehicle load increases, vehicle engine consumes more fuel to produce more power required for hauling. Assuming that vehicle load is the only factor affecting the fuel consumption, placing customers with higher demand at the beginning of the route can reduce the fuel consumption. In Figure 2, for a constant speed, fuel consumption increases with the increase of vehicle load.

The assumptions of the model are stated in the following subsection.

### 3.1. Model assumptions

The main assumptions of the model are given as follows:

- Distribution is carried out through multiple depots.
- Each vehicle starts and ends its tour at the same depot.
- Distribution is carried out using multiple types of vehicles. Vehicles have their own parameters including capacity, fixed cost, etc.
- Every available vehicle is in standard condition and its fuel system is properly adjusted.
- Demand of each customer is smaller than the capacity of the largest vehicle.
- Optimum number and composition of vehicles are uncertain and are determined by the model.
- Demands of customers are constant.

Based on the above assumptions, parameters and variables of the provided model are given below. In the next step, the objective function, constraints and linearization of the model will be discussed.

### 3.2. Sets, parameters and variables

The sets and parameters used in the formulations are described below:

$N$	Set of customer and depot vertices with related indices, $i, j$ , and $p$ ,
$N_C$	Set of customer vertices,
$N_D$	Set of depot vertices,
$H$	Set of vehicles with related index $h$ ,
$M$	Set of time periods with related index $m$ ,
$R$	Set of subinterval speeds with related index $r$ ,
$dis_{ij}$	Distance of arc $(i, j)$ ,
$g_{ij}$	Road slope of arc $(i, j)$ ,
$d_i$	Demand of customer $i$ ,
$LB_m$	Lower bound of time $m$ ,
$UB_m$	Upper bound of time $m$ ,
$v_m$	Speed at the beginning of time period $m$ ,
$s_i$	Serving time for customer $i$ ,
$vr_r$	Mean speed of subinterval $r$ ,
$vm_{ij}^{hm}$	Maximum speed that vehicle $h$ can traverse the arc between customers $i$ and $j$ according to the traffic congestion at time period $m$ and arrival time,
$B$	A large number.

Variables are indicated as follows:

$x_{ij}^{hm}$	Binary variable indicating whether the arc between two customers $i$ and $j$ is traversed by vehicle $h$ at time period $m$ or not,
$fb_{ij}^h$	Load of vehicle $h$ traversing the arc between customers $i$ and $j$ ,
$v_{ij}^{hm}$	Speed of vehicle $h$ traversing the arc between customers $i$ and $j$ at time period $m$ ,
$z_{ij}^{rhm}$	Binary variable indicating whether the arc between two customers $i$ and $j$ is traversed by vehicle $h$ with subinterval speed $r$ at time period $m$ or not,
$t_i$	Arrival time at customer $i$ .

Besides the above parameters, Tables 1 and 2 represent the remaining parameters related to fuel consumption.

### 3.3. Mathematical model

$$\begin{aligned}
 \text{minimize } Z = & \sum_{i \in N} \sum_{j \in N} \sum_{h \in H} \sum_{m \in M} \lambda f_c k_h N_h V_h \text{dis}_{ij} / v_{ij}^{hm} x_{ij}^{hm} \\
 & + \sum_{i \in N} \sum_{j \in N} \sum_{h \in H} \sum_{m \in M} \lambda f_c \gamma_h g(\sin(g_{ij})) \\
 & + C_r \cos(g_{ij}) \text{dis}_{ij} (w_h x_{ij}^{hm} + f_{ij}^{hm}) \\
 & + \sum_{i \in N} \sum_{j \in N} \sum_{h \in H} \sum_{m \in M} \lambda f_c (0.5 C_d^h A_h \rho) \gamma_h \text{dis}_{ij} (v_{ij}^{hm})^2 \\
 & + \left( \sum_{i \in N} \sum_{j \in N} \sum_{h \in H} \sum_{m \in M} \text{dis}_{ij} / v_{ij}^{hm} + \sum_{i \in N_c} s_i \right) f_d \\
 & + \sum_{j \in N_D} \sum_{j \in N_c} \sum_{h \in H} \sum_{m \in M} x_{ij}^{hm} f_h
 \end{aligned}$$

$$\text{subject to} \quad (12)$$

$$\sum_{i \in N_D} \sum_{j \in N_c} \sum_{m \in M} x_{ij}^{hm} \leq 1 \quad \forall h \in H, \quad (13)$$

$$\sum_{i \in N} \sum_{h \in H} \sum_{m \in M} x_{ij}^{hm} = 1 \quad \forall j \in N_c, \quad (14)$$

$$\sum_{i \in N} \sum_{m \in M} x_{ip}^{hm} = \sum_{j \in N} \sum_{m \in M} x_{pj}^{hm} \quad \forall p \in N, h \in H, \quad (15)$$

$$\sum_{m \in M} \sum_{j \in N} \sum_{i \in N} d_i x_{ij}^{hm} \leq Q_h \quad \forall h \in H, \quad (16)$$

$$\sum_{j \in N} \sum_{h \in H} f b_{ji}^h - \sum_{j \in N} \sum_{h \in H} f b_{ij}^h = d_i \quad \forall i \in N_c, \quad (17)$$

$$d_j \sum_{m \in M} x_{ij}^{hm} \leq f b_{ij}^h \leq (Q_h - d_i) \sum_{m \in M} x_{ij}^{hm} \quad \forall i, j \in N, h \in H, \quad (18)$$

$$t_i = 0 \quad \forall i \in N_D, \quad (19)$$

$$t_i + s_i + \text{dis}_{ij} / v_{ij}^{hm} \leq t_j + B(1 - x_{ij}^{hm}) \quad \forall i, j \in N_c, h \in H, m \in M, \quad (20)$$

$$t_i + s_i + \text{dis}_{ij} / v_{ij}^{hm} \geq t_j - B(1 - x_{ij}^{hm}) \quad \forall i, j \in N_c, h \in H, m \in M, \quad (21)$$

$$LB_m x_{ij}^{hm} \leq t_i + s_i \leq UB_m x_{ij}^{hm} \quad \forall i, j \in N, h \in H, m \in M, \quad (22)$$

$$v_m^{hm} = v_m x_{ij}^{hm} + \frac{(v_{m+1} - v_m)(t_i + s_i - LB_m) x_{ij}^{hm}}{(UB_m - LB_m)}$$

$$\forall i, j \in N, h \in H, m \in M, \quad (23)$$

$$v_{ij}^{hm} \leq vm_{ij}^{hm} \quad \forall i, j \in N, h \in H, m \in M, \quad (24)$$

$$x_{ij}^{hm} \in \{0, 1\}, fb_{ij}^h, t_i, v_{ij}^{hm} \geq 0 \quad \forall i, j \in N, h \in H, m \in M. \quad (25)$$

Equation (12) is the objective function of the provided model that is minimization type and consists of 5 parts. The first part estimates the fuel cost pertaining to vehicles' equipment. The second part is related to fuel cost arising from vehicle weight and its load. The third part represents the fuel cost due to velocity, the fourth part estimates the driver wage, and the fifth part is related to the vehicle fixed cost. Constraint (13) states that every vehicle must start its travel from one of the depots. Constraint (14) states that every customer must be served exactly once. Based on Constraint (15), a vehicle must exit from the same node it has been entered. Constraint (16) states that the total demand of customers positioned on a given tour must be less than the vehicle capacity. Constraint (17) states that when a vehicle visits a node, the difference between its loads before and after that visit is equal to demand at that node.

