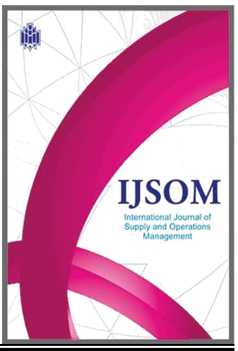
International Journal of Supply and Operations Management

IJSOM

August 2022, Volume 9, Issue 3, pp. 299-320
ISSN-Print: 2383-1359
ISSN-Online: 2383-2525
www.ijsom.com



The Selective Full Truckload Multi-depot Vehicle Routing Problem with Time Windows: Formulation and a Genetic Algorithm

Karim EL Bouyahyiouy*[a] and Adil Bellabdaoui[a]

[a] *Information Technology and Management, ENSIAS - Mohammed V University, Rabat, Morocco*

Abstract

The process of an empty backhaul truck returning to its home domicile after a regular delivery journey has attracted many logistics companies in the modern world economy. This paper studies a selective full truckload multi-depot vehicle routing problem with time windows (SFTMDVRPTW) in an empty return scenario. This problem aims at planning a set of backhaul routes for a fleet of trucks that serve a subset of selected transportation demands from a number of full truckload orders to maximize the overall profit, given constraints of availability and time windows. After reviewing the literature related to full truckload vehicle routing problems, based on the professional characteristics encountered as well as on the resolution approaches used, we formulate a mixed-integer linear programming (MILP) model for the SFTMDVRPTW. Since the problem is NP-hard, we propose a genetic algorithm (GA) to yield a near-optimal solution. A new two-part chromosome is used to represent the solution to our problem. Through a selection grounded on the elitist and roulette method, a new crossover operator called "selected two-part crossover chromosome (S-TCX)" and an exchange mutation operator, new individuals are generated. The proposed MILP model and GA are evaluated on newly randomly generated instances. The findings prove that the GA significantly outperforms the CPLEX solver in solution quality and CPU time.

Keywords: Full truckload; Time windows; Empty return trip; MILP; Genetic algorithm; Two-part chromosome crossover.

1. Introduction

The vehicle routing problem (VRP) is one of the most studied combinatorial optimization problems in transport and logistics. The issue is an extension of the famous travelling salesman problem (TSP) in which one or more vehicles travel around a network, departing from and returning to a depot node. Customers are located on the network, and each customer must be visited by exactly one vehicle. The problem objective is to plan a set of vehicle routes at a minimum cost. This planning involves two decisions: the assignment of each customer to a specific route and the visiting sequence of the customers on the routes. The Capacitated vehicle routing problem (CVRP) is the most widely practiced variant of VRP. In this way, there is a demand associated with each customer, representing an amount of goods that must be collected (or delivered). Each vehicle has a capacity that cannot be exceeded, and when the vehicle is full (or empty), it returns to the depot (Lysgaard et al., 2004; Uchoa et al., 2017). To date, a large number of CVRP variants with additional constraints have been proposed and studied in the literature; a survey of such variants can be found in (Toth and Vigo, 2002; Laporte, 2009). The predominant types of CVRP include the multi-depot VRP (MDVRP) in which more than one depot is considered (Mirabi et al., 2016; Lahyani et al., 2019); the VRP with time windows (VRPTW) in which each customer must be serviced within a time window (Solomon, 1987; Yu et al., 2011; Bouziyane et al., 2018); the heterogeneous VRP (HVRP) in which the vehicles have different load capacities, fixed or variable cost, driving time restrictions, or different speeds, etc. (Bettinelli et al., 2011; Bolaños et al., 2018); the selective VRP (SVRP) in which each potential customer has an associated profit and both the number of vehicles and the maximum travel time each vehicle can travel are limited.

Corresponding author email address: karim.bouyahyaoui2015@gmail.com
DOI: 10.22034/ijsom.2022.109076.2168

As a result, it is not obligatory to visit all customers, and only profitable ones are served within a certain time period (Aras et al., 2011; Lahyani et al., 2017; Rincon-Garcia et al., 2017); the trucking company provides only full truckload (FTL) transportation, where orders are transported directly from the pickup location to the delivery location, using a fleet of trucks located at one or several depots (FTVRP) (Ball et al., 1983; Desrosiers et al., 1988; Gronalt et al. 2003). The FTVRP is essential in many industries, such as food and beverage, consumer packaged goods, building and materials, oil and gas, manufacturing, automotive. Despite its prominence, FTVRP has not been much studied vigorously as other VRP variants; only a few studies treated this problem without focusing on the empty back return of trucks. We believe that it has not been given its due merit when it comes to research in the field of operational research. This is why we are motivated and interested in studying this issue of FTL.

