Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Time-dependent vehicle routing problem with backhaul with FIFO assumption: Variable neighborhood search and mat-heuristic variable neighborhood search algorithms

Esmaeil Akhondi Bajegani[a*], Naser Mollaverdi[a] and Mahdi Alinaghian[a]

[a]*Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran*

C H R O N I C L E

Article history:
Received August 17 2020
Received in Revised Format October 7 2020
Accepted October 21 2020
Available online October, 21 2020

Keywords:
VRPB
Time-dependent vehicle routing
FIFO assumption
VNS algorithm
Mat-VNS algorithm

A B S T R A C T

This paper presents a mathematical model for a single depot, time-dependent vehicle routing problem with backhaul considering the first in first out (FIFO) assumption. As the nature of the problem is NP-hard, variable neighborhood search (VNS) meta-heuristic and mat-heuristic algorithms have been designed. For test problems with large scales, obtained results highlight the superior performance of the mat-heuristic algorithm compared with that of the other algorithm. Finally a case study at the post office of Khomeini-Shahr town, Iran, was considered. Study results show a reduction of roughly 19% (almost 45 min) in the travel time of the vehicle.

1. Introduction

The vehicle routing problem (VRP) is amongst the important problems in operational research and physical distribution. Generally, this problem is defined using a graph like $G = (V, E)$ where $V = \{0,1, \dots, n\}$ denotes the demand nodes or customers. The node $\{0\}$ represents the depot and E includes the edges of the transportation network, showing the routs between the nodes. The VRP is symmetric if the transportation cost from edge (i, j) is equal to C_{ij} and $C_{ij} = C_{ji}$; otherwise, the problem is asymmetric. Extensive research has been conducted so far for the classic VRP and its variants including capacitated vehicle routing problem (CVRP), VRP with heterogeneous fleet, VRP with time window (VRPTW), periodic VRP (PVRP), dynamic VRP (DVRP), VRP with multiple trips (VRPMT), and split delivery VRP (SDVRP). According to the definition of the VRP with Backhaul (VRPB) provided by (Toth & Vigo, 1997), set V, including all network nodes, is split into three subsets: $L = \{1, \dots, m\}$: the set of linehaul customers, $B = \{m + 1, \dots, n\}$: the set of backhaul customers, and $\{0\}$: the depot. Moreover, good delivery to the customer is assumed as the main profitable activity, meaning that a vehicle is utilized only if it services at least one first-type customer.

In simple terms, the VRPB is looking for determining an optimal program for vehicles, where each vehicle is loaded in the depot and after full delivery of the good to linehaul customers it received a good from the backhaul customer and returns it back to the central depot.

Traffic and traversing a specific route by vehicles with different speeds during a day is an undeniable fact in the real work, considering which can make the model more realistic. This feature has led to the emergence of the time-dependent VRP (TDVRP) in the literature. Considering this feature, the present paper models the TDVRP with backhaul (TDVRPB). In

* Corresponding author Tel.: +989137230679
E-mail: e.akhondi@gmail.com (E. A. Bajegani)

2021 Growing Science Ltd.
doi: 10.5267/j.ijiec.2020.10.003

addition, to approach the real-world solutions, the time-dependence constraint along with the FIFO assumption is applied to the model.

In the rest of the paper, Section 2 provides the literature review. Section 3 thoroughly presents the definition and mathematical modeling of the problem. Section 4 explains the proposed solution methods. Section 5 provides the experimental proofs of the proposed solution methods efficiency. Section 6 is allocated to numerical results. A case study is described in Section 7 and finally, section 8 provides the conclusions.

2. Literature review

To the best of the author's knowledge, this paper proposes the first model on TDVRPB and there has not been any research in this scope in the literature; therefore, in order to find the research gaps, the literature on VRPB and TDVRP as the areas that have the most similarity to TDVRPB will be reviewed.

2.1. VRPB

To the best of our knowledge, Toth and Vigo (1997) proposed an integer linear programming model for the VRPB for the first time. Then Mingozzi et al. (1999) presented such a model based on a set partitioning approach. (Osman & Wassan, 2002) developed a Tabu Search heuristic method for the problem. This was the first paper at that time that used a meta-heuristic algorithm for solving the VRPB. Meta-heuristic approaches gained the attention of researches in the next studies, for example, in references (Brandao, 2006; Brandão, 2016; Gajpal & Abad, 2009; Vidal et al., 2014). Salhi et al. (2013) proposed an Integer Linear Programming (ILP) formulation of the VRPB with heterogeneous fleet. They also proposed a set partitioning problem based heuristic and tested on a set of new test problems which they generated. (Arab et al., 2018) considered the combination of a VRPB and a multi-period multi-product inventory routing problem with a heterogeneous transportation fleet. In addition to the conventional objective of system costs, they considered the minimization of route risk as a separate objective and used two algorithms for solving the problem: Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Multi-Objective Imperialist Competitive Algorithm (MOICA). Granada-Echeverri et al. (2019) proposed a new mixed-integer linear programming (MILP) for the VPRB based on extending the open routing problem. This extension had made it possible to generate a viable solution using the new constraints and removing sub-tour.

2.2. TDVRP

For the first time, Malandraki (1989) modeled the time-dependent traveling salesman problem. Hill and Benton (1992) presented a model for time-dependent vehicle travel speeds. The main weakness of the research was dissatisfying the FIFO assumption. In other words, the vehicle that leaves node i toward node j earlier than the other vehicle is expected to reach the destination sooner than that. This feature was first noticed by Ichoua et al. (2003). They utilized the step function of the travel speed and as a result the linear-piecewise function of travel times to solve the TDVRP with soft time windows and employed meta-heuristic tabu search method to solve test problems of Solomon (1987) with three time periods. Ever since, the use of meta-heuristic algorithms in the TDVRP was embraced by scholars, for instance, (Haghani & Jung, 2005; Kritzinger et al., 2012; Zhang et al., 2014). (Setak et al., 2015) investigated the TDVRP in a multi-graph case, meaning that more than one edge is imaginable between every two nodes of the network. The authors even considered the FIFO assumption for the problem. They proposed a tabu search algorithm to solve the problem. Huang et al. (2017) studied the TDVRP with flexible routes (the same feature as the multi-graph in (Setak et al., 2015). The authors stated that when the objective is of cost type, and not time type, the FIFO assumption has no application in the given model. This has been mentioned in (Savelsbergh & Van Woensel, 2016) as well. It is clear that this is true if travel time has no effect on determining the value of the cost function. Hou et al. (2018) considered ride-mathching and routing optimization. They considered the ridesharing problem on how to match riders to drivers and how to choose the best routes for vehicles. The authors developed a Large Neighborhood Search (LNS) meta-heuristic to solve the problem. They did not consider traffic and FIFO assumption in their model. (Rincon-Garcia et al., 2018) regarding the rules concerning driving hours, considered the TDVRP with a hard time window without presenting a mathematical model and studied the minimization of the number of vehicles employed and the whole travel time. In the same paper, the authors presented a LNS meta-heuristic algorithm to solve the problem. Keskin et al. (2019) analyzed the electric vehicle routing problem assuming the minimization of the costs and taking into account time dependency of the charging time in the station to time windows. They also defined the FIFO assumption to service the vehicles returned back to the station and presented an adaptive large neighborhood search (ALNS) meta-heuristic algorithm. Soon et al. (2019) proposed a proactive Eco-friendly Pheromone-based Green Vehicle Routing (E-PGVR) strategy to prioritize the greener aspect of routing by reducing fuel consumption. They did not estimate traffic functions in network and as a result neglect FIFO assumption. Delgado-Antequera et al. (2020) considered four different objectives to model the waste collection problem: travel cost, rout length balance, route time balance and number of routs. They proposed an iterated greedy algorithm coupled with a VNS to determine a good approximation to the Pareto front. Unfortunately they did not consider traffic and FIFO assumption in their model.

A list of the outstanding literature is provided in Table 1.

Table 1
Review of the outstanding literature

Paper	Backhaul	Time-dependency	FIFO	Mathematical model	Solution method				Type of objective function	
					Exact	Heuristics	Meta-heuristic	Mat-heuristic	Time	Cost
(Toth & Vigo, 1997)	✓			✓	✓					✓
(Mingozzi et al., 1999)	✓			✓	✓	✓				✓
(Salhi et al., 2013)	✓			✓		✓				✓
(Osman & Wassan, 2002)	✓						✓			✓
(Brandao, 2006; Brandão, 2016; Gajpal & Abad, 2009; Vidal et al., 2014)	✓						✓			✓
(Arab et al., 2018)	✓			✓			✓			✓
(Granada-Echeverri et al., 2019)	✓			✓			✓			✓
(Ichoua et al., 2003)		✓	✓				✓		✓	
(Haghani & Jung, 2005)		✓	✓	✓			✓		✓	
(Kritzinger et al., 2012)		✓	✓				✓		✓	
(Zhang et al., 2014)		✓	✓	✓			✓			✓
(Setak et al., 2015)		✓	✓	✓			✓			✓
(Huang et al., 2017)		✓		✓		✓				✓
(Rincon-Garcia et al., 2018)		✓					✓		✓	✓
(Keskin et al., 2019)		✓	✓	✓				✓		✓
(Soon et al., 2019)		✓		✓		✓			✓	
(Delgado-Antequera et al., 2020)				✓			✓		✓	✓
This paper	✓	✓	✓	✓			✓	✓	✓	

3. Definition and mathematical modeling

This paper presents a mathematical model for a time-dependent routing problem with backhaul by applying the first in first out (FIFO) assumption. The objective of the problem is to minimize the distance traversed by vehicles. There is one depot in the problem from where the vehicles start their trip. At the start of the trip, vehicles deliver the products required by linehaul customers (type 1) and receive goods from backhaul customers (type 2) in the return route, and return them back to the depot. Homogenous transportation fleet and capacities are equal and limited. It is evident that the capacity limit for both the delivery route from the depot to service the linehaul customers and the pickup (return) route to the depot to service the backhaul customers is separately applied. The limitation on the distance traveled is also discarded for vehicles. Split delivery for the product demanded by the type 1 customers and split reception for goods received from the type 2 customers are not permitted, and each type 1 customer has to fully receive its required product from one vehicle. Similarly, each type 2 customer has to fully deliver its good to only one vehicle. Working hours of a day are divided into separate time periods as such that each of the periods is a specific function depending on traffic conditions of each edge, the time required to traverse the edge by vehicles, in terms of the departure time of the trip from the origin of that edge. The function is called the traffic function.