Constraint (18) ensures that when a vehicle visits a node, vehicle load at arrival is greater than or equal to demand at a node and vehicle load at departure is less than or equal to vehicle capacity minus demand at that node. Constraints (19)–(21) calculate the time at which servicing starts. Constraint (22) ensures that vehicle  $h$  can travel across arc  $(i, j)$  at time period  $m$  only when departure time from node  $i$  is at time interval related to period  $m$ . Constraints (23) and (24) calculate the speed of the vehicle traversing the arc  $(i, j)$ . Constraint (25) shows the types of the model variables.

### 3.3.1. Linearization of the provided model

In the provided formulation, a continuous variable is multiplied by a binary one, a parameter is divided by a continuous variable, and a continuous variable is raised to the second power, and the presence of these terms and expressions in the formulation indicates that this model is nonlinear. In the following subsections, this model will be linearized.

**3.3.1.1. Multiplication of a continuous variable by a binary variable.** In the provided model, the product of  $t_i$  and  $x_{ij}^{hm}$  in Constraint (23) is the multiplication of a continuous variable by a binary one. Suppose that variables  $x$  and  $y$  are continuous and binary variables, respectively. The term  $xy$ , can be linearized by replacing the multiplication operation of these two variables with continuous variable  $z$

while also adding the following constraints to the model [26]:

$$\begin{aligned} z &\leq x, \\ z &\leq By, \\ z &\geq x - B(1 - y). \end{aligned} \quad (26)$$

**3.3.1.2. Dividing a parameter by a continuous variable and raising a continuous variable to the second power.** Equations (12), (20) and (21), in which variable  $v_{ij}^{hm}$  is in the denominator or is raised to the second power are linearized using a piecewise linear method. In this method, the entire range of possible vehicle speeds will be partitioned into a number of subintervals and, vehicle on each arc can only move at a speed equal to the mean of one of these subintervals. At first, the following constraints are added to the model:

$$\sum_{r \in R} vr_r z_{ij}^{rhm} \leq B x_{ij}^{hm} \quad \forall i, j \in N, h \in H, m \in M, \quad (27)$$

$$\sum_{r \in R} vr_r z_{ij}^{rhm} \leq v_{ij}^{hm} \quad \forall i, j \in N, h \in H, m \in M, \quad (28)$$

$$z_{ij}^{rhm} \in \{0, 1\} \quad \forall i, j \in N, h \in H, m \in M, r \in R. \quad (29)$$

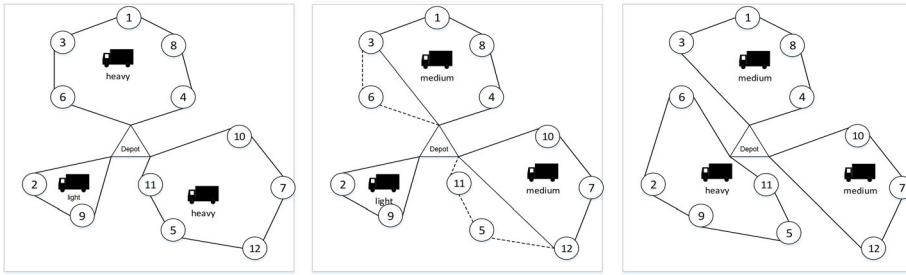
Constraint (27) states that if arc  $(i, j)$  is traversed by vehicle  $h$  at time period  $m$  then  $\sum_{r \in R} vr_r z_{ij}^{rhm}$  can take a non-zero value. Constraint (28) specifies the upper limit of speed, and Constraint (29) is related to the domain of the defined variable. Substituting  $\sum_{r \in R} z_{ij}^{rhm} / vr_r$  instead of  $v_{ij}^{hm}$  into expressions including  $1/v_{ij}^{hm}$  linearizes the model. Also,  $v_{ij}^{hm}$  raised to the second power is replaced with  $\sum_{r \in R} vr_r^2 z_{ij}^{rhm}$ .

## 4. Solution algorithms

This section describes the methods proposed for solving the problem. At first, the solution representation is presented, and then the proposed algorithm and the VNS meta-heuristic are explained. Finally, the process of parameter adjustment by the Taguchi design method is explained.

### 4.1. Initial solution

To produce an initial solution, it will be first assumed that all vehicles are light-duty due to their lower fuel consumption rate. Then a vehicle will be chosen and a random customer will be assigned to this vehicle. The customer closest to customer(s) on the tour will be then added to the tour. This process will continue until none of the remaining unallocated customers can be allocated to the tour. It should be noted that each tour is allocated to the depot having the lowest allocation cost.



**Figure 3.** An example of the destroy and repair operators.

Constraints (13)–(15) are never violated due the steps followed in generating solutions, but the total cost incurs a large penalty in case of violating Constraints (16)–(24) and (27)–(28). Therefore, just feasible solutions are generated and kept.

#### 4.2. Adaptive large neighbourhood search

ALNS is an extension of Large Neighbourhood Search Algorithm (LNSA), with its only difference being the use of a number of destroy and repair operators. LNSA was first introduced by Shaw [27]. Recently, LNSA-based meta-heuristic algorithms have shown their ability to produce outstanding results for transportation scheduling problems. LNSA examines a number of neighbourhoods and enhances the quality of an initial solution periodically by a set of destroy and repair operators. The destroy operator destructs a part of the solution, and then repair operator remakes it in another form. Here, the feasibility of solutions is maintained by applying penalty costs. For an instance, a simple destroy operator randomly removes 15% of customers and a repair operator reinserts them into a solution based on a greedy heuristic. Figure 3 shows an example of destroy and repair operators for a 12-customer problem.

#### 4.3. Improved adaptive large neighbourhood search algorithm

The IALNS algorithm employs a Tabu list concept. This Tabu list prevents the customers selected in the last iteration to be removed in the current iteration. The pseudo-code of the IALNS used in this paper is shown in Figure 4. In the following pseudo-code, the sets  $w^{current} = (1, \dots, 1)$  and  $Nu = (0, \dots, 0)$  are the initial weights of operators and the number of iterations in which operators are used, respectively. The IALNS pseudo-code is displayed in Figure 4.

In line with the above description, the execution steps of the IALNS algorithm are described below:

##### 4.3.1. Generating an initial solution

The process of generating an initial solution was explained in Subsection 4.1.