In a particular context of an empty return scenario, after a delivery trip of its customers' freight has been made from its original point A of departure (his home domicile) to its destination B, a backhaul truck is obliged to return to its home depot within a precise time so that the regular transportation program will not be disturbed. It has to select the best orders (transportation demands) out of all the ones available in the transportation network on its way back to improve utilization and profitability. Transportation demands have high expectations of service quality, requiring on-time delivery from a loading location to an unloading location within a small pickup and delivery time window. This article has studied a full truckload transportation problem in the context of an empty return scenario, particularly an order selection and vehicle routing problem with full truckload, multiple depots and time windows (SFTMDVRPTW). The aim is to construct a set of backhaul truck routes that serve a subset of selected transportation demands from a number of full truckload orders to maximize the total profit. Each truck route is an arrangement of selected transportation demands to serve, originating at a departure point and terminating at an arriving point of the truck in a way that respects the constraints of availability and time windows. The overall profit will depend on the selection of transportation demands and the routing sequence.

There are several concerns related to this work, such as reducing the number of empty returns, increasing efficiency in the use of equipment of transport companies, reducing the cost of operations of transport companies, improving service, reducing carbon emissions, etc. They are noting that no matter how the constraints of the problem are, it remains NP-hard. This means that no recognized algorithm can guarantee to find, in polynomial time, the exact solution of large-sized instances of these problems. Therefore, it will be judicious to use a meta-heuristic algorithm to solve this problem. The problem presented here is prevalent in the transportation industry. However, to the best of the authors' knowledge, there has been a little study about it in the literature. EL Bouyahyiouy and Bellabdaoui (2016, 2017) have only presented a brief description of the problem in a partial way; they have tried to adapt some heuristic approaches in a paper published in a conference proceeding that exposed the first result of a small instance, including one central depot. Thus, the contribution of this paper is as follows:

- We present a variant of the FTL transportation problem inspired by logistic transportation in the context of an empty return scenario, which comprises various constraints such as selective transportation demands, MD, loading and unloading time windows of orders, etc.

- We formulate our SFTMDVRPTW problem as a mixed-integer linear programming (MILP) model.

- We propose a genetic algorithm that can be used to solve large-sized problems efficiently.

The remainder of the paper is organized as follows. In Section 2**,** we review the relevant literature on the FTL transportation problem. We describe the SFTMDVRPTW and present a mathematical formulation in Section 3. The proposed genetic algorithm is elucidated in Section 4. The computational tests and analyses are presented in Section 5. We give some conclusions and suggest directions for future research in Section 6.

2. Related works

This section reviews the previous research on FTVRPs, based on the professional characteristics encountered and the resolution approaches used. Table 1 summarizes the FTL literature and the characteristics of the present study.

To our knowledge, the first efforts in the study of the FTL transportation problem dated to the 80s. Ball et al. (1983) were the firsts that have introduced the FTVRP, which is about creating routes for private trucks and subcontracting orders of chemical products to common carriers. It can be seen as a full truckload multi-depot pickup and delivery problem (FTMDPDP). The objective function is to reduce the cost while meeting maximum route-time restrictions. Three heuristics are suggested to resolve the problem. One of them is a greedy insertion procedure (GI); others are based on the route-first, cluster-second (RF-CS) method. Later, Desrosiers et al. (1988) presented an exact approach for the FTMDPDP, transforming it into an asymmetric traveling salesman problem (aTSP). There are no TWs requirements at pickup and delivery points, but a time limit is imposed on truck routes. The goal is to minimize empty movements of trucks. To resolve the problem, a branch-and-bound (BB) method is applied. Arunapuram et al. (2003) have introduced FTVRP with multiple depots and pickup TWs. They formulated the problem as an integer programming (IP) formulation, and they develop a column generation (CG) approach inside a BB method to solve it. Gronalt et al. (2003) have presented an exact formulation for FTPDPTW where full truckload orders are moved from and to distribution centers. Trucks are available at various warehouses and can be assigned several tours throughout the time of planning. The authors developed four different savings-based heuristics that are compared on randomly generated instances. Venkateshan and Mathur (2011) have extended the FTVRP for heterogeneous trucks, where multiple visits are required by the same or different trucks to satisfy each order. Relying on the notations of Arunapuram et al. (2003), Venkateshan and Mathur (2011) formulated this problem as an IP problem, and they developed an exact approach from CG to resolve it. Currie and Salhi (2003, 2004) have tackled FTPDPTW with heterogeneous goods and trucks, where the location of orders' pickup is undetermined. Their 2003 work presented a MILP model whose objective is to minimize the total costs. Then, they proposed a hybrid method combining a GI heuristic with three neighborhood operators. Thereafter, these would be used in a tabu search (TS) algorithm (Currie and Salhi, 2004). Gronalt and Hirsch (2007) studied a FTMDPDPTW in the context of log-truck scheduling with the goal of minimizing the total duration of trucks' empty movements. They have proposed a TS to solve 20 instances with 30 to 85 orders and 9 to 28 trucks.