3.1. Problem assumptions

The following assumptions are taken into account in the proposed method:

- The goods required by linehaul customers are supplied by vehicles from a central depot.
- The received good by vehicles from backhaul customers, after servicing linehaul customers, is collected in the return route and is transformed to the central depot.
- Transportation fleet consists of homogenous vehicles with a specific capacity.
- The size of the transportation fleet is fixed and specific.
- The demands of all customers are deterministic.
- The traffic between any pair of nodes and between nodes and the central depot is definite using the travel time function in terms of day hours.
- The FIFO assumption is considered in the problem.
- The objective is to minimize the total travel time of vehicles.
- Each customer is visited exactly one time and by exactly one vehicle.

3.2. FIFO assumption

Normally, the traffic on the routes changes during the day. This means that, in a time-dependent routing problem, the travel time of a vehicle on an edge has to be considered dependent on the departure time of the trip from the origin of the edge. Malandraki and Daskin (1992) considered the travel time a known step function of the time of day (as Figure. 1). This idea brought about two main deficiencies:

1. It is likely that the travel time is reduced of the vehicle waits in the origin of the art until the traffic is light (Malandraki & Dial, 1996). For instance, in Figure. 1, if the vehicle starts its trip within the interval b + 1, instead of b, the travel time is reduced. However, this is very difficult in the real world it not impossible (Mugayskikh et al., 2018). Therefore, using such a function seems illogical.
2. When two vehicles leave a common node within respectively time intervals b and b + 1, if the travel time for interval b + 1 is shorter than that of interval b, the second vehicle will approach the destination sooner than the first vehicle although it has started its trip later than the first vehicle. This deficiency, in TDVRP problems, is called passing, which is contradictive in the real world. Refer to Fig. 1 (Alinaghian et al., 2017).

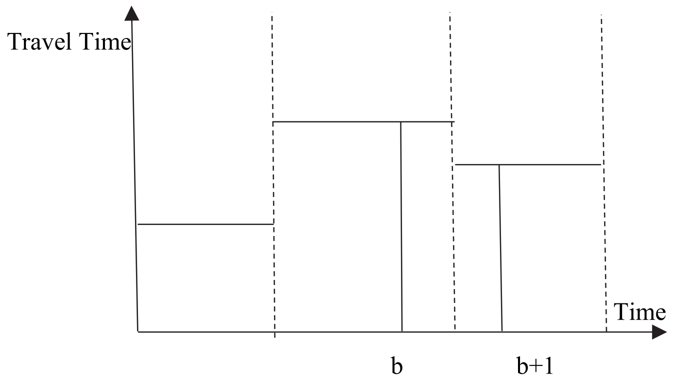

Fig. 1. Step function of travel time-time

Two abovementioned deficiencies, in fact, remind us of considering the FIFO assumption in TDVRPs, which was first introduced by (Ichoua et al., 2003). According to this assumption, at any time during a day, if two vehicles leave a common origin to a specific destination on a common route, the vehicle that has started its trip earlier will reach the destination earlier as well. The authors used a speed step function to this aim. This function showed abrupt changes in the vehicle speed at the end of time intervals, which was far away from reality (refer to Fig. 2).

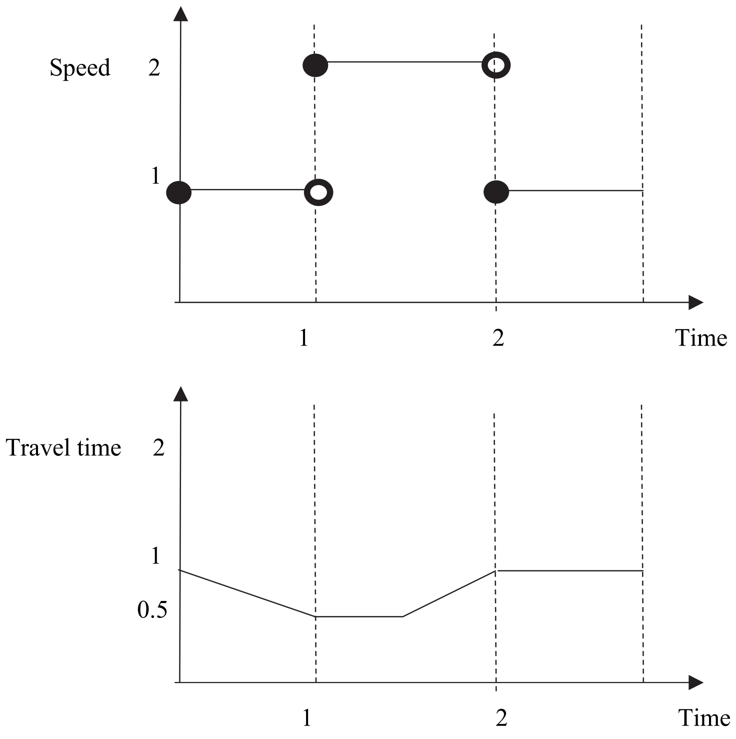

Fig. 2. The curve of speed-time and travel time-time

Mancini (2014) proposed using polynomial functions of travel time to solve the aforementioned problem. Then (Alinaghian et al., 2017), specifically, suggested using continuous functions of travel time. They highlighted the fact that the changes in the travel time and speed will occur continuously, and not abruptly, in this case. Additionally, if the linear slope of the function is smaller than -1, the non-passing rule, i.e. the FIFO assumption, will also be met. The current paper utilizes continuous linear functions with a slope greater than -1 for each time interval for travel time to apply traffic to each edge.

3.3. Mathematical modeling

Table 2 shows the notations used in the model.

Table 2
Notations used in the model.

Sets	
V	The set of all nodes of the network with indices i and j
L	The set of linehaul customers
B	The set of backhaul customers
M	The set of time intervals with uniform traffic in travel time function with index m
K	The set of vehicles with index k
Parameters	
cap	Capacity of each vehicle
d_i	Demand of node i
S_i	Service time of node i
LB_{ij}^m	Lower boundary of time of travel during the m th time interval of travel time function of edge (i,j)
UB_{ij}^m	Upper boundary of time of travel during the m th time interval of travel time function of edge (i,j)
LC_{ij}^m	Lower boundary of travel time duration during the m th time interval of travel time function of edge (i,j)
UC_{ij}^m	Upper boundary of travel time duration during the m th time interval of travel time function of edge (i,j)
BN	A large number
Decision variables	
x_{ijk}^m	Equal to 1 if the edge (i,j) is traveled by vehicle k during the m th interval; otherwise, is equal to 0
u_i	Time for the vehicle to reach node i
θ_{ij}	Time to traverse the edge (i,j)
z_{ijk}^m	The time when vehicle k leaves node i towards node j during m th time interval
obj	Objective function value

3.4. Mathematical model

This section describes the proposed mathematical model.

$$\min obj = \sum_{i\in V}\sum_{j\in V}\theta_{ij} \tag{1}$$

$s.t:$

$$\theta_{ij} = \sum_{k\in K}\sum_{m\in M}\left[LC_{ij}^m x_{ij}^m + \frac{UC_{ij}^m - LC_{ij}^m}{UB_{ij}^m - LB_{ij}^m}\left(z_{ijk}^m - LB_{ij}^m x_{ijk}^m\right)\right];\ \forall i\in V, j\in V \tag{2}$$

$$z_{ijk}^m \le u_i + s_i\ ;\ \forall i,j\in V, \forall k\in K, \forall m\in M \tag{3}$$

$$z_{ijk}^{m} \le BNx_{ijk}^{m} \ ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{4}$$

$$z_{ijk}^{m} \ge u_i + s_i - BN\left(1 - x_{ijk}^{m}\right) ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{5}$$

$$\sum_{i \in V} \sum_{k \in K} \sum_{m \in M} x_{iik}^{m} = 0 \tag{6}$$

$$\sum_{i \in V} \sum_{m \in M} x_{ijk}^{m} = \sum_{i \in V} \sum_{m \in M} x_{jik}^{m} \ ; \ \forall j \in L \cup B, \forall k \in K \tag{7}$$

$$\sum_{j \in L} \sum_{m \in M} x_{0jk}^{m} = 1 \ ; \ \forall k \in K \tag{8}$$

$$\sum_{j \in B} \sum_{m \in M} x_{0jk}^{m} = 0 \ ; \ \forall k \in K \tag{9}$$

$$\sum_{i \in B} \sum_{j \in L} \sum_{k \in K} \sum_{m \in M} x_{ijk}^{m} = 0 \tag{10}$$