**Algorithm 1:** The framework of the proposed IALNS

---

**Input :** a set of destroy operators  $D$ , a set of repair operator  $R$ ,  $w^{current} = (1, \dots, 1)$ , cooling rate  $h$  and  $Nu = (0, \dots, 0)$ ;

**Output:**  $X_{best}$

- 1 Generate an initial random solution  $X_{init}$ .
- 2 **for**  $iter_1 = 1 : Maxiter_1$
- 3 **for**  $iter_2 = 1 : Maxiter_2$
- 4 Initialize probability  $P_d$  for each destroy operator  $d \in D$  and probability  $P_r$  for each repair operator  $r \in R$
- 5 Let  $T$  be the temperature
- 6 Let  $X_{current} \leftarrow X_{best} \leftarrow X_{init}$
- 7 **repeat**
- 8 Select a removal operator  $d^* \in D$  with its probability  $P_d$  using roulette wheel selection, update  $Nu$ . Customers who have been removed from the solution string will be placed in a tabu list preventing them from being removed in the next iteration (tabu list).
- 9 Select an insertion operator  $r^* \in R$  with probability  $P_r$  using roulette wheel selection and then update  $Nu$ .
- 10 Let  $X_{new}^*$  be the solution obtained by applying operators  $d^*$  and  $r^*$  to  $X_{current}$  (SA algorithm: Steps 11-18)
- 11 **if**  $c(x_{new}) < c(x_{current})$  **then**
- 13  $X_{current} \leftarrow X_{new}$
- 14 **else**
- 15 Let  $\gamma \leftarrow e^{(-(c(X_{new}) - c(X_{current}))/T)}$
- 16 Generate the random number  $\omega \in [0, 1]$
- 17 **if**  $\omega < \gamma$  **then**
- 18  $X_{current} \leftarrow X_{new}$
- 19 Update  $T \leftarrow hT$
- 20 **if**  $iter_2 = Maxiter_2$
- 21 Update probabilities using the adaptive weight adjustment procedure
- 22 **until** the maximum number of iteration is reached.

---

**Figure 4.** IALNS pseudo-code.

#### 4.3.2. Assigning weights to destroy and repair operators

At the beginning, all weights are equal and the weight of every operator is shown by  $w_j$ . Then, operator  $j$  will be accepted as destroy operator or repair operator based on probabilities expressed in Equations (30) and (31), respectively:

$$P_{rj} = \frac{w_j}{\sum_{j \in r} w_j}, \quad (30)$$

$$P_{dj} = \frac{w_j}{\sum_{j \in d} w_j}. \quad (31)$$

At each iteration, destroy and repair operators will be chosen by the Roulette wheel selection method [28].

#### 4.3.3. Destroying the solution

In the destroy procedure, a certain percentage of customers will be removed from the solution string. The number of removed customers is defined by  $S$ , and the set of destroy operators is represented by  $D$ . The suggested ALNS includes 9 destroy operators of which 7 are adapted from Demir et al. [24] and the last two operators are proposed by the current study.

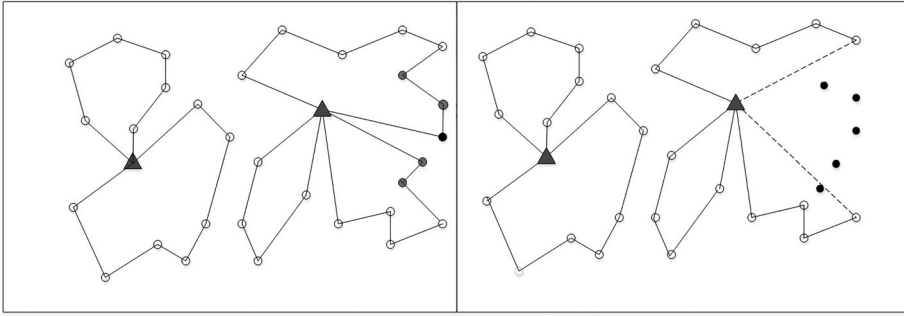
- (1) Random Removal (RR) operator: In this operator,  $S$  customers will be randomly removed from the solution string.
- (2) Worst-Distance Removal (WDR) operator: In this operator, first the position cost of every customer will be calculated (this cost is defined as the sum of distances of the customer from previous and next customer in the tour), then  $S$  customers with maximum position cost will be removed from the solution string.
- (3) Shaw Removal (SR) operator: In this operator, first, a customer will be randomly removed from the solution string; then the score of each customer with respect to this removal will be obtained by Equation (32). In this equation, customers who are in the same tour and are closer to the selected customer will have a higher chance of being selected. Then,  $S - 1$  customers with the lowest scores will be removed from the solution string.

In this equation,  $\phi_1$  and  $\phi_2$  are criteria weights,  $l_{ij}$  equals  $(-1)$  if customer  $i$  and customer  $j$  are in the same tour, and is otherwise zero. After evaluating the results, values of  $\phi_1$  and  $\phi_2$  are assumed to be  $\phi_1 = 0.57$ ,  $\phi_2 = 0.43$ .

$$aw_j = \arg \min \{ \phi_1 Dis_{ij} + \phi_2 l_{ij} \}. \quad (32)$$

- (4) Proximity-based Removal (PR) operator: This operator is a variant of SR operator. In this operator, first, a customer will be randomly removed from the solution string, and then  $S - 1$  customers who are closest to the removed





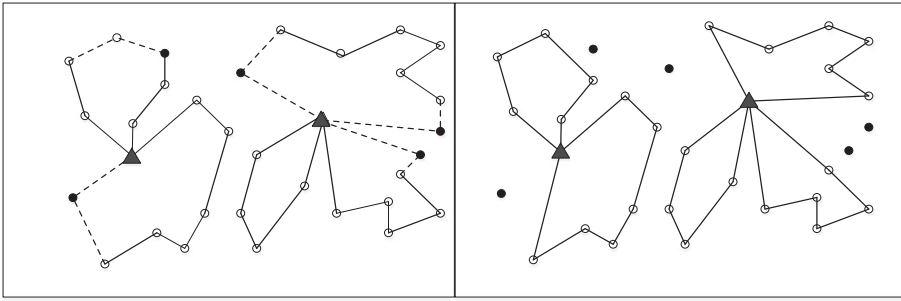
**Figure 5.** An example of the PR operator.

customer will be removed. An example of PR operator is illustrated in Figure 5.

- (5) **Historical Knowledge Node Removal (HR) operator:** In this operator, the position cost of every customer at each iteration is calculated. Then  $S$  customers whose current position cost has the greatest difference with their best position cost during the past iterations will be removed from the set of customers.
- (6) **Zone Removal (ZR) operator:** In this operator, first the entire area will be divided into smaller zones. Then one zone is randomly chosen and all points in that zone will be removed (see Equation (33)). This process will continue until the number of removed points reaches  $S$ .

$$s = \{j^* | x(i_1) \leq x(j^*) \leq x(i_2), y(i_1) \leq y(j^*) \leq y(i_2)\}. \quad (33)$$

- (7) **Node Neighbourhood Removal (NNR) operator:** this operator first selects a random point, and then creates a rectangular boundary around the point and removes all points inside that rectangle. In case that the number of removed points is less than  $S$ , the operator enlarges the boundary of the created rectangle until the number of removed points reaches  $S$ .
- (8) **Tour-based Removal (TR) operator:** this operator is a variant of SR operator, and its only difference is that the weights have changed to  $\phi_1 = 0, \phi_2 = 1$ . In this operator, there is a high probability that all customers of a tour get removed.
- (9) **Zone Worst Distance Removal (ZWDR) operator:** This operator is a combination of ZR and WDR operators. In this operator, the entire area considered for customer allocation will be divided into smaller zones. Then  $S$  zones will be selected randomly, and in each zone, the customer with the highest position cost will be removed. An example of Zone Worst Distance Removal (ZWDR) operator is given in Figure 6.



**Figure 6.** An example of the ZWDR operator.

#### 4.3.4. Updating the tabu list

In this study, the suggested algorithm is improved by a tabu list; meaning that customers who have been removed from the solution string will be placed in a tabu list preventing them from being removed in the next iteration.