Het: heterogeneous fleet, MD: multi-depot, S: selective, RD: route duration, TD: total distance, NV: number of vehicles, EVM: Empty-Vehicle-Movement, E: exact, H: heuristic, SSBM: single solution based metaheuristic, PBM: population-based metaheuristic, DA: deterministic annealing algorithm, R: real data, RG: randomly generated data. Liu et al. (2010a) proposed an exact formulation and a two-phase GI heuristic to study FTL capacitated ARP with multiple depots (MDCARPFL) in the context of carrier collaboration with the aim of reducing empty movements of trucks. Liu et al. (2010b) developed a memetic algorithm for FVRP with order selection in collaborative transportation. It is assumed that carriers receive two types of orders: one given by their customers, while the other by outer carriers. The orders from the first type have to be served by one of the carriers' internal vehicles or assigned to an external collaborative transporter, but a penalty cost is incurred. For each order given by outer partners, the carriers can refuse or process it with a compensative payment. The goal is to come up with a set of private truck routes that serve a subset of selected orders, aiming at minimizing the total cost minus the compensative payments for accepting orders from other carriers. Grimault et al. (2017) investigated the FTPDPTW with resource synchronization that appears in the context of public-work companies. The problem consists of optimizing the transport of materials between sites, using a fleet of heterogeneous trucks. The objective is to minimize the total cost (traveling, service time, truck utilization). They propose an Adaptive Large Neighborhood Search (ALNS) to solve real instances.

Table 1. Summarizing the most related studies to the current study

Article	Acronym	Constraints					Objective function	Solution approach		Instances	
		TW	Het	MD	S	FTL		type	Method	type	# orders
Ball et al. (1983)	FTMDPDP			●	●	●	Min. cost	H	• RF-CS • Greedy insertion	R	Up to 200
Desrosiers et al. (1988)	VRPFL			●		●	Min. EMV	E	BB	RG	Up to 104
Wang and Regan (2002)	FLPDPTW	●		●	●	●	Min. EMV	H	WPB	R	Up to 75
Arunapuram et al. (2003)	VRPFL	●		●		●	Min. cost	E	CG	RG	Up to 200
Gronalt et al. (2003)	FLPDPTW	●		●		●	Min. EMV	H	savings algorithm	RG	Up to 512
Currie and Salhi (2003)	FTPDPTW	●	●	●		●	Min. cost	E H	• MILP • 3-phase CH	R RG	Up to 208 Up to 500
Currie and Salhi (2004)	FTPDPTW	●	●	●		●	Min. cost	SSBM	TS	R RG	Up to 208 Up to 500
Jula et al. (2005)	FTPDPTW	●				●	Min. cost	E PBM	• 2Phase DP • Hybrid DP with AG	RG	Up to 20 Up to 100
Imai et al. (2007)	VRPFC		●			●	Min. cost	H	LG	RG	Up to 200
Gronalt and Hirsch (2007)	FT-PDPTW	●		●		●	Min. EMV	SSBM	TS	R	Up to 85
Caris and Janssens (2009)	FTPDPTW	●				●	Min. cost	H	• 2PIH • LS	RG	Up to 200
Zhang et al. (2009)	am-TSPTW	●		●		●	Min. RD	E SSBM	• CPLEX • RTS	RG	Up to 200
Liu et al. (2010b)	SFTVRP				●	●	Min. cost	PBM	MA	RG	Up to 400
Zhang et al. (2010)	am-TSPTW	●		●		●	Min. RD	H SSBM	• WPB • RTS	RG	Up to 75
Liu et al. (2010a)	FTMDCVRP			●		●	Min. EVM	H	2PH	RG	Up to 300