$$\sum_{i \in V} \sum_{k \in K} \sum_{m \in M} x_{ijk}^{m} = 1 \ ; \ \forall j \in L \bigcup B \tag{11}$$

$$u_0 = 0 \tag{12}$$

$$u_i + S_i + \theta_{ij} \le u_j + BN\left(1 - x_{ijk}^{m}\right) \ ; \ \forall i \in V, \forall j \in L \bigcup B, \forall k \in K, \forall m \in M \tag{13}$$

$$u_i + S_i + \theta_{ij} \ge u_j - BN\left(1 - x_{ijk}^{m}\right) \ ; \ \forall i \in V, \forall j \in L \bigcup B, \forall k \in K, \forall m \in M \tag{14}$$

$$u_i + BNx_{ijk}^{m} \le UB_{ij}^{m} + BN \ ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{15}$$

$$u_i \ge LB_{ij}^{m} x_{ijk}^{m} \ ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{16}$$

$$\sum_{i \in L} \sum_{j \in V} \sum_{m \in M} d_i x_{ijk}^{m} \le cap \ ; \ \forall k \in K \tag{17}$$

$$\sum_{i \in B} \sum_{j \in V} \sum_{m \in M} d_i x_{ijk}^{m} \le cap \ ; \ \forall k \in K \tag{18}$$

$$x_{ijk}^{m} \in \{0,1\} \ ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{19}$$

$$u_i, \theta_{ij}, z_{ijk}^{m} \ge 0 \ ; \ \forall i,j \in V, \forall k \in K, \forall m \in M \tag{20}$$

Eq. (1) is the objective function of the problem, which includes the minimization of the total travel time of all vehicles. Eq. (2) provides the limitation on travel time between the nodes. Eqs. (3-5) give the departure time of vehicles from the nodes. Eq. (6) and Eq. (7) guarantee the departure of vehicles from the nodes. Eq. (8) and Eq. (9) ensure that the vehicles definitely enter a linehaul node after they left the depot. Equation (10) ensures not arrival of vehicles to linehaul after leaving a backhaul. According to Eq. (11), the demand of each customer is supplied by exactly one vehicle and at one time interval. Eq. (12) shows the departure time of vehicles from the depot. Moreover, the arrival times of vehicles to the nodes are presented by Eqs. (13-16) ensure that the arrival time to each node is within one of the vehicle's departure time intervals. Eq. (17) and (18) concern the capacity of the vehicles for two groups of customers, i.e. linehaul and backhaul, available in the vehicle tour. Also, constraints (19) and (20) represent the domain of decision variables of the problem.

4. Methodology

The problem under study is an extension of the basic VRPB. As the VRPB is strongly NP-hard and can be solved in the form of a matching problem (Toth & Vigo, 2002), the current problem is at least as difficult as the VRPB to be solved. Furthermore, the efficiency of heuristic and meta-heuristic algorithms in solving the VRPB has been proved and most of the literature on the VRPB is concerned with these methods and their high-performance solutions (Granada-Echeverri et al., 2019). As a result, this paper as well employs meta-heuristic methods and an extension of such methods, called mat-heuristic methods, to solve the problem.

In the remaining, the presented meta-heuristic and mat-heuristic algorithms are described. To this end, initially, solution representation and its description are provided. Then, the algorithms designed for the problem, including variable neighborhood search meta-heuristic and mat-heuristic algorithms, are detailed.

4.1. Solution representation

Each solution for the TDVRPB is encoded as a set of routs, in which the customers appear in the order they are visited (see fiqure 3). Delimiters are used as a separator between different routes (grey color).A segment of the string is set for each customer in the given string. The point that is worth mention in the rout of each vehicle is the existence of at least one linehaul customer in the rout and how it is positioned at the beginning of the string before backhaul customers. To better represent the solution, Fig. 3 provides an example with 5 customers (set V) where customers $\{1,2,3\}$, linehaul customers (set L), and customers $\{4,5\}$, backhaul customers (set B), and two vehicles (set K) are shown.

3	1	0	2	5	4

Fig. 3. Solution representation

In the solution string related to each vehicle, customers are numbered respectively from left to right. For instance, on the tour of the first vehicle, initially, customer #3 and then customer #1 are visited. As is inferred from this tour, backhaul customers can even not be present in the tour, but the existence of at least one linehaul customer is mandatory.

4.2. Basic VNS algorithm

The variable neighborhood search (VNS) algorithm is a meta-heuristic algorithm for solving combinational and problems and global optimization. The main idea behind this algorithm is changing the neighborhood using a local search. For the first time (Mladenović & Hansen, 1997) proposed the VNS algorithm. The main idea of this algorithm is a systematic neighborhood change using a local search algorithm. The general structure of this method is such that initially several different neighborhood structures and one initial solution are determined and then, using the first neighborhood structure, starts the search process with either approach of selecting the first improvement or the best improvement. In case the solution is improved, the better solution replaces the current solution and the neighborhood structure turns back to the first state. Additionally, after several iterations and if there is no improvement in the solution of the selected neighborhood structure, it goes to the next structure. The termination criteria in the algorithm can involve items such as the limitation on the number of iterations, the solution time, the number of iterations between each two sequential improvement in the solutions, or finding an optimal local solution in the all neighborhood structure and the search process continues until one of the termination criteria is satisfied. The VNS algorithm has a simpler structure compared with other meta-heuristic algorithms and its advantage is the fewer number of parameters, which leads to a higher speed of the algorithm in comparison to its counterparts.

```
Algorithm 1: Variable Neighborhood Search
1. Input:a set of neighborhood structures $N_l$, $l = 1,2,...,l_{\max}$
2. $S$ = generate initial solution();
3. Repeat
4.         $l = 1$ ;
5.     while($l <= l_{max}$)
       {
6.         $S' = \text{Shaking}(S, N_l)$
7.         $S'^{*} = \text{Local search}(S')$
8.         if $f(S'^{*}) < f(S)$
9.             $S \leftarrow S'^{*}$
10.            $l = 1$ ;
11.        else
12.            $l = l + 1$;
       }
13. Until stoping conditions are met;
14. Output: The best solution;
```

Fig. 4. Pseudo-code of the VNS algorithm

The VNS algorithm is comprised of two main steps: Shake and local search. The aim of these steps is to establish an abrupt change in the available solution (Fleszar et al., 2009) and to find local optimum solutions. Assume that $N_l(l = 1,2,\dots,l_{max})$ is a predetermined neighborhood structure and $N_l(X)$ is the set of neighborhoods of X under structure N_l. In the shake step, the algorithm moves from the current solution (S) toward a neighbor solution (S') randomly using the lth neighborhood structure. In the local search step, the algorithm is executed on solution S' as the initial solution of neighborhood search methods so that it reaches the local optimum solution S'^* . Then to decide on moving or staying, if the local optimum solution obtained so far is better than the current solution, replaces it; otherwise, the next neighborhood structure ($l <= l_{\max}$) is used to continue the search process until the condition is satisfied while the termination criteria are not met yet. Fig. 4 provides the pseudo-code of the algorithm.

4.3. Proposed VNS algorithm

The proposed VNS algorithm consists of the following fine components.

1. Input
2. Generation of the initial solution
3. Shake
4. Local search
5. Termination criteria

Each component is explained below.

4.3.1. Input

Two sets of inputs including algorithm parameters and problem parameters are considered for the proposed algorithm.

Algorithm parameters:

These parameters are specific to the algorithm and include the following items.

1. The number of sequential times of failed executions of a shake ($m_{\max}$): If a shake is executed $m_{\max}$ sequential times on a tour and cannot help reduce the tour travel time of the related vehicle, its execution is stopped and a higher shake is executed on the same tour.

2. The minimum number of total executions of shakes until the algorithm stops ($n_{\min}$): The number of executions of shakes is recorded in the global variable NSR, and the algorithm will be stopped if the value of this variable is not smaller than parameter $n_{\min}$.

Problem parameters:

The total number of customers (n), the number of linehaul customers (m), the number (K) and capacity (cap) of vehicles, travel time functions of the network edges (UC_{ij}^m , LC_{ij}^m , UB_{ij}^m , LB_{ij}^m), and the demand amount of customers (d_i), as inputs, are fed to the algorithm as the problem parameters.

4.3.2. Initial solution generation

At first, linehaul and backhaul customers are determined. Next, in a random way, linehaul customers and then backhaul customers are allocated to the vehicles' tour. In replacing customers in the vehicles tour it should be noted that at least one linehaul customer is allocated to each tour and the capacity of vehicles for both customer types is observed. This arrangement continues until a viable initial solution in terms of the capacity of vehicles is generated.

4.3.3. Shake

In this phase of the algorithm, 18 neighborhood establishment structures are used. Shakes are sorted from the simplest shake to the most complicated shake based on the volume of changes applied to the solution, and their order of usage is determined. This means that, at first, the simplest shake and finally the most perplex shake are applied. The shake changes if no improvement is seen after 30 sequential iterations. The number 30 has been determined by Taguchi analysis. These 18 structures are described in the following.

1. Swapping two linehaul points: A tour with at least two linehaul customers is selected randomly, and two randomly selected linehaul customers of the tour are swapped.

2. Swapping two backhaul points: A tour with at least two backhaul customers is selected randomly, and two randomly selected backhaul customers of the tour are swapped.

3. Transfer of one backhaul point: In this neighborhood structure, the exchange method, presented by (Potvin & Rousseau, 1995) is used for backhaul customers. In this method, two routes are selected randomly and then a backhaul point on one of the routes is chosen randomly and is placed after linehaul customers of the other route considering the limitation on the capacity of that route's vehicle.