#### 4.3.5. Repairing the solution

In this stage, one of the repair operators is chosen according to the computed probabilities to repair the solution destroyed in the previous step. The repair operators are as follows:

- (1) Greedy Insertion (GI) operator: This operator places the removed points in the best possible position of the tour. Accordingly, all possible edges will be evaluated and the one with the lowest cost ( $f_{i1}$ ) will be selected.
- (2) Regret insertion (RI) operator: one of the problems of GI is that it postpones the insertion of some points, and thus may ultimately lead to an unfeasible solution. For solving this problem, RI operator uses 2-regret criteria. This operator first computes the insertion cost of points like GI does, but then searches for the best and the second best insertion points  $f_{i1}$  and  $f_{i2}$ , and inserts customer with a maximum difference between two values  $\text{regret} = f_{i2} - f_{i1}$ .
- (3) Greedy Insertion with Noise function (GIN) operator: This operator is an improved version of GI operator, and its difference is that in this operator, the degree of freedom is used to select the best point of insertion. In this operator, the cost of inserting a customer to an edge will be summed with  $\bar{d}\mu\varepsilon$ . In this formula,  $\bar{d}$  is the maximum distance between points,  $\varepsilon \in [0, 1]$ , and  $\mu$  is a parameter for diversifying the solution ( $\mu = 0.1$ ).
- (4) Regret Insertion with Noise function (RIN) operator: This operator is also an improved form of GI operator. The difference is that in this operator, the degree of freedom is used to select the place of insertion.

#### 4.3.6. Acceptance procedure

In this stage, SA algorithm is used as an acceptance criterion. For more information, see Kuo [4]. This means that temporary solution  $x_{new}$  will always be accepted if  $c(x_{new}) < c(x_{current})$  and is accepted with a probability of  $e^{-(c(x_{new}) - c(x_{current}))/T}$  if  $c(x_{new}) > c(x_{current})$ . Moreover,  $T$  is set equal to the objective function value for the initial solution and decreases by a rate of  $h = 0.999$ .

#### 4.3.7. Updating the weights of operators

In each iteration of the algorithm, destroy and repair operators are given a score. When the operator manages to improve the best solution or the incumbent solution its score increases by  $\sigma_1, \sigma_2$ , respectively, and when it leads to a weaker solution, its score increases by  $\sigma_3$ . Equation (34) will be used for updating the weights of operators. Here, these scores are considered to be:

$$\begin{aligned}\sigma_1 &= 0.5, \sigma_2 = 0.25, \sigma_3 = 0.25, \\ w_j^{new} &= (1 - \lambda) \times w_j^{current} + \lambda \times \pi_j / Nu_j,\end{aligned}\tag{34}$$

where  $Nu_j$  stands for the number of times operator  $j$  has been used,  $\pi_j$  is the score of operator  $j$ ,  $w_j^{current}$  is the current weight of operator  $j$  and  $w_j^{new}$  is the new weight of this operator. Moreover,  $\lambda$  is a parameter that controls the staticity of the weights. For example, when  $\lambda$  is close to 0, the weights will be further affected by their previous values and when  $\lambda$  is close to 1, the weights will be updated based on the latest scores. Here, this parameter is considered to be  $\lambda = 0.13$ .

#### 4.3.8. Stopping condition

The proposed algorithm terminates after a certain number of iterations.

### 4.4. Variable neighbourhood search algorithm

Variable neighbourhood search algorithm (VNS) is one of the popular heuristic algorithms employed for dealing with optimization problems. The main idea of this algorithm is to change a neighbourhood by local search (LS). Variable neighbourhood search algorithm (VNS) was first introduced by Hansen and Mladenović in 1997 [29]. VNS algorithm consists of two main steps: shaking and local search. Suppose that  $N_l$  where  $l = 1, 2, \dots, l_{\max}$  is a predetermined neighbourhood structure and  $N_l(x)$  is set of neighbourhood  $x$  under structure  $N_l$ . In the shaking step, the algorithm randomly moves from current solution to a neighbourhood solution ( $S'$ ) by using  $i$ -th neighbourhood structure and performs a local search on solution ( $S'$ ) (it considers  $S'$  as the initial solution of the neighbourhood) to obtain a locally optimal solution  $S'^*$ . If  $S'^*$  is better than current solution  $S$ , the algorithm uses it as the new incumbent solution and resets the neighbourhood structure; otherwise, it moves to the next neighbourhood ( $N_{l+1}$ ).

---

**Algorithm 2:** variable neighborhood search

---

```

1 Input: a set of neighborhood structures  $N_l$ ,  $l = 1, 2, \dots, l_{max}$ ,
2  $S = \text{generate initial solution } ()$ ;
3 repeat
4    $l = 1$ ;
5    $Shake_{iter} = 0$ ;
6   while ( $Iteration \leq Max_{iter}$ )
7      $s' = \text{Shaking } (s, N_l)$ ;
8      $s'^* = \text{Local search } (s')$ ;
9     if  $c(s'^*) < c(s)$ 
10       $s \leftarrow s'^*$ 
11       $l = 1$ ;
12       $Shake_{iter} = 0$ ;
13   else
14     if  $Shake_{iter} = MaxShake_{iter}$ 
15       if  $l = l_{max}$ 
16          $l = 1$ ;
17          $Shake_{iter} = 0$ ;
18       else
19          $l = l + 1$ ;
20          $Shake_{iter} = 0$ ;
21   until stopping condition are met;
22 Output: The best solution;

```

---

**Figure 7.** Pseudo-code of the VNS algorithm.**4.5. VNS algorithm**

VNS algorithm consists of four main steps: generating an initial solution, shaking phase, local search and termination. In the following, all these steps are explained. Figure 7 represents the pseudo-code of the VNS.

**4.5.1. Generating an initial solution**

The procedure of generating an initial solution was explained in Subsection 4.1.

**4.5.2. Shaking step**

The aim of this phase is to create a significant change in the current solution [30]. In this phase, 5 structures are used to create a neighbourhood. The neighbourhood structures are ordered based on the extent of their impacts on the solution from simple neighbourhood structure to complex neighbourhood structure, and

they will be employed in that order. This means that the most simple neighbourhood structures will be used first and the most complex ones will be finally employed. It is noteworthy to highlight that only 5 selected neighbourhood structures are obtained by evaluating 12 initial neighbourhood structures. The neighbourhood structure will be changed when solutions remain unimproved after 5 successive iterations. This number is determined by the Taguchi design method [31].

- (1) *One-point transfer*: one customer will be selected in a tour and will be transferred to another tour.
- (2) *Two-point transfer*: two customers will be selected in a tour and will be transferred to another tour.
- (3) *One-point swap*: one customer in a tour will be randomly swapped with another customer in another tour.
- (4) *Two-point swap*: two customers in a tour will be randomly swapped with two customers in another tour.
- (5) *Tour removal*: the smallest tour in terms of allocated demand will be selected and all its customers will be transferred to other tours. In this transfer, each customer will be transferred to the tour for which this transfer leads to the lowest cost.