Venkateshan and Mathur (2011)	FTPDP		●	●		●	Min. cost	E	CG	RG	Up to 20
Braekers et al. (2013)	FTPDPTW	●				●	Min. NV Min. TD	SSBM	DA	RG	Up to 200
Nossack and Pesch (2013)	FTPDPTW	●		●		●	Min. RD	H	2-stage heuristic	RG	Up to 75
Braekers et al. (2014)	FTPDPTW	●				●	Min NV Min. TD	H SSBM SSBM	• IM • DA • hybrid DA with TS	RG	Up to 200
Grimault et al. (2017)	FTPDP-RS	●	●	●		●	Min. cost	SSBM	ALNS	R	Up to 85
Xue et al. (2021)	MS-FTPDPTW	●				●	Min. cost	SSBM	hybrid CG with GA/VNS	RG	Up to 127
This paper	**SFTMDVRPTW**	●		●	●	●	**Max. profit**	**E**	**• CPLEX** **• GA**	**RG**	**Up to 30**

Container drayage problem (CDP) represents a special class of FTPDP **(**Braekers et al., 2013**)**. The goal is to carry containers, representing full truckloads, between customers and container terminals. When there are TWs at customers or terminals, it becomes FTPDPTW. When we load a merchandise at a loading point in the FTL case, we are obliged to unload it at the unloading point in the next step. So, the FTPDPTW can be transformed into an asymmetric multiple traveling salesman problem with time windows (am-TSPTW) by considering each pickup-delivery pair as a node (Wang and Regan, 2002; Jula et al., 2005; Zhang et al., 2009; Braekers et al., 2013). Wang and Regan (2002) studied FTPDPTW, in which only pickup TWs are considered. The objective is to reduce the empty truck movement cost while the number of served orders within their TWs is as large as possible. The authors developed an iterative method for solving the problem using the window-partition-based (WPB) method. Later, Jula et al. (2005) extended the earlier work considering TWs at both pickup and delivery locations. The authors propose and compare three solving approaches: an exact two-phase algorithm based on dynamic programming (DP) for small instances with up to 20 orders, a hybrid approach consisting of DP and GA, and an insertion heuristic method for large instances. Imai et al. (2007) addressed the same problem of Gronalt et al. (2003) without TWs, arising during the pickup and delivery of full containers that transport them between intermodal terminals. They present a new formulation for this problem as a 0-1 linear programming (LP) model and propose a Lagrangian relaxation (LR) to solve the mathematical formulation. Later, Caris and Janssens (2009) have extended the problem definition of Imai et al. (2007) to FTPDPTW by imposing hard TWs at the customer locations and the depot. Their solution approach employed a two-phase insertion heuristic to construct an initial solution, which was then improved by a local search. Zhang et al. (2009) defined an integrated CDP that handles full and empty containers between customers, one single terminal and multiple depots. Trucks don’t have to return to their departure depot; hard TWs are defined at the customer locations and the terminal. The objective is to minimize total travel time. The problem is formulated as am-TSPTW with multiple depots and is solved by a reactive tabu search (RTS) algorithm. Zhang et al. (2010) extended the work of Zhang et al. (2009) to multiple terminals and multiple depots case. In order to solve the problem, the others modified the WPB method presented by Wang and Regan (2002).The proposed method has been tested on randomly generated instances with up to 75 orders and compared with the RTS in Zhang et al. (2009). Inspired by Zhang et al. (2010), Nossack and Pesch (2013) came with a new formulation for FTPDPTW that aims at minimizing the total operating time of all trucks in use. They propose a two-phase heuristic algorithm for solving the FTPDPTW. The results indicate that their 2-stage heuristic outperforms the WPB method used by Zhang et al. (2010) in terms of CPU time and solution quality. Braekers et al. (2013, 2014) studied FTVRP in drayage operation that is very similar to the ones reviewed by Zhang et al. (2009, 2010). Braekers et al. (2013) propose two formulations that are based on an am-TSPTW problem. A hierarchical objective function, which first minimizes the number of vehicles and second minimizes the total distance, is proposed. For both formulations, they develop a single-phase and a two-phase DA. Braekers et al. (2014) extended their previous work by considering, for the first time, two objectives with equal priority: minimizing the number of vehicles and minimizing total distances. They propose three solution algorithms: iterative method, DA, and hybrid of DA and TS. They concluded that among the three methods, the hybrid algorithm DA-TS yielded the best results. Xue et al. (2021) presented a hybrid CG with GA and VNS for the FTVRPTW with multiple shifts; TWs of transportation containers cover several shifts, and each order's shift identification is part of decision-making.