4. Transfer from a linehaul point: This neighborhood structure is similar to the third structure (mentioned above), but it is performed for a linehaul customer. Two remarks need to be paid attention:

   - The customer is selected randomly from a route with at least two linehaul customers.

   - The customer selected from the second vehicle's tour is placed before backhaul customers observing the capacity of the vehicle.

Shakes 5 and 6 respectively include the transfer of two backhaul points and two linehaul points, and the process is identical to items 3 and 4 described above.

5. Exchange of one backhaul point: A backhaul customer in a tour is exchanged randomly for a backhaul customer in another tour observing the capacity of vehicles.

6. Exchange of one linehaul point: A backhaul customer in a tour is exchanged randomly for a linehaul customer in another tour observing the capacity of vehicles.

Shakes 9 to 18 respectively include the exchange of two backhaul points, exchange of two linehaul points, exchange of tree backhaul points, exchange of three linehaul points, exchange of two backhaul points for one backhaul point, exchange of two linehaul points for one linehaul point, exchange of three backhaul points for one backhaul point, exchange of three linehaul points for one linehaul point, exchange of one backhaul point for two backhaul point, and exchange of three linehaul points for two linehaul points. Each of these items is performed by observing the capacity of vehicles.

4.3.4. Local search

In this algorithm, two efficient local searches, namely, $2-opt$ and $swap$ have been used, meaning that after each execution of the shake, one of these two local searches is selected randomly and applied to all solution routes resultant from executing the shake. Two remarks are worth mentioning regarding applying the local search algorithms to the solution:

1. The process of applying the local search to one tour of a vehicle continues until no further improvement is observed for the travel time of that tour.

2. In applying the local search always it has to be noted that the solution is viable, meaning that at each tour of a vehicle, initially, linehaul customers and then backhaul customers are serviced.

$2-opt$ algorithm:

This algorithm is one type of $\lambda-opt$ local search algorithms (M. W. P. Savelsbergh, 1985) that changes the sequence of a segment of the given route. For this to happen, two customers of an identical route are selected randomly and in addition to the swap of two nodes, the sequence of all nodes between them is also changed. This type of local search is shown in Fig. 5.

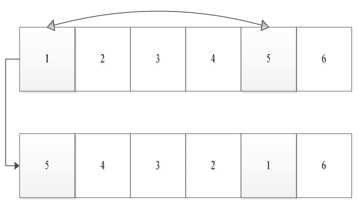

Fig. 5. 2-opt local search (the upper and lower tours are before and after applying the local search, respectively)

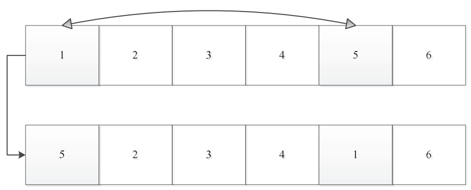

Fig. 6. Swap local search (the upper and lower tours before and after applying the local search algorithm, respectively)

swap algorithm:

In the swap algorithm, two customers of the route are selected randomly and only these two nodes are swapped (See Fig. 6.).

Two criteria are considered for termination of the proposed VNS algorithm:

1. Execution of $m_{\max}$ sequential times without any improvement in the 18th shake: If the 18th shake on the last tour for sequential times cannot improve the travel time of the vehicle.
2. Satisfying the condition $NSR >= n_{\min}$: It means that the number of execution of shakes (the global variable of NSR) has reached the default minimum number.

The algorithm will stop if both abovementioned criteria are met. This means that if the execution of the 18th shake for $m_{\max}$ sequential times has not improved the last tour of vehicles of the best solution and $NSR >= n_{\min}$ is met, then the algorithm stops; otherwise, the execution of the algorithm continues from the first shake. It is worth mentioning that during the execution of the algorithm, the number of failed executions of a shake is recorded in parameter NFSR.

The pseudo-code of the proposed VNS algorithm is shown in Fig. 7.

```
Algorithm 2 : Proposed Variable Neighborhood Search
1. Input: $N_l\ (l = 1,2,...,l_{\max}), n_{\min}, m_{\max}, NSR = 0, NFSR = 0$;
2. $S$ = Generate initial solution ();
3. Repeat
4.            $l = 1$
5.        while$(l <= l_{max}\ \&\ NSR < n_{min})$
          {
6.            $S' = \text{Shaking}(S, N_l)$
7.            $NSR = NSR + 1$
8.            $S'' = \text{Local search}(S')$
9.            if $f(S'') < f(S)$
              {
10.               $S \leftarrow S''$
11.               $NFSR = 0$
              else
12.               $NFSR = NFSR + 1$
              }
13.           if $NSR >= m_{max}$
              {
14.                   $l = l + 1$
              }
          }
15. Until stoping conditions are met;
16. Output: The best solution;
```

Fig. 7. Pseudo-code of the proposed VNS algorithm

4.4. Proposed Mat-VNS algorithm

Mat-heuristics are heuristic algorithms established by merging meta-heuristics and mathematical programming techniques. The main coordinates of this approach is the utilization of the features obtained from the mathematical model of the considered problem in some sections of meta-heuristic algorithms. The proposed Mat-NS algorithm stores a specific number of suitable solutions obtained during the execution of the proposed VNS algorithm. Then, using the obtained solution, it solves a mathematical model and extracts a solution with the minimum total travel time (minimum cost) among all routes available in the set and then applies the neighborhood on that solution. In general, one of the two following conditions is assumed for implementing the mathematical model:

1. If the number of execution of shakes has reached a specific number
2. If the number of well-produced solutions has reached a specific number

Furthermore, after implementing the mathematical model for each of the abovementioned conditions, two cases will appear for re-implementing the mathematical model:

1. Saving the solution obtained from solving the mathematical model together with other well-produced solution (inputs to the mathematical model) for re-implementing the model.

2. Saving only the solution obtained from solving the mathematical model and discarding other well-produced solutions (inputs to the mathematical model) for re-implementing the model.

Consequently, the following four scenarios are suggested for solving the mathematical model to determine the best solution among all well-produced routes:

1. Solving the mathematical model after each execution of a specific number of shakes and saving all routes for re-implementing the model.
2. Solving the mathematical model after each execution of a specific number of shakes and saving only the best routes obtained from re-implementing the model.
3. Solving the mathematical model after finding a specific number of good solutions and saving all routes for re-implementing the model.
4. Solving the mathematical model after finding a specific number of good solutions and saving only the best routes obtained from re-implementing the model.

Testing the above four scenarios on four selected test problems with 70 to 130 nodes, the fourth scenario was chosen as it provides rather better solutions with less solving time.

To elucidate on, in the presented Mat-VNS algorithm, well-produced solutions during the execution of the proposed VNS algorithm are stored in a set named $Sols$ and the number of such solutions is stored in the variable NS . Whenever the value of this variable is equal to parameter *a*, the mathematical model for selecting the best solution is executed. The rest of the algorithm components are similar to the proposed VNS algorithm. The pseudo-code of the algorithm is given in Fig. 8.

4.4.1. The mathematical model in the proposed Mat-VNS algorithm

Table 3 shows the notations used in the model.

Table 3
Notations used in the model

Sets	
V	The set of all demand nodes of the network with index i
U	The set of all tours of vehicles available in set $Sols$ with index j
Parameters	
K	The total number of vehicles
C_j	Travel time of the j th tour of vehicles
b_{ij}	is 1 if the demand node i is present in the j the tour; otherwise, is 0
Decision variables	
x_j	is 1 if the j th tour is selected; otherwise, is 0

Mathematical model:

$$\min \sum_{j\in U} C_j x_j \tag{21}$$

$$s.t:$$

$$\sum_{j\in U} b_{ij} x_j = 1 \quad ; \quad \forall i \in V \tag{22}$$

$$\sum_{j\in U} x_j = K \tag{23}$$

$$x_j \in \{0\, or\, 1\} \tag{24}$$

In this model, Eq. (21) denotes the objective of the problem, i.e. minimization of the total travel time. In addition, Eqs. (22) and (23) are model limitations showing respectively insurance of supplying the demand of each customer and the number of vehicles. Finally, Eq. (24) represents the domain of the variable.

5. Experimental proofs of the proposed methods efficiency

In order to evaluate the efficiency of the designed algorithms, a number of sample problems were randomly selected from the areas close to the designed problem and the algorithms were implemented on them. By investigating the research in the related literature, all TDVRP have a hard time window, and given that the algorithms in this article are essentially designed for problems without hard time window, their testing will not be possible with those test problems. To the best of the author's knowledge, article (Kritzinger et al., 2017) is the latest article in the field of TDVRP with soft time window, in which violation amount of time window constraint will be added to the objective function by specific penalty unit and its test problems have been designed based on (Ichoua et al., 2003), but unfortunately, access to all the information of time dependency of either of these two sources was not possible. Therefore, to test the algorithms, the CVRP test problems presented in (Augerat et al., 1995) and the VRPB test problems presented in (Goetschalckx & Jacobs-Blecha, 1989) were used. Each of the problems is executed ten times by each of the algorithms and the results are given in Tables 4 and 5, respectively. In these tables, the BKS, Best, and CPU (s) symbols represent the best-known solution, the best answer found in ten repetitions, and the execution time leading to the best answer, respectively.

```
Algorithm 3: Proposed Mat-Variable Neighborhood Search
1. Input: $N_l\,(l = 1,2,\ldots,l_{\max}), n_{\min}, m_{\max}, a, NSR = 0, NFS = 0, NS = 0, Sols = \{\}$;
2. $S$ = Generate initial solution();
3. Repeat
4.          $l = 1$
5.      $while(l <= l_{max}$ & $NSR < n_{min})$
        {
6.          $S' = \text{Shaking}(S, N_l)$
7.          $NSR = NSR + 1$
8.          $S'' = \text{Local search}(S')$
9.          if $f(S'') < f(S)$
            {
10.                 $Sols \leftarrow Sols, S''$
11.                 $S \leftarrow S''$
12.                 $NFS = 0$
13.                 $NS = NS + 1$
14.         else
15.                 $NFS = NFS + 1$
            }
16.         if $NFS >= m_{max}$
            {
17.             $l = l + 1$
            }
18.         if $NS == a$
            {
19.             $S$ = Solve mathematical model($Sols$)
                $Sols \leftarrow S$
            }
        }
20. Until stoping conditions are met;
21. Output: The best solution;
```