#### 4.5.3. Local search

The goal of this step is to seek local optimums. After forming the neighbourhood, a local search will be performed on the changed routes. Here, 2-opt and swap local search algorithms are utilized for this purpose. At the beginning of this step, one of the algorithms will be randomly selected to be applied to the solution. The policy for selecting solutions in local search methods is to select the best improvement.

- Swap algorithm

In the swap algorithm, two points selected from one tour will be swapped. To do this, the algorithm evaluates all possible changes in the selected tour and selects the one making the best change in the objective function. The algorithm will continue this process until reaching an improvement in the selected tour.

- 2-opt algorithm

This algorithm chooses two points  $i$  and  $j$  from the tour  $\{1, \dots, i, i+1, \dots, j, j+1, \dots, n\}$  and changes them to  $\{1, \dots, i, j, j-1, \dots, i+1, j+1, \dots, n\}$ . It then evaluates all possible values for  $i$  and  $j$  (where  $i \in \{1, 2, \dots, n-1\}$  and  $j \in \{i+1, \dots, n\}$ ) and chooses the best change for the objective function. This process will continue until an improvement is achieved in the selected tour.

**Table 4.** Values of adjusted parameters of algorithms.

VNS		ALNS and IALNS	
Parameter	Value	Parameter	Value
Total number of iterations	120	Number of iterations of main loop	80
Number of iterations without improvement in each shake	5	Number of iterations of internal loop	10
–	–	Number of removed customers	0.1 $N$

#### 4.5.4. Stopping condition

When the algorithm fails to find a better solution in a certain neighbourhood structure in a pre-specified number of iterations ( $Shake_{iter}$ ), it increases the neighbourhood index by 1 (which means that it moves to the next neighbourhood). When the algorithm finishes searching the last neighbourhood  $l_{max}$ , it returns to the first neighbourhood  $l_1$ . The algorithm will stop after a pre-specified number of iterations ( $Max_{iter}$ ).

#### 4.6. Parameter adjustment

Parameter adjustment of heuristic algorithms has a major impact on their performance. The Taguchi design method as an efficient design of experiments tool was introduced in 1960 [31]. It can determine the optimal conditions with the lowest number of experiments. In comparison with the classical factorial method, the application of this method can lead to a remarkable decrease in the time and the cost of experiments required for parameter adjustment. In this study, the Taguchi design method is utilized to adjust the parameters of the proposed algorithms, and in the next step, Minitab software is employed to do the relevant data analysis.

Here, a standard table was developed based on the number of factors and levels selected for analysis, in which 3 problem instances were randomly selected and solved 10 times using the values in the Taguchi table. The average values obtained for the objective function were imported into Minitab software. For each parameter, the level with maximum S/N value was considered as the best level. The final results are given in Table 4.

### 5. Numerical results

This section evaluates the performance of the proposed algorithms and provides an evaluation of cost-saving of the proposed model. The problem instances are generated by making some changes in multi-depot VRPs to make them more conforming to the assumptions of the studied problem. These problem instances have 3 types of vehicles with capacities of 26, 5 and 17 tons, respectively. Moreover, customers' demands were multiplied by 50. The number of vehicles of each

type was selected in a way that all customers can be completely serviced by one type of vehicle. The service time was assumed to be 0.48 of demand size. The small-scale problem instances are generated by selecting the first 9 problem instances introduced by Cordeau,<sup>1</sup> reducing the number of their customers, and making some changes in vehicle capacity and demands. For the large-scale problems, only 23 out of 33 problem instances are employed since the type of vehicles needed for each tour is determined by the proposed model.

The suggested algorithms were coded using MATLAB 2013 software and were executed on a computer with Core i5 CPU, 4GB of RAM and Windows 7 64bit Operating system. Furthermore, each algorithm is run 10 times and the best obtained solution is reported in each problem instance. On the other hand, CPLEX solver/GAMS software was employed as the exact solution method to deal with small-sized instances.

### 5.1. Results on small scale problems

The results obtained from solving small-scale problems are shown in Table 5. In the first column of this table, the first digit represents the instance number, second digit denotes the number of customers and the third digit shows the number of depots. The deviation between the objective function values obtained by different algorithms and the minimum value (gap%) is calculated by Equation (35). In this equation, *Objective* denotes the solution (objective function value) obtained from the algorithm and *BO* denotes the objective function value of the best solution.

$$gap\% = \frac{Objective - BO}{BO} \times 100. \quad (35)$$

According to the results shown in Table 5, the average errors of VNS, ALNS and IALNS algorithms are 0.30%, 0.26% and 0.18%, respectively. This demonstrates the good performance of these three algorithms in solving small-scale problems. The average run time of the exact method for small-scale problems is about 1400 s but the run times of VNS, ALNS and IALNS algorithms are 25.28, 1.93 and 2.01 s, respectively, which reflect the good speed of ALNS and IALNS algorithms.

**Table 5.** Computational results for small-sized problems instances.

Pro.	CPLEX solver		VNS			ALNS			IALNS		
	Total cost (\$)	Time (s)	Total cost (\$)	Time (s)	%gap	Total cost (\$)	Time (s)	%gap	Total cost (\$)	Time (s)	%gap
1-4-1	81.96	150.15	82.06	6.02	0.12	82.07	1.15	0.13	82.04	1.50	0.10
2-5-1	93.24	241.22	93.28	9.50	0.69	93.3	1.53	0.06	93.25	1.55	0.01
3-6-2	139.51	438.08	139.51	10.13	0.00	139.51	1.86	0.00	139.54	1.65	0.02
5-7-2	161.43	594.64	161.46	14.20	0.27	161.46	2.17	0.37	161.45	1.90	0.01
5-8-2	173.94	937.01	173.94	18.72	0.05	173.97	1.34	0.68	174.01	1.85	0.04
6-9-2	156.81	1077.19	157.15	19.23	0.22	157.11	2.45	0.19	156.81	2.35	0.14
7-10-2	197.44	1951.03	197.46	32.10	0.23	197.48	1.70	0.02	198.64	2.15	0.61
8-11-2	220.32	3149.67	222.13	50.35	0.82	220.75	2.23	0.20	220.34	2.46	0.11
9-12-3	256.43	4037.88	257.19	67.49	0.30	258.13	2.95	0.66	257.91	2.70	0.58
<b>Average</b>	164.56	1397.43	165.08	25.28	0.30	165.06	1.93	0.26	164.94	2.01	0.18