This study focuses on a vital FTL transportation problem in an empty return scenario called SFTMDVRPTW. The problem is characterized by several attributes: FTL, selective orders, loading and unloading TWs of the orders, waiting times, profit maximization, departure and arrival point for each truck subjected to time limit constraints. In contrast to most of the existing studies on FTL, the departure time of each truck from its starting location is a decision variable. The selective aspect, which relaxes the constraint that all orders should be serviced, refers to the situation where commercial trucks return back empty and must select certain orders out of all the orders placed in the network. However, Ball et al.’s, Wang and Regan’s and Liu et al.’s studies deal with the selective orders in the context of carrier collaboration, mainly the exchange of orders between carriers. There are three highlights in our work. First, we introduce an original selective

FTL transportation problem, which combines several attributes as shown in Table 1, and accounts for characteristics that are not yet addressed in the literature but required in actual applications. Second, a MILP model is formulated for this problem. Third, we propose a genetic algorithm that can be used to solve large-sized instances efficiently, whereas in **EL Bouyahyiouy and Bellabdaoui** (**2017**), we have only presented a brief description of the problem in a paper published in a conference proceeding; an ant colony optimization algorithm was used to solve a small instance, including one central depot.

3. Description and mathematical formulation of the SFTMDVRPTW

In this section, we formally define the SFTMDVRPTW and then formulate a MILP model for the problem

3.1 Problem description

3.1.1 SFTMDVRPTW structure

Consider a set $K = \{1, \dots, m\}$ of m backhaul trucks in a common transportation network, and a set $O = \{O_1, \dots, O_n\}$ of n orders that need to be transported. The SFTMDVRPTW can be defined on an oriented graph $G = (V, E)$, where V represents the node set and E is the arc set representing the feasible links. The nodes are the points of extremity $\{(L_1, U_1), \dots, (L_n, U_n)\}$ of the n orders that are connected to two sets of points: $D = \{D_1, \dots, D_m\}$ and $A = \{A_1, \dots, A_m\}$, specifying the departure and arrival locations of trucks, respectively.

3.1.2 Constraints

Selective route. The truck routes may not necessarily include all orders but rather select only those that are profitable.
*Full truckload***.** Each truck can handle one order at a time. Each order O_i is defined by a loading point L_i and an unloading point U_i, a revenue p_i, a processing duration time $t_i$which equals the sum of the loading time, the traveling time from loading to unloading point and the unloading time. Like what is popular in the literature (e.g., Powell et al., 1988; Zolfagharinia and Haughton, 2014), the revenue associated with each order is proportional to the distance between its loading and unloading points.
Time windows. Each order O_i has a loading time window $\left[L_i^{min},\ L_i^{max}\right]$ and unloading time window $\left[U_i^{min},\ U_i^{max}\right]$. If a truck arrives early at a loading or unloading point, it has to wait until the start of loading or unloading time window. A truck cannot finish its loading (or unloading) after time L_i^{max} (or U_i^{max}).
Multiple depots. Each backhaul truck k $(k \in K)$ must start from its departure point D_k (its final delivery destination after a regular delivery route) no earlier than a time D_k^{min} and ends at its arrival point A_k (its home domicile) no later than a time A_k^{max}. Thus, each truck k is subject to time constraints in the interval $\left[D_k^{min},\ A_k^{max}\right]$.

3.1.3 Objective

The aim is to design a solution composed of a set of truck routes T_k $(k \in K)$ to maximize the overall profit that the trucks generate during their backhaul routes. Each backhaul route T_k is an ordered sequence of chosen orders to serve, originating at the departure point D_k and terminating at the arrival point A_k so that the constraints of availability and time windows are respected. Therefore , when there is no delivery order assigned to a truck k, the truck route T_k will be the shortest path from point D_k to point A_k to get the lowest cost. Figure 1 illlustrates an example of the SFTMDVRPTW involving nine orders and two trucks. Truck 1 has moved orders 1,4, 7, and 9, respectively. Truck 2 completed orders 2, 6, 5, and 3; order eight is unselected.