Fig. 8. Pseudo-code of the proposed Mat-VNS algorithm

Table 4
The obtained results from VNS and Mat-VNS algorithms for CVRP test problems

Instance	BKS	VNS			Mat-VNS		
		Best	Error (%)	CPU(s)	Best	Error (%)	CPU(s)
A-n32-k5	784	784	0	64	784	0	64
A-n34-k5	778	778	0	56	778	0	56
A-n37-k5	669	669	0	74	669	0	74
A-n39-k5	822	822	0	77	822	0	77
A-n46-k7	914	922	0.8	171	914	0	171
A-n48-k7	1073	1116	4	166	1084	1	166
A-n60-k9	1354	1394	2.9	195	1359	0.4	195
A-n63-k10	1634	1684	3	205	**1628**	0	205
A-n64-k9	1401	**1343**	0	224	**1327**	0	224
A-n69-k9	1159	1196	3.2	316	1188	2.5	316
Average			1.39	154.8		0.39	154.8

For this set of test problems, BKS is extracted from (Saha et al., 2020). In this table, better answers than BKS are shown in **bold**. For this set of test problems, BKS is extracted from (Queiroga et al., 2020). The results of both Tables 4 and 5 shows the good performance of both algorithms designed to solve two types of problems, and by increasing the dimensions of the problems and keeping the execution time constant, the Mat-VNS algorithm performs better. It is necessary to mention two facts: 1) half of the shakes of either VNS or Mat-VNS algorithms have been designed for backhaul customers, which were not used in the algorithms testing process by CVRP test problems. 2) all of the above-mentioned experiments were executed using a Core i7 CPU PC; while, in (Queiroga et al., 2020) the experiments were conducted on a supercomputer with Intel

Xeon 2.4 GHz CPU. That is the main reason for the long execution times and the errors indicated on Table 4 and Table 5, respectively.

Table 5
The obtained results from VNS and Mat-VNS algorithms for VRPB test problems

Instance	BKS	VNS			Mat-VNS		
		Best	Error (%)	CPU(s)	Best	Error (%)	CPU(s)
A3	163405	163405	0	15	163405	0	15
B2	198047	198047	0	43	198047	0	43
C2	215020	215020	0	48	215020	0	48
D3	239478	239482	0	50	239482	0	50
E1	238879	242462	1.5	55	241985	1.3	55
F1	263173	265804	1	63	265804	1	63
G3	229507	234098	2	80	233179	1.6	80
H2	253365	259432	2.4	100	258432	2	100
I2	309943	317692	2.5	170	317072	2.3	170
K1	394071	406287	3.1	300	403135	2.3	300
Average			1.25	92.4		1.05	92.4

6. Computational results

This section measures the efficiency of the proposed algorithm using the test problems presented in (Goetschalckx & Jacobs-Blecha, 1989). It has to be noted that, at first, using Taguchi analysis (Montgomery, 2017) the parameters of the proposed algorithms are tuned and then algorithms are applied to the test problems. For problems with small scale dimensions, VNS and Mat-VNS algorithms outputs were compared with the output of Cplex solver. Moreover, obtained results from solving problems with large scale dimensions prove the superiority of the proposed Mat-VNS algorithm to the designed VNS algorithm.

6.1. Generating test problems

Characteristics of the selected test problems are listed in Table 6.

Table 6
Characteristics of the selected test problems

Problem type	Problem	$\|L\|$	$\|B\|$	$\|K\|$	cap
Small scale dimensions	1	5	3	2	1500
	2	5	4	3	950
	3	5	3	3	2000
	4	5	4	2	1450
	5	5	5	2	1500
	6	6	3	2	2000
	7	6	3	3	2500
	8	7	4	4	3000
	9	7	4	3	3000
	10	8	7	5	4500
Large scale dimension	1	20	20	7	1800
	2	20	20	7	2600
	3	20	20	4	4150
	4	30	15	7	2650
	5	30	15	4	4300
	6	30	15	4	5225
	7	30	30	6	3000
	8	30	30	5	4400
	9	45	12	6	4300
	10	45	12	5	6400
	11	45	23	6	4000
	12	45	23	4	6100
	13	45	23	5	7100
	14	45	45	10	3000
	15	45	45	6	5700
	16	75	19	10	4400
	17	75	19	8	5600
	18	75	19	6	8200
	19	75	19	7	6600
	20	75	38	8	5200

5.2. Determining travel time function

In each selected test problem, five time intervals are considered for the travel time function of vehicles. As mentioned in (Goetschalckx & Jacobs-Blecha, 1989), the distance between the points in the network are assumed Euclidian; hence, taking into account this assumption, the equivalent VRPB problem of each selected test problem is solved and the length of the longest tour of vehicles is divided into four segments. Therefore, four time intervals are determined for the departure time of vehicles. Finally, the upper limit of the last (the fifth) time interval was assumed a large number. As a result, the values of LB_{ij}^{m} and UB_{ij}^{m} were determined using the travel time function.

As elaborated in Section 3.2, if the slope of the travel time function is greater than -1, the result is that the FIFO assumption will be observed in the problem as well. To this end, the speed function of vehicles at each time interval is assumed as a linear function of time. Consider that dis_{ij} denotes the length of the edge (i, j). Now, if the start and finish times of the m th time interval for the departure of vehicles on this edge are LB_{ij}^{m} and UB_{ij}^{m} the speed of vehicles in these times respectively are LS_{ij}^{m} and US_{ij}^{m}, the FIFO assumption is observed in case Eq. (25) is satisfied.

$$\left({dis_{ij}}/{US_{ij}^{m}} - {dis_{ij}}/{LS_{ij}^{m}}\right) \Big/ \left(UB_{ij}^{m} - LB_{ij}^{m}\right) \geq -1 \tag{25}$$

For simplicity, Eq. (26) is taken into account.

$$\begin{aligned} US_{ij}^{m} &= US^{m} \\ LS_{ij}^{m} &= LS^{m} \end{aligned} \tag{26}$$

Now, if D is equal to the length of the longest edge of the VRPB problem, it would suffice to determine LS^{m} and US^{m} such that Eq. (27) is met.

$$\left({1}/{US^{m}} - {1}/{LS^{m}}\right) \Big/ \left(UB_{ij}^{m} - LB_{ij}^{m}\right) \geq \left({-1}/{D}\right) \tag{27}$$

By multiplying dis_{ij} by ${1}/{LS^{m}}$ and ${1}/{US^{m}}$, parameters LC_{ij}^{m} and UC_{ij}^{m} are obtained. It is evident that the FIFO assumption is met in this case. Refer to Eq. (28).

$$\left({dis_{ij}}/{US^{m}} - {dis_{ij}}/{LS^{m}}\right) \Big/ \left(UB_{ij}^{m} - LB_{ij}^{m}\right) = \left(UC_{ij}^{m} - LC_{ij}^{m}\right) \Big/ \left(UB_{ij}^{m} - LB_{ij}^{m}\right) \geq \left(-dis_{ij}\right) \big/ D \geq -1 \tag{28}$$

5.3. Parameters tuning

Parameters of (meta- and mat-) heuristic algorithms have a significant impact on the proper performance of the algorithms. Thus, achieving the suitable values of these parameters can improve the performance of the algorithm up to a considerable extent. The Taguchi method is among the statistical methods used for parameter tuning. This method tries to determine the parameters at such a level that the efficiency of the algorithm is at its maximum and stability of the algorithm is maintained. In the Taguchi method, a ratio called signal-to-noise (S/N), instead of the solution value, is utilized for analyzing the solution. S and N in this ratio represent the level of suitability and non-suitability and the objective is to increase this value as much as possible (Montgomery, 2017). Objective functions of this method are classified into three groups depending on the problem objective:

1. More-better: In this group, it is desired to increase the objective function value.
2. Less-better: In this group, it is desired to decrease the objective function value.
3. Nominal-better: In this group, it is desired to make close the objective function of the problem to a desirable specific value.

As the objective function of this study is the minimization of the total travel time, the problem is included in the less-better group to be analyzed. Effective parameters for each algorithm are determined by trial and error, and three levels are considered for each parameter. These parameters and their related levels for VNS and Mat-VNS algorithms are given in Tables 7 and 8, respectively.