**Table 6.** Computational results for large-sized problems instances.

Pro.	VNS			ALNS			IALNS		
	Total cost (\$)	Time (s)	%gap	Total cost (\$)	Time (s)	%gap	Total cost (\$)	Time (s)	%gap
1-50-2	518.95	558.44	0.00	529.25	3.05	1.98	521.59	5.74	0.51
3-75-2	664.52	861.67	0.69	670.20	7.61	1.55	659.98	7.97	0.00
5-100-2	906.43	975.22	0.12	931.69	14.35	2.92	905.30	15.94	0.00
6-100-3	858.36	998.95	0.38	885.88	14.83	3.60	855.11	19.67	0.00
7-100-4	891.71	1014.65	1.40	907.98	15.94	3.25	879.43	22.09	0.00
8-249-2	4689.61	2737.25	4.93	4704.18	371.64	5.26	4469.28	412.68	0.00
9-249-3	4432.22	2669.17	4.12	4480.11	349.06	5.24	4256.85	394.67	0.00
10-249-4	4309.38	2708.95	5.04	4285.30	364.02	4.45	4102.55	425.07	0.00
11-249-5	4458.80	2509.40	5.67	4413.78	341.98	4.61	4219.37	445.91	0.00
12-80-2	685.34	793.51	0.00	708.00	4.58	3.31	689.12	5.26	0.55
15-160-2	1289.43	1249.75	3.98	1240.08	31.09	0.00	1253.93	36.44	1.12
18-240-4	2212.19	1729.40	2.95	2152.67	78.64	0.18	2148.90	99.09	0.00
21-360-6	3281.76	2597.43	4.32	3301.00	261.09	4.93	3145.81	297.66	0.00
1-48-2	654.66	509.43	1.40	647.72	2.74	0.32	645.65	3.45	0.00
2-96-3	1061.66	905.67	0.00	1109.00	12.46	4.46	1067.40	12.09	0.54
3-144-4	1579.70	1027.04	3.68	1573.07	26.50	3.25	1523.60	31.63	0.00
4-192-3	1992.93	1652.72	2.09	1970.90	99.10	0.96	1952.20	115.03	0.00
5-240-3	2309.07	1949.49	3.87	2223.05	126.06	0.00	2239.38	168.74	0.73
6-288-4	2875.44	2031.18	3.57	2860.50	158.79	3.04	2776.20	193.19	0.00
7-72-3	845.91	829.64	1.37	839.60	7.64	0.61	834.50	7.05	0.00
8-144-3	1438.25	1109.06	2.53	1419.00	31.08	1.16	1402.73	37.50	0.00
9-216-5	1988.76	1872.22	3.12	1978.43	83.94	2.58	1928.65	99.62	0.00
10-288-6	2769.37	2145.19	2.69	2746.20	281.97	1.83	2696.72	214.45	0.00
Average	2031.06	1540.67	2.52	2025.11	116.88	2.59	1964.10	133.52	0.15

## 5.2. Results on large scale problems

In Table 6, the results obtained from solving the large-scale problem instances with three proposed algorithms are presented.

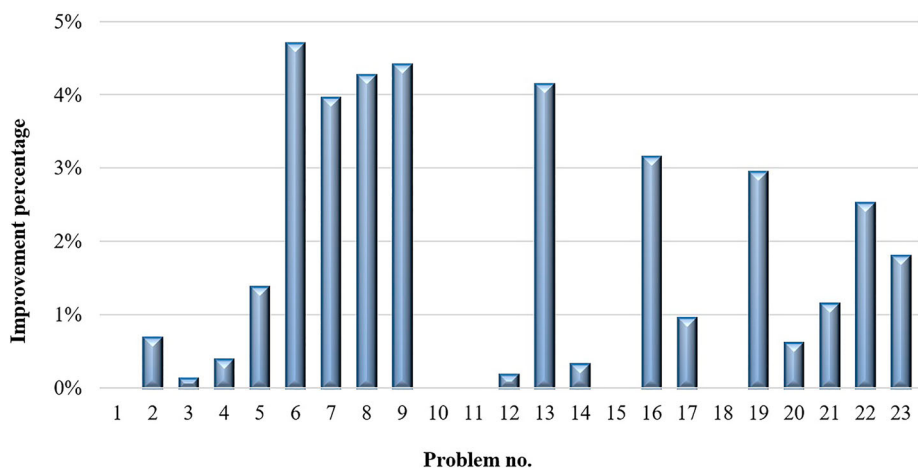
The errors of VNS, ALNS and IALNS algorithms when solving large-scale problems are 2.52%, 2.59% and 0.15%, respectively, and their average run times are 1540.67, 116.88 and 133.52 s, respectively. In terms of solution quality, the IALNS algorithm obtained the best solutions for 18 out of 23 assessed instances, while VNS and ALNS algorithms did so in only 3 and 2 problem instances, respectively. Figure 8 shows the improvement percentage of the solutions obtained by the IALNS algorithm in comparison with the best solutions achieved by VNS and ALNS algorithms.

The maximum improvement by the proposed algorithm is about 4.7%, and this algorithm was able to make the average improvement of 1.63% in the solutions. These results demonstrate the appropriate performance of the proposed IALNS algorithm.

## 5.3. Statistical analysis

In order to have a confident comparison of the algorithms in large-scale problems, it is necessary to conduct a comparative assessment using the statistical test between the proposed IALNS algorithm and the other algorithms. Due to this, the Friedman rank test is utilized under a 95% confidence level. For more information, please see Costa and Fernandes [32]. SPSS software is used to implement





**Figure 8.** Improvement of solutions in IALNS algorithm.

the test and the obtained results are given by Table 7. In this table,  $N$ ,  $df$ , Asymp. Sig. and Chi-Square stand for the number of samples, degree of freedom, significant level and value of Chi-Square statistic, respectively. The algorithm with the lowest mean rank is known as the best algorithm. Hence, IALNS is selected as the best algorithm to solve the problem.

**5.4. Managerial insights: impacts of the proposed model on cost-saving**

In this subsection, we first investigate the effect of using a heterogeneous fleet on reducing the distribution costs and then study the effect of fuel-saving objective function on the quality of solutions. To investigate the effect of using a heterogeneous fleet, first, a homogenous fleet was assumed and all problem instances were solved accordingly by the IALNS method. In this step, the capacity of all vehicles was assumed to be 5 tons (medium size vehicle), and the results were compared with the solution obtained with a heterogeneous fleet. The results of this comparison are illustrated in Table 8. In this table, the first column represents the problem specification, the second column shows the results obtained from the problem with heterogeneous fleet, and the third column presents the results obtained from the problem with a homogenous fleet of vehicles. To examine the effect of fuel-saving (greenness) objective function on the quality of solution, the

**Table 7.** Friedman rank test report.

<i>(N = 23, df = 2, Asymp. Sig. = .000, Chi-Square = 21.217)</i>		
Algorithm	Mean rank (Friedman rank)	Final rank
VNS	2.43	3
ALNS	2.35	2
IALNS	1.255	1

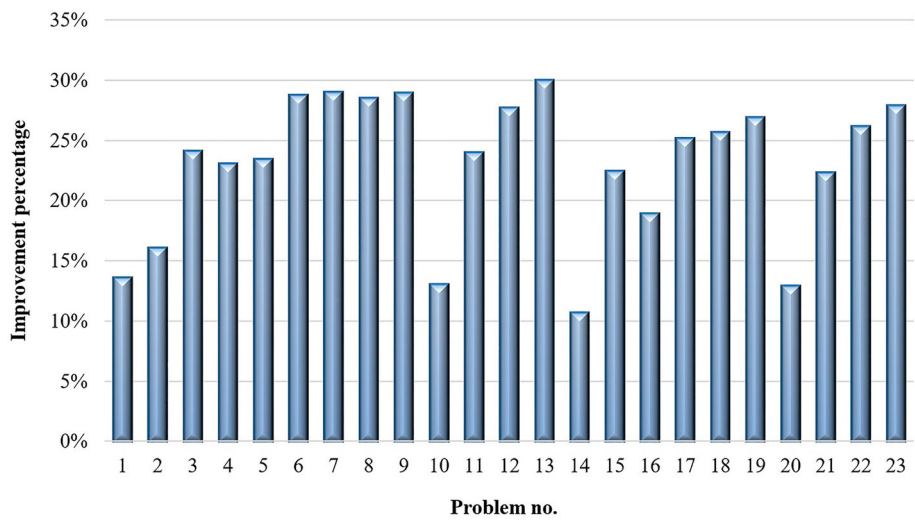
**Table 8.** Effects of the heterogeneous fleet and fuel-saving expressions on the quality of solutions.