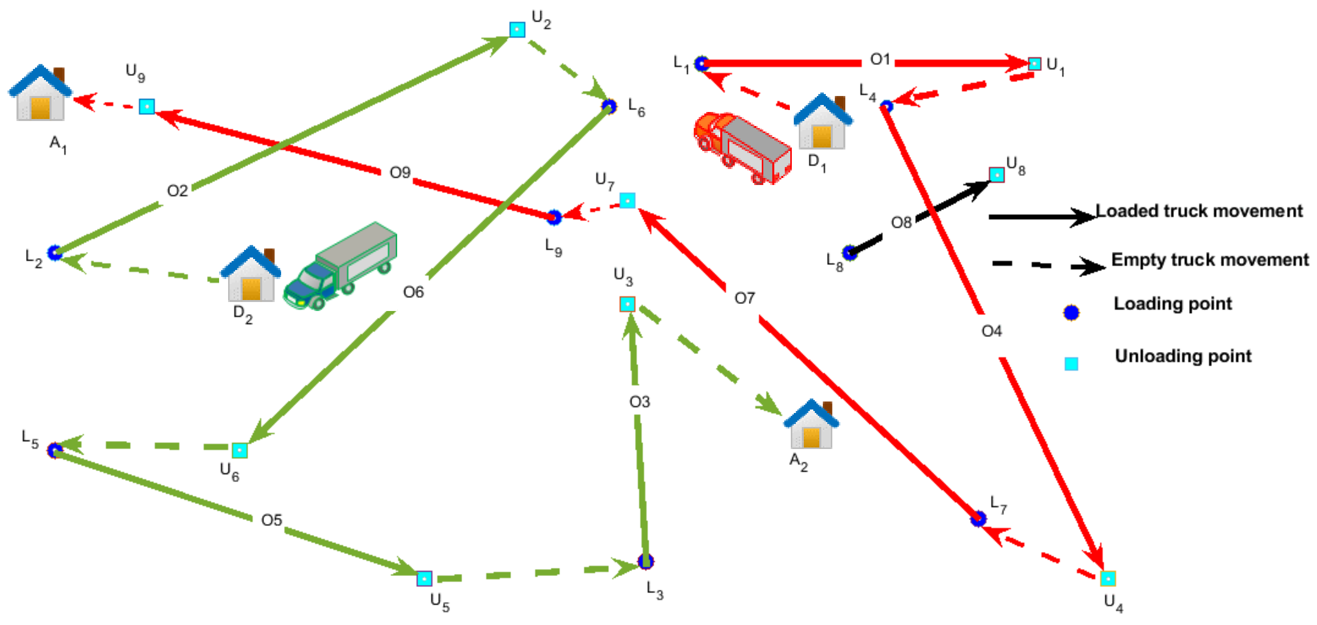


Figure 1. Example of the SFTMDVRPTW with nine orders and two trucks

3.2 Mathematical model

The proposed MILP model uses the following notations.

Indices and sets

k	truck index, $k = 1, \dots, m$, where m represents the number of available trucks
i, j, h	order index, $i, j, h = 1, \dots, n$, where n represents the number of orders
$K = \{1, \dots, m\}$	the set of all backhaul truks
$D = \{D_1, \dots, D_m\}$	the set of departure points of trucks
$A = \{A_1, \dots, A_m\}$	the set of arrival points of trucks
$O = \{O_1, \dots, O_n\}$	the set of all orders
$L = \{L_1 \dots, L_n\}$	the set of loading points of orders
$U = \{U_1 \dots, U_n\}$	the set of unloading points of orders

Parameters

D_k^{min}	earliest departure time of truck k
A_k^{max}	latest arrival time of truck k
p_i	revenue derived from moving order O_i
L_i^{min}	the earliest time to load the order O_i
L_i^{max}	the latest time to load the order O_i
U_i^{min}	the earliest time to unload the order O_i
U_i^{max}	the latest time to perform the unloading of the order O_i
c_i	loaded traveling cost to serve the order O_i
c^w	wait cost per unit time before an order's loading or unloading
c_{ij}	cost of empty travel between loading location of order O_i and unloading location of order O_j
$c_{0,i}^k$	cost of travel from the start depot D_k to the loading location L_i of order O_i
$c_{i,n+1}^k$	cost of travel from delivery location of order O_i to arriving depot A_k
t_i	processing duration time of order O_i
t_{ij}	traveling time from the delivery location of order O_i to the pickup location of order O_j
$t_{0,i}^k$	traveling time from the start depot D_k to the pickup location L_i of the order O_i
$t_{i,n+1}^k$	traveling time from the delivery location of order O_i to the arriving location A_k
M	a big number

Decision variables

x_{ij}^k	boolean variable $x_{ij}^k = 1$ if truck k travels from the unloading point of order O_i to the loading point of order O_j
x_{0i}^k	boolean variable $x_{0i}^k = 1$ If truck k serves order O_i as its first order
$x_{i,n+1}^k$	boolean variable $x_{i,n+1}^k = 1$ If truck k serves order O_i as its final order
$x_{0,n+1}^k$	boolean variable $x_{0,n+1}^k = 1$ If no order assigned to truck k