Table 7
Parameters of the proposed VNS algorithm and their levels

Parameter	Value	Level
$m_{\max}$	10	1
	20	2
	30	3
$n_{\min}$	10n	1
	20n	2
	30n	3

Table 8
Parameters of the proposed Mat-VNS algorithm and their levels

Parameter	Value	Level
$m_{\max}$	10	1
	20	2
	30	3
$n_{\min}$	10n	1
	20n	2
	30n	3
a	10	1
	20	2
	30	3

To analyze the data, Minitab software is used. Regarding the number of selected factors and levels for analysis, a suitable standard table is provided. Then, three test problems are selected randomly in average and large dimensions and, regarding the values considered at each row of the table for parameters, each problem is executed three times. The obtained values for each objective function are recorded in Minitab software. In the rest, the values of the S/N ratio for each parameter and at each level are determined. Next, averaging the obtained values from three problems, the level with the maximum ratio is considered for the parameter in the related algorithm. These levels in Table 7 and Table 8 are highlighted in grey. For instance, the output of Minitab for parameters of the VNS algorithm and with respect to one of the considered examples is shown in Fig. 9.

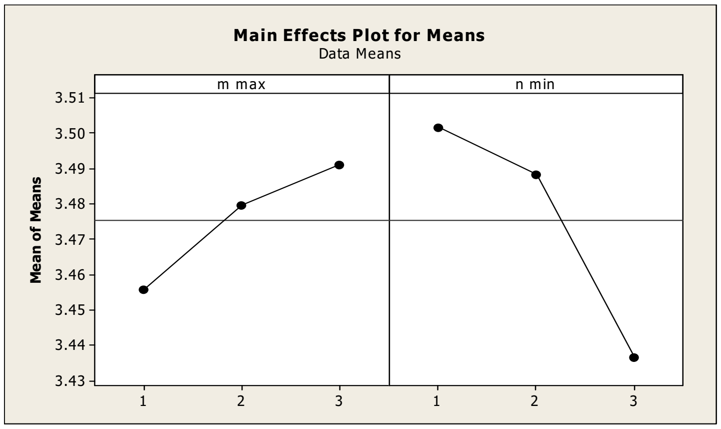

Fig. 9. The output of Minitab software for parameters of the proposed VNS algorithm

5.4. Numerical results

Concerning the designed items in the model and extension of the basic optimization methods, VSN and Mat-VNS algorithms and programs were executed using MATLAB R2015b software on a Core i7 CPU, 3 GB RAM PC. In displaying the program outputs, routes of vehicles and their travel times besides the value of the objective function (total travel time of vehicles) were determined. To evaluate the validity of the proposed algorithms, the optimal solution of small scale problems using GAMS software and Cplex solver and is compared with the solutions of VNS and Mat-VNS algorithms. The results for this group of problems are tabulated in Table 9. In this table, the error percentage is calculated from Eq. (29).

$$\left({(Obj - Obj^*)}/{Obj^*} \right) * 100 \tag{29}$$

Table 9
The obtained results from VNS and Mat-VNS algorithms and Cplex for small scale problems

problem	Cplex		VNS			Mat-VNS		
	obj^*	Time (s)	obj	Time (s)	Error (%)	obj	Time (s)	Error (%)
1	129910	21	129910	4	0	129910	4	0
2	160860	28	160860	3	0	160860	3	0
3	125470	63	125470	4	0	125470	4	0
4	124380	82	124380	5	0	124380	5	0
5	124870	250	124870	6	0	124870	6	0
6	112410	512	112410	7	0	112410	7	0
7	112780	679	112780	7	0	112780	7	0
8	139660	3965	139660	7	0	139660	7	0
9	121300	5400	121300	8	0	121300	8	0
10	202070	43200	202070	8	0	202070	8	0
Average	135371	5420	135371	5.9	0	135371	5.9	0

As it is seen from Table 9, both VNS and Mat-VNS algorithms have reached the optimum solution in all 10 given problems. It is worth noting that due to the small scale of the problems, the number of suitable solutions obtained during the process of the VNS algorithms is less than 30 (the optimal level of parameter a). Thus, the function of the Mat-VNS algorithm is similar to the VNS algorithm, but the former, as seen in Table 10, demonstrates its efficiency and superiority in solving large scale problems.

Table 10
Obtained results from VNS and Mat-VNS algorithms for large scale problems

problem	VNS			Mat-VNS		
	obj	Time	Error	obj^*	Time	Error
1	370418	18.9	1.4	365416	18.9	0
2	321371	36.8	2.1	314628	36.8	0
3	292513	78.7	1	289535	78.7	0
4	371137	38.6	1.6	365391	38.6	0
5	313728	171.6	0.5	312222	171.6	0
6	299333	146.2	1.3	295523	146.2	0
7	426321	140.5	1	421947	140.5	0
8	363768	284	1	360254	284	0
9	383760	83.5	5	365410	83.5	0
10	316434	199.8	0.4	315150	199.8	0
11	403797	241.6	2.8	392772	241.6	0
12	378090	940	0.2	377476	940	0
13	390235	371	6.9	364946	371	0
14	600657	233.8	1.8	590061	233.8	0
15	495993	830	1.9	486689	830	0
16	580786	328	0.9	575486	328	0
17	518731	285	1.3	512212	285	0
18	486924	748	4.1	467858	748	0
19	489994	465	2.9	476210	465	0
20	599145	883	3.1	581247	883	0
Average	420157	326.2	2.06	411522	326.2	0

In this table, identical to Table 8, the error percentage is obtained from Eq. (29). As one can observe in Table 8, the proposed Mat-VNS algorithm shows superior efficiency compared with the proposed VNS algorithm in solving all large scale problems. To compare these two suggested algorithms in this paper, the values of objective functions in both algorithms for problems with large dimensions are presented in Fig. 10.

5.5. The effect of time dependency assumption

By explaining a small scale problem it is shown that the application of time dependency assumption is a suitable approach for the problem under study in this research. Neglecting this assumption states the fact that travel time between points is independent of the departure time, meaning that travel time is identical between points during all hours of a day. Hence, it can be inferred that the travel speed is equal during all hours. Therefore, the objective function can be assumed as the minimization of the total distance traversed by vehicles. To this end, a problem with 6 linehaul customers, 3 backhaul customers, and 3 vehicles are taken into account. At first, the problem model is solved assuming time dependency. The optimum tour obtained from this solution for vehicles is shown in Fig11. The bold triangle represents the depot, and tours of the first, second, and third vehicles are represented by bold, dashed, and dotted arrows.

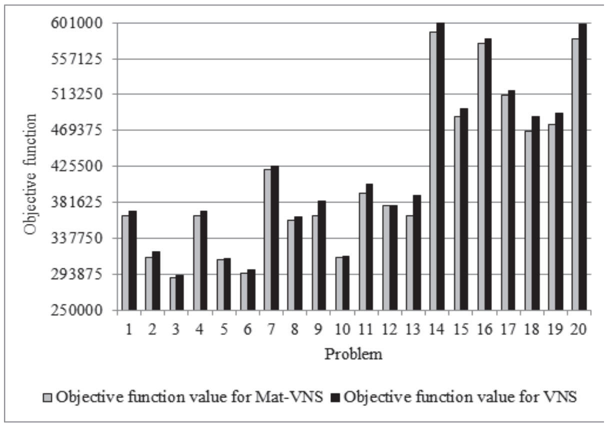

Fig. 1. The value of objective functions of VNS and Mat-VNS algorithms for problems with high dimensions

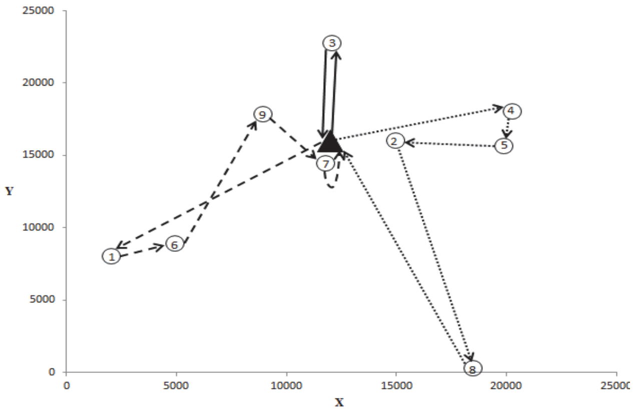

Fig. 2. The optimum tours of vehicles assuming time dependency on travel time

Considering time dependency and travel time function for network edges, the total travel time of vehicles is 140849.189 time unit. Furthermore, the distance travelled by vehicles is 90096 length unit. Nonetheless, neglecting travel time functions for network edges, the total travel time of vehicles will be 178946 time unit because traffic in the real world is overlooked while the total distance traversed is 89473 length unit and it is expected that the tour of vehicles follows suitable regularity and points of tours are close to each other as much as possible. Please refer to Fig.12.

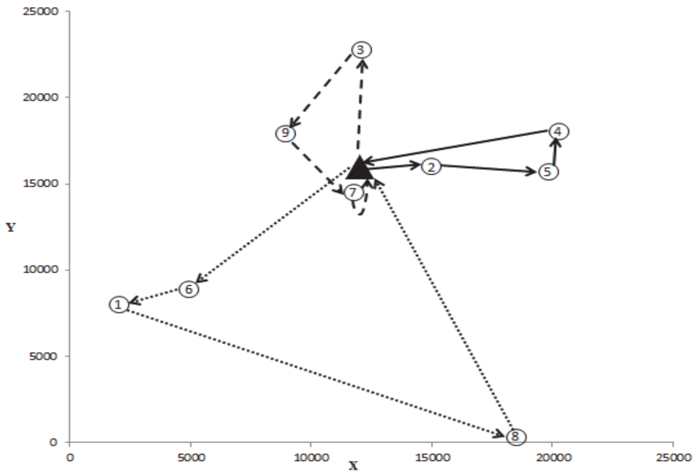

Fig. 3. Tours of vehicles by neglecting the time dependency on travel time

6. Case study

To show the applicability of the proposed model, the Post Office of Khomeini-Shahr town, Isfahan province, Iran, was studied.