Pro.	Heterogeneous	Homogenous		Heterogeneous and Green Problem	
	Total cost (\$)	Total cost (\$)	%gap	Total cost (\$)	%gap
1-50-2	521.59	603.74	15.75	553.79	6.17
3-75-2	659.91	786.14	18.67	693.88	5.15
5-100-2	900.9	1186.77	31.73	965.18	7.14
6-100-3	853.21	1109.27	30.01	897.33	5.17
7-100-4	879.43	1148.21	30.56	937.14	6.56
8-249-2	4469.28	6271.98	40.34	5301.67	18.26
9-249-3	4256	5993.51	40.82	5198.43	22.14
10-249-4	4100.86	5738.14	39.93	4899.71	19.48
11-249-5	4219.37	5937.38	40.72	4937.18	17.01
12-80-2	689.12	792.38	14.98	722.91	4.9
15-160-2	1253.9	1649.11	35.11	1381.22	10.15
18-240-4	2148.9	2971.35	38.27	2491.7	15.95
21-360-6	3145.81	4493.17	42.83	3735.89	18.76
1-48-2	645.65	723.09	11.98	663.79	2.81
2-96-3	1067.4	1376.22	28.93	1139.71	6.77
3-144-4	1523.6	1879.2	23.34	1679.12	10.21
4-192-3	1952.2	2607.88	33.59	2259.87	15.76
5-240-3	2237.3	3009.78	34.53	2608.17	16.58
6-288-4	2776.2	3796.55	36.75	3469.77	24.98
7-72-3	834.5	958.4	14.85	871.69	4.46
8-144-3	1397	1798.22	28.72	1507.8	7.93
9-216-5	1927.3	2609.81	35.41	2291.16	18.88
10-288-6	2659.7	3687.12	38.63	3207.93	20.61
Average	1961.7	2657.71	30.72	2278.91	12.43

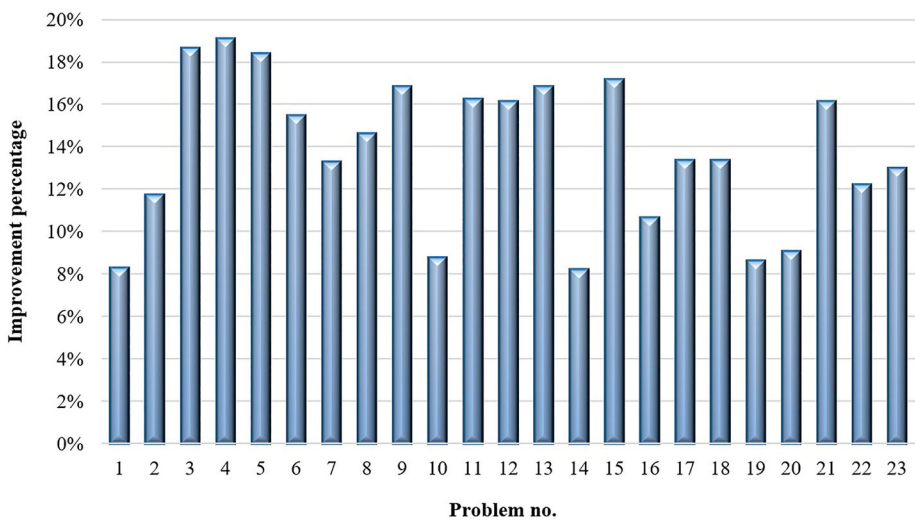
fuel-saving expressions of the proposed model were replaced with travel time reduction terms, and problem instances were solved again. The results are given in the last column of Table 8.

According to the obtained results, using a heterogeneous fleet of vehicles in the proposed problem provides a significant cost-saving where there is 30.72% increase in the total cost of the problem with homogeneous problem. Furthermore, incorporating fuel-saving expressions in the problem with heterogeneous leads to the average increase of 12.43% in the total cost compared to the sole heterogeneous problem. Figure 9 shows the effects of using a heterogeneous fleet on the quality of solutions against homogenous fleet in the non-green problem. Furthermore, Figure 10 depicts the effects of using heterogeneous fleet in the green problem against the non-green problem with homogenous fleet.

According to Figure 9, the maximum and minimum improvement made in the solutions as the result of using a heterogeneous fleet are observed in Instances 21 and 1 exhibiting 29.99% and 10.71% quality improvement, respectively. Moreover, the average improvement is 23.01%. As Figure 10 illustrates, the maximum improvement made in the solutions as the result of using fuel-saving expressions is observed in Instance 6 exhibiting a 19.11% improvement against the non-green problem with homogeneous fleet. Furthermore, the average improvement is 13.74%. The obtained results demonstrate the desirable effects of employed assumptions on the quality of obtained solutions.



**Figure 9.** Improvements made as the result of using a heterogeneous fleet against homogenous fleet in the non-green problem.



**Figure 10.** Improvements made as the result of using a heterogeneous fleet in the green problem against homogenous fleet in the non-green problem.

## 6. Conclusions

This research developed a novel MILP model to formulate the multi-depot TD-FSMGVRP. The objective function of the proposed model was to minimize the total cost including the vehicles' fixed cost, drivers' cost, fuel costs, and GHG emission costs. This model was developed based on the presence of a heterogeneous fleet and multiple depots, and also incorporated the FIFO feature. Given the NP-hard complexity of the proposed problem, an improved meta-heuristic

algorithm; i.e. IALNS meta-heuristic algorithm was designed to tackle the complexity of the problem, especially in large-scale problem instances. The Taguchi design method was also employed to tune the parameters of the algorithm and enhance its efficiency. To test the performance of the IALNS, two rival algorithms of VNS and ALNS were taken into account. The solutions of VNS, ALNS and IALNS algorithms for small-scale problem instances exhibited 0.30%, 0.26% and 0.18% error in comparison with the exact solutions obtained by CPLEX solver/GAMS software. For large-scale problem instances, the error rate of VNS, ALNS and IALNS algorithms was 2.52%, 2.59% and 0.15%, respectively. Furthermore, the Friedman test was utilized in order to demonstrate the superiority of the proposed IALNS algorithm. Finally, the effects of using a heterogeneous fleet and incorporating fuel-saving expressions in the model were investigated. It was revealed that using a heterogeneous fleet for the proposed model leads to a remarkable cost-saving and makes an average improvement of 23.01%. On the other hand, incorporating fuel-saving expressions in the model leads to an average cost-saving of 13.74% in the green problem with heterogeneous fleet against the non-green problem with homogenous fleet.

For future research directions, several suggestions can be considered based on the main limitations of the study. Other state-of-the-art meta-heuristic algorithms; e.g. Grey Wolf Optimizer (GWO) [33], can be utilized and compared to the proposed IALNS. Moreover, the uncertainty of the parameters can be studied using efficient techniques such as fuzzy programming, robust optimization, stochastic optimal control and grey systems [33–39].