$t_{i,L}^{k}$ start time to load order O_i on the truck k
$t_{i,U}^{k}$ start time to unload order O_i of truck k
$a_{i,L}^{k}$ waiting time before loading order O_i on truck k
$t_{0,L}^{k}$ departure time of truck k from D_k
$t_{n+1,U}^{k}$ arriving time of truck k at A_k

The MILP model of the SFTMDVRPTW problem is given as follows:

$$\max \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n+1} p_i x_{ij}^k - \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n+1} c_i x_{ij}^k - \sum_{k=1}^{m}\sum_{j=1}^{n+1} c_{0,j}^{k} x_{0,j}^{k} - \sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij}^k - \sum_{k=1}^{m}\sum_{i=1}^{n} c_{i,n+1}^{k} x_{i,n+1}^{k} - c^w \sum_{k=1}^{m}\sum_{i=1}^{n} (a_{i,L}^k + t_{i,U}^k - t_{i,L}^k - t_i) \quad (1)$$

Subject to:

$$D_k^{\min} \le t_{0,L}^{k} \qquad \forall k = 1,...,m \quad (2)$$

$$t_{n+1,U}^{k} \le A_k^{\max} \qquad \forall k = 1,...,m \quad (3)$$

$$L_i^{\min} \le t_{i,L}^{k} \le L_i^{\max} \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (4)$$

$$U_i^{\min} * \sum_{j=1}^{n+1} x_{ij}^{k} \le t_{i,U}^{k} \le U_i^{\max} \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (5)$$

$$t_{i,L}^{k} + t_i \le t_{i,U}^{k} \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (6)$$

$$t_{0,L}^{k} + t_{0,i}^{k} + a_{i,L}^{k} \le t_{i,L}^{k} + \mathrm{M} * \left(1 - x_{0i}^{k}\right) \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (7)$$

$$t_{0,L}^{k} + t_{0,i}^{k} + a_{i,L}^{k} \ge t_{i,L}^{k} - \mathrm{M} * \left(1 - x_{0i}^{k}\right) \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (8)$$

$$t_{i,U}^{k} + t_{i,j} + a_{j,L}^{k} \le t_{j,L}^{k} + \mathrm{M} * \left(1 - x_{ij}^{k}\right) \qquad \forall i, j = 1,...,n,\ \forall k = 1,...,m \quad (9)$$

$$t_{i,U}^{k} + t_{i,j} + a_{j,L}^{k} \ge t_{j,L}^{k} - \mathrm{M} * \left(1 - x_{ij}^{k}\right) \qquad \forall i, j = 1,...,n,\ \forall k = 1,...,m \quad (10)$$

$$t_{i,U}^{k} + t_{i,n+1}^{k} \le t_{n+1,U}^{k} + \mathrm{M} * \left(1 - x_{i,n+1}^{k}\right) \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (11)$$

$$t_{i,U}^{k} + t_{i,n+1}^{k} \ge t_{n+1,U}^{k} - \mathrm{M} * \left(1 - x_{i,n+1}^{k}\right) \qquad \forall i = 1,...,n,\ \forall k = 1,...,m \quad (12)$$

$$\sum_{j=1}^{n+1}\sum_{k=1}^{m} x_{ij}^{k} \le 1 \qquad \forall i = 1,...,n \quad (13)$$

$$\sum_{i=0}^{n}\sum_{k=1}^{m} x_{ij}^{k} \le 1 \qquad \forall j = 1,...,n \quad (14)$$

$$\sum_{j=1}^{n+1} x_{0j}^k = 1 \qquad \forall k = 1,\dots,m \tag{15}$$

$$\sum_{i=0}^{n} x_{ih}^k - \sum_{j=1}^{n+1} x_{hj}^k = 0 \qquad \forall h = 1,\dots,n, \ \ \forall k = 1,\dots,m \tag{16}$$

$$\sum_{i=0}^{n} x_{i,n+1}^k = 1 \qquad \forall k = 1,\dots,m \tag{17}$$

$$x_{i,0}^k = 0 \qquad \forall k = 1,..,m, \ \ \forall i = 0,\dots,n+1 \tag{18}$$

$$x_{n+1,i}^k = 0 \qquad \forall k = 1,..,m, \ \ \forall i = 0,\dots,n+1 \tag{19}$$