Parcels are delivered to the post office of the town at 6:00 a.m. Then they are weighed, recorded, and finally separated into light and heavy parcels. Heavy parcels are those that cannot by handles using a motorcycle because of either high volume or large weight (greater than 3 kg). Distribution of such parcels is performed using a track, and the vehicle starts its travel at 8:00 a.m. After delivering the received parcels from the center to customers and the return route, it refers to the communication service offices within its mission area and besides receiving their parcels delivers them to the post office of Khomeini-Shahr. It is essential to note that servicing the communication service offices by the considered vehicle has a soft time window, meaning that receiving their parcels before 11:30 a.m. is not possible. If the vehicle arrives too early, it has to wait to start its service. Particularly, we have studied one working day of the considered vehicle. During this day, the vehicle plans to service 40 customers, among which 28 points are linehaul customers (points 1 to 28) and only 12 points are considered as backhaul customers (points 29 to 40), where it is not possible to meet them before 11:30 a.m. It is worth mentioning that consulting with the driver, the service time for each customer is set 5 min on average.

6.1. Determining travel time functions

Initially, the map of the daily service range of the vehicle was extracted and streets with traffic and traffic time intervals of each were determined, as given in Table 11.

Table 11
The traffic pattern of streets available within the study range

Street	Time interval (hour)
South Shariati	9-10 10:30-11
North Imam Khomeini	10-11:30
Amirkabir Blvd	11-12
Ashrafi esfahani Blvd	8-9
Taleghani Blvd	12-13:30

For each time interval provided in Table 10, the travel time matrix between points is determined. To this end, at each abovementioned time intervals, the travel time matrix between points was determined using Google Map (Maps.google.com) and for several different days and several times each day. Finally, at each time interval, one single travel time matrix is obtained by averaging all these matrices. As a result, the travel time function of all network edges was specified as four separate matrices as such that each matrix shows the travel time of the network edges at the beginning of each time interval 8-10, 10-12, 12-14, and 14-16.

6.2. Vehicle routing using the driver experience

The vehicle route is determined based on the driver experience. The route is as shown in Fig. 13 and its geographical representation is given in Fig. 14.

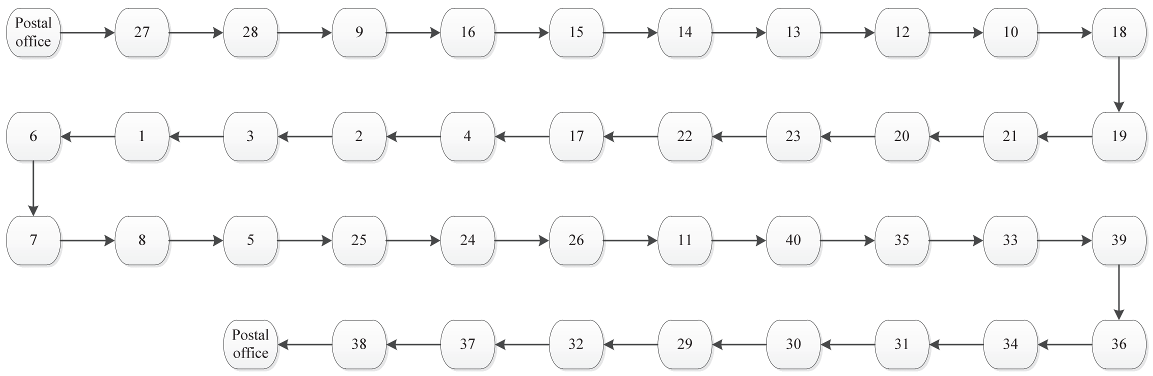

Fig. 4. The driver's considered route for servicing customers

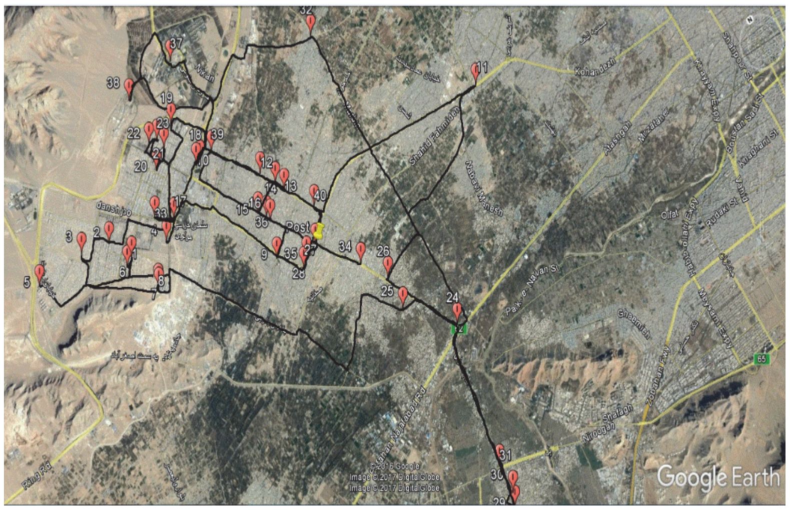

Fig. 5. Geographical representation of the deriver's considered route for servicing customers

According to this route, the mission time of the vehicle will be 435 minutes. In other words, when the vehicle's departure time from the post office is 8 a.m. and the vehicle will finally return to the office at 15:15, the travel time of the vehicle between points of the network (subtracting the service time from the mission time) will be 5 hours and 35 minutes. As mentioned earlier, servicing backhaul customers cannot be provided before 11:30; hence, this should be observed for the solutions as well. As is shown in Figure.13, the first backhaul customer on this route is node 40 which is visited at 12:38.

6.3. Optimum route

The optimum route for the vehicle, considering route traffics and based on the objective of minimization of the travel time of the vehicle, is given as Figure.15 by solving the related mathematical model. The geographical representation of the solution is shown in Fig. 16.

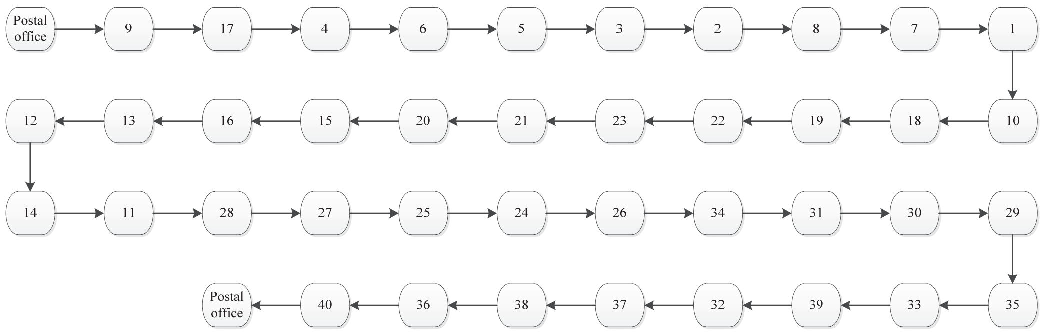

Fig 6. The optimum route for servicing customers

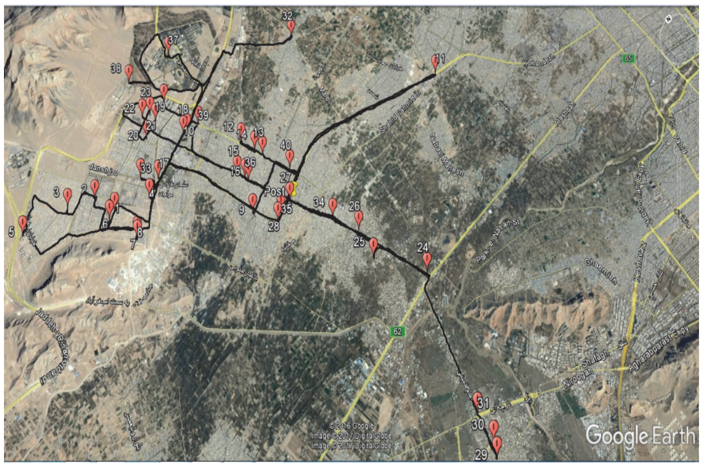

Fig 7. Geographical representation of the optimum route for servicing customers

With regard to the optimum route, the mission time of the vehicle will be 390 minutes. In other words, the vehicle after leaving the post office at 8 a.m. will return to the office at 14:30 and the travel time between the points will be 190 minutes. It has to be remembered that in this case the average service time of each point is considered 5 minutes. To put it simply, a 10.34% improvement in the mission time of the vehicle is observed (the mission time of the vehicle has reduced by 45 minutes). Moreover, the travel time of the vehicle has improved by 19.15% (the travel time of the vehicle has reduced by 45 minutes). In this approach also the arrival time to the first backhaul node (node 34) is 12:11.

6.4. Sensitivity analysis of the departure time of the vehicle

In this section, we attempt to answer the question that what are the impacts of 15 minutes haste in the departure time of the vehicle from the post office on the optimum route? Through consulting with the experts of the post office we concluded that the route traffics are uniform between 7 to 8 a.m., hence we determined the travel time matrix of the network edges similar to four time intervals before this period.

6.4.1. Optimum route with the departure time at 7:45

In this case, the optimum tour of the vehicle will be as shown in Figure. 17.