## Note

1. <https://neo.lcc.uma.es/vrp/vrp-instances/multiple-depot-vrp-instances/>.

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## ORCID

Erfan Babaee Tirkolaee  <http://orcid.org/0000-0003-1664-9210>

## References

- [1] Sbihi A, Eglese R. Combinatorial optimization and Green logistics. *4OR*. 2007;5:99–116.
- [2] Bodin L, Golden B. Classification in vehicle routing and scheduling. *Networks*. 1981;11:97–108.
- [3] Demir E, Bektaş T, Laporte G. A comparative analysis of several vehicle emission models for road freight transportation. *Trans Res Part D: Trans Environ*. 2011;16:347–357.
- [4] Kuo Y. Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem. *Comput Ind Eng*. 2010;59:157–165.

- [5] Demir E, Bektaş T, Laporte G. A review of recent research on Green road freight transportation. *Eur J Oper Res*. 2014;237:775–793.
- [6] Dantzig G, Fulkerson R, Johnson S. Solution of a large-scale traveling-salesman problem. *J Oper Res Soc Am*. 1954;2:393–410.
- [7] Clarke G, Wright JW. Scheduling of vehicles from a central depot to a number of delivery points. *Oper Res*. 1964;12:568–581.
- [8] Malandraki C, Daskin MS. Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. *Transp Sci*. 1992;26:185–200.
- [9] Hill AV, Benton WC. Modelling intra-city time-dependent travel speeds for vehicle scheduling problems. *J Oper Res Soc*. 1992;43:343–335.
- [10] Ichoua S, Gendreau M, Potvin J-Y. Vehicle dispatching with time-dependent travel times. *Eur J Oper Res*. 2003;144:379–396.
- [11] Haghani A, Jung S. A dynamic vehicle routing problem with time-dependent travel times. *Comput Oper Res*. 2005;32:2959–2986.
- [12] Chen H-K, Hsueh C-F, Chang M-S. The real-time time-dependent vehicle routing problem. *Trans Res Part E: Logis Trans Rev*. 2006;42:383–408.
- [13] Balseiro SR, Loiseau I, Ramonet J. An ant colony algorithm hybridized with insertion heuristics for the time dependent vehicle routing problem with time windows. *Comput Oper Res*. 2011;38:954–966.
- [14] Jin T, Guo S, Wang F, et al. One-stage search for multi-depot vehicle routing problem. *Proc Inform Syst Cont Conf*. 2004; p.446–129.
- [15] Tsirimpas P, Tatarakis A, Minis I, et al. Single vehicle routing with a predefined customer sequence and multiple depot returns. *Eur J Oper Res*. 2008;187:483–495.
- [16] Liu R, Jiang Z, Fung RYK, et al. Two-phase heuristic algorithms for full truckloads multi-depot capacitated vehicle routing problem in carrier collaboration. *Comput Oper Res*. 2010;37:950–959.
- [17] Gulczynski D, Golden B, Wasil E. The multi-depot split delivery vehicle routing problem: an integer programming-based heuristic, new test problems, and computational results. *Comput Ind Eng*. 2011;61:794–804.
- [18] Salhi S, Imran A, Wassan NA. The multi-depot vehicle routing problem with heterogeneous vehicle fleet: formulation and a variable neighborhood search implementation. *Comput Oper Res*. 2014;52(Part B):315–325.
- [19] Xiao Y, Zhao Q, Kaku I, et al. Development of a fuel consumption optimization model for the capacitated vehicle routing problem. *Comput Oper Res*. 2012;39:1419–1431.
- [20] Maden W, Eglese R, Black D. Vehicle routing and scheduling with time-varying data: a case study. *J Oper Res Soc*. 2009;61:515–522.
- [21] Bektaş T, Laporte G. The pollution-routing problem. *Transp Res B, Methodol*. 2011;45:1232–1250.
- [22] Koç Ç, Bektaş T, Jabali O, et al. The fleet size and mix pollution-routing problem. *Transp Res B, Methodol*. 2014;70:239–254.
- [23] Karakostas P, Sifaleras A, Georgiadis MC. Adaptive variable neighborhood search solution methods for the fleet size and mix pollution location-inventory-routing problem. *Expert Syst Appl*. 2020;153:113444.
- [24] Demir E, Bektaş T, Laporte G. An adaptive large neighborhood search heuristic for the pollution-routing problem. *Eur J Oper Res*. 2012;223:346–359.
- [25] Barth M, Boriboonsomsin K. Real-world carbon dioxide impacts of traffic congestion. *Transp Res Rec: J Transp Res Board*. 2008;2058:163–171.
- [26] Chang C-T, Chang C-C. A linearization method for mixed 0–1 polynomial programs. *Comput Oper Res*. 2000;27:1005–1016.

- [27] Shaw P. Using constraint programming and local search methods to solve vehicle routing problems. Presented at the proceedings of the 4th international conference on principles and practice of constraint programming; 1998.
- [28] Tasan AS, Gen M. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Comput Indut Eng.* [2012](#);62:755–761.
- [29] Mladenović N, Hansen P. Variable neighborhood search. *Comput Oper Res.* [1997](#);24:1097–1100.
- [30] Fleszar K, Osman IH, Hindi KS. A variable neighbourhood search algorithm for the open vehicle routing problem. *Eur J Oper Res.* [2009](#);195:803–809.
- [31] Roy RK.. A primer on the Taguchi method. 2nd ed. New York: Van Nostrand Reinhold; [1990](#).
- [32] Costa MF, Fernandes EMGP. Assessing the potential of interior point barrier filter line search methods: nonmonotoneversusmonotone approach. *Optimization.* [2011](#);60:1251–1268.
- [33] Khalilpourazari S, Hashemi Doulabi H, Özyüksel Çiftçioğlu A, et al. Gradient-based grey wolf optimizer with Gaussian walk: application in modelling and prediction of the COVID-19 pandemic. *Expert Syst Appl.* [2021](#);177:114920.
- [34] Khalilpourazari S, Khalilpourazary S, Özyüksel Çiftçioğlu A, et al. Designing energy-efficient high-precision multi-pass turning processes via robust optimization and artificial intelligence. *J Intell Manuf.* [2020](#);32:1621–1647.
- [35] Khalilpourazari S, Hashemi Doulabi H. Designing a hybrid reinforcement learning based algorithm with application in prediction of the COVID-19 pandemic in Quebec. *Ann Oper Res.* [2021](#).doi:[10.1007/s10479-020-03871-7](#).
- [36] Khalilpourazari S, Pasandideh SH. Designing emergency flood evacuation plans using robust optimization and artificial intelligence. *J Comb Optim.* [2021](#);41:640–677.
- [37] Lotfi R, Mardani N, Weber GW. Robust Bi-level programming for renewable energy location. *Int J Energy Res.* [2021](#);45:7521–7534.
- [38] Lotfi R, Mehrjerdi YZ, Pishvae MS, et al. A robust optimization model for sustainable and resilient closed-loop supply chain network design considering conditional value at risk. *Num Algebra Cont Optim.* [2021](#);11:221.
- [39] Lotfi R, Yadegari Z, Hosseini SH, et al. A robust time-cost-quality-energy-environment trade-off with resource-constrained in project management: a case study for a bridge construction project. *J Indus Manage Optim.* [2021](#). doi:[10.3934/jimo.2020158](#).