$$x_{i,i}^k = 0 \qquad \forall k = 1,..,m, \ \ \forall i = 0,\dots,n+1 \tag{20}$$

$$x_{i,j}^k + x_{j,i}^k \le 1 \qquad \forall i, j = 0,\dots,n+1, \qquad \forall k = 1,..,m, \tag{21}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall i, j = 0,\dots,n+1, \ \ \forall k = 1,\dots,m \tag{22}$$

$$t_{i,L}^k \ge 0 \qquad \forall i = 0,\dots,n, \ \ \forall k = 1,\dots,m \tag{23}$$

$$t_{i,U}^k \ge 0 \qquad \forall i = 1,\dots,n+1, \ \ \forall k = 1,\dots,m \tag{24}$$

$$a_{i,L}^k \ge 0 \qquad \forall i = 1,\dots,n, \ \ \forall k = 1,\dots,m \tag{25}$$

The objective function (1) aims at maximizing the total profit, which is equal to total revenue from the selected orders (first term) minus the overall transportation cost of the routes. The transportation costs are composed of the cost of moving loaded trucks (second term), empty cost (third, fourth and fifth terms), and cost of waiting (sixth term).

Constraints (2)-(12) are related to time windows and guarantee the feasibility of the schedule. Constraints (2) ensure that each truck never departs from the start depot before it opens. Constraints (3) and (4) indicate that the time window at each loading and unloading location are met. Constraints (5) ensure that each truck never enters the arriving depot after it closes. Constraints (6) impose that, at the level of an order, the unloading time has to be more than the summation of the loading time and the service time of the order. This also implies that the waiting time $t_{i,U}^k - t_{i,L}^k - t_i$ of truck k before unloading the order i is a positive integer. Constraints (7) and (8) require that loading of the order O_i on truck k may not start before moving the truck to the start depot D_k. Constraints (9) and (10) stipulate that if $x_{ij}^k = 1$, then $t_{j,L}^k = t_{i,U}^k + t_{i,j} + a_{j,L}^k$. This means that between two successive orders, the picking of the next one can't start unless the previous order is delivered and the displacement has happened. Constraints (11) and (12) require that the truck k can only deliver the transportation demand O_i when it is likely to reach the arriving depot before the latest time. Constraints (13) and (14) necessitate that each time a truck k enters a pickup or delivery location, only one exit to any other location (pickup, deliver or arriving locations) can be performed. Thus, each loading and unloading location can be visited by a single truck at most once. Constraints (15)-(17) specify the flow conservation constraints. Constraints (15) indicate that each truck could depart from one starting depot. Constraints (16) make sure that once a truck k loads a merchandise, it must reach and leave the unloading location in the next step. Constraints (17) force each truck to go to the arriving depot. Constraints (18) and (19) entail that no vehicle can return to its start depot, and the arriving depot is the last point of each truck route. Constraints (20) eliminate cycles at the orders. Constraints (21) state that either O_i precedes O_j or vice versa or orders i and j are not into the same tour. Finally, constraints (22)-(25) define the decision variables.

4. Solution procedure

The SFTMDVRPTW is NP-hard because it includes the VRP which is NP-hard (Lenstra et al., 1981). Hence, utilizing a heuristic or metaheuristic algorithm will become necessary for obtaining high-quality solutions in reduced computation time. Genetic algorithm (GA) belongs to the family of approximate approaches inspired by Darwin's evolution and natural selection theory. The pioneer in the field Holland (1975) was the first to implant this evolutionary computing artificially. Afterward, it has successfully evolved into robust optimization approaches for various VRPs (Baker and Ayechew, 2003; Berger and Barkaoui, 2004; Ho et al., 2008). Among the most important mechanisms underlying evolution are evolution occurs on chromosomes that represent each individual in a population; the process of natural selection results in the best-adapted chromosomes reproducing more often and contributing more to future populations; the information in the parents' chromosomes is combined and mixed to produce the child chromosomes (crossover) during reproduction; the result of the crossover can, in turn, be modified by random perturbations (mutation).

To adapt a GA to our problem, it is necessary to specialize certain components to the latter's particular structure, including an encoding method for solutions into individuals (chromosomes), parameters, a mechanism for generating an initial population, a fitness function to evaluate the solutions, a mechanism for selecting parents during reproduction, a crossover operator to produce children from selected parents, a mutation operator that can modify some individuals of the population, a stopping criterion. The program flowchart of the solution procedure is shown in