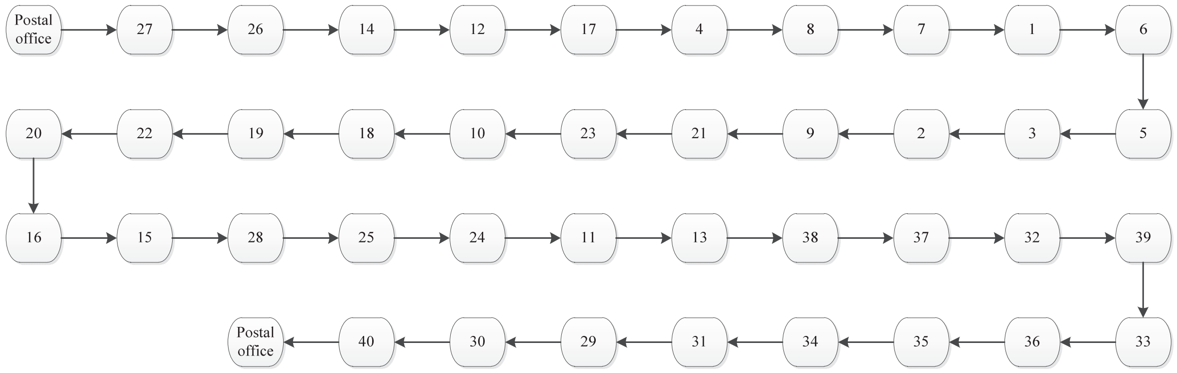

Fig. 8. The optimum route for servicing customers with the departure time at 7:45

According to this route, the mission time of the vehicle will be 337 minutes. In other words, when the vehicle leaves the post office at 7:45 and returns at 13:22, the travel time of the vehicle between points is 137 minutes. Compared with the driver's experience, there is a 22.53% improvement in the total mission time (1 hour and 38 minutes) or 98 minutes, i.e. almost 42%, improvement in the travel time of the vehicle. In this case, the time the vehicle reaches the first backhaul customer (node 38) is 11:30.

7. Conclusions and perspectives

This paper presents a novel MIP model for the time-dependent VRP with backhaul. To adapt more to the real-world problems, the FIFO assumption is included in the model with a new approach. Two variable neighborhood search (VNS) meta-heuristic and mat-heuristic algorithms were implemented to solve the model in different design dimensions and in test problems. The results highlight the efficiency and high performance of the proposed mat-heuristic algorithm compared with the meta-heuristic algorithm in solving large scale problems. Finally, to verify and prove the validity of the model and solution methods, a case study in the post office of Khomeini-Shahr town, Iran, was investigated. The obtained results from the study show roughly 19% (almost 45 minutes) reduction in the travel time of the vehicle. Moreover, this amount of reduction along with 15 minutes haste in the departure time of the vehicle adds up to 42% (about 98 minutes).

As recommendations for future developments of the model, we can take into account transportation risks, time windows for customers, uncertainty in the demand value of customers with approaches such as fuzzy or random scheduling, multi-graph feature, using two or several contradictive objectives and other similar topics. Additionally, using other solving methods such as Lagrangian relaxation or the Column generation may lead to high-performance solutions.

Acknowledgment

The authors thank the personnel of the post office of Khomeini-Shahr town for their cooperation.

References

Alinaghian, M., Ghazanfari, M., Norouzi, N., & Nouralizadeh, H. (2017). A novel model for the time dependent competitive vehicle routing problem: Modified random topology particle swarm optimization. *Networks and Spatial Economics*, *17*(4), 1185–1211.

Arab, R., Ghaderi, S. F., & Tavakkoli-Moghaddam, R. (2018). Bi-objective inventory routing problem with backhauls under transportation risks: two meta-heuristics. *Transportation Letters*, 1–17.

Augerat, P., Belenguer, J. M., Benavent, E., Corberán, A., Naddef, D., & Rinaldi, G. (1995). *Computational results with a branch and cut code for the capacitated vehicle routing problem* (Vol. 34). IMAG.

Brandao, J. (2006). A new tabu search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, *173*(2), 540–555.

Brandão, J. (2016). A deterministic iterated local search algorithm for the vehicle routing problem with backhauls. *Top*, *24*(2), 445–465.

Delgado-Antequera, L., Caballero, R., Sánchez-Oro, J., Colmenar, J. M., & Martí, R. (2020). Iterated greedy with variable neighborhood search for a multiobjective waste collection problem. *Expert Systems with Applications*, *145*, 113101.

Fleszar, K., Osman, I. H., & Hindi, K. S. (2009). A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, *195*(3), 803–809.

Gajpal, Y., & Abad, P. L. (2009). Multi-ant colony system (MACS) for a vehicle routing problem with backhauls. *European Journal of Operational Research*, *196*(1), 102–117.

Goetschalckx, M., & Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, *42*(1), 39–51.

Granada-Echeverri, M., Toro, E., & Santa, J. (2019). A mixed integer linear programming formulation for the vehicle routing problem with backhauls. *International Journal of Industrial Engineering Computations*, *10*(2), 295–308.

Haghani, A., & Jung, S. (2005). A dynamic vehicle routing problem with time-dependent travel times. *Computers & Operations Research*, *32*(11), 2959–2986.

Hill, A. V, & Benton, W. C. (1992). Modelling intra-city time-dependent travel speeds for vehicle scheduling problems. *Journal of the Operational Research Society*, *43*(4), 343–351.

Hou, L., Li, D., & Zhang, D. (2018). Ride-matching and routing optimisation: Models and a large neighbourhood search heuristic. *Transportation Research Part E: Logistics and Transportation Review*, *118*, 143–162.

Huang, Y., Zhao, L., Van Woensel, T., & Gross, J.-P. (2017). Time-dependent vehicle routing problem with path flexibility. *Transportation Research Part B: Methodological*, *95*, 169–195.

Ichoua, S., Gendreau, M., & Potvin, J.-Y. (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, *144*(2), 379–396.

Keskin, M., Laporte, G., & Çatay, B. (2019). Electric Vehicle Routing Problem with Time-Dependent Waiting Times at Recharging Stations. *Computers & Operations Research*.

Kritzinger, S., Doerner, K. F., Hartl, R. F., Kiechle, G. Ÿ., Stadler, H., & Manohar, S. S. (2012). Using traffic information for time-dependent vehicle routing. *Procedia-Social and Behavioral Sciences*, *39*, 217–229.

Kritzinger, S., Tricoire, F., Doerner, K. F., Hartl, R. F., & Stützle, T. (2017). A unified framework for routing problems with a fixed fleet size. *International Journal of Metaheuristics*, *6*(3), 160–209.

Malandraki, C. (1989). *Time dependent vehicle routing problems: Formulations, solution algorithms and computational experiments*.

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

Malandraki, C., & Dial, R. B. (1996). A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, *90*(1), 45–55.

Mancini, S. (2014). Time dependent travel speed vehicle routing and scheduling on a real road network: the case of Torino. *Transportation Research Procedia*, *3*, 433–441.

Mingozzi, A., Giorgi, S., & Baldacci, R. (1999). An exact method for the vehicle routing problem with backhauls. *Transportation Science*, *33*(3), 315–329.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John wiley & sons.

Mugayskikh, A. V, Zakharov, V. V, & Tuovinen, T. (2018). Time-Dependent Multiple Depot Vehicle Routing Problem on Megapolis Network under Wardrop's Traffic Flow Assignment. *2018 22nd Conference of Open Innovations Association (FRUCT)*, 173–178.

Osman, I. H., & Wassan, N. A. (2002). A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. *Journal of Scheduling*, *5*(4), 263–285.

Potvin, J.-Y., & Rousseau, J.-M. (1995). An exchange heuristic for routeing problems with time windows. *Journal of the Operational Research Society*, *46*(12), 1433–1446.

Queiroga, E., Frota, Y., Sadykov, R., Subramanian, A., Uchoa, E., & Vidal, T. (2020). On the exact solution of vehicle routing problems with backhauls. *European Journal of Operational Research*.

Rincon-Garcia, N., Waterson, B., Cherrett, T. J., & Salazar-Arrieta, F. (2018). A metaheuristic for the time-dependent vehicle routing problem considering driving hours regulations–An application in city logistics. *Transportation Research Part A: Policy and Practice*.

Saha, B., Suthar, K., & Kumar, A. (2020). Optimizing Generalized Capacitated Vehicle Routing Problem Using Augmented Savings Algorithm. In *Computational Intelligence in Data Mining* (pp. 527–541). Springer.

Salhi, S., Wassan, N., & Hajarat, M. (2013). The fleet size and mix vehicle routing problem with backhauls: Formulation and set partitioning-based heuristics. *Transportation Research Part E: Logistics and Transportation Review*, *56*, 22–35.

Savelsbergh, M., & Van Woensel, T. (2016). 50th anniversary invited article—city logistics: Challenges and opportunities. *Transportation Science*, *50*(2), 579–590.

Savelsbergh, M. W. P. (1985). Local search in routing problems with time windows. *Annals of Operations Research*, *4*(1), 285–305.

Setak, M., Habibi, M., Karimi, H., & Abedzadeh, M. (2015). A time-dependent vehicle routing problem in multigraph with FIFO property. *Journal of Manufacturing Systems*, *35*, 37–45.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*(2), 254–265.

Soon, K. L., Lim, J. M.-Y., Parthiban, R., & Ho, M. C. (2019). Proactive eco-friendly pheromone-based green vehicle routing for multi-agent systems. *Expert Systems with Applications*, *121*, 324–337.

Toth, P., & Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *Transportation Science*, *31*(4), 372–385.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. SIAM.

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2014). A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, *234*(3), 658–673.

Zhang, T., Chaovalitwongse, W. A., & Zhang, Y. (2014). Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery. *Journal of Combinatorial Optimization*, *28*(1), 288–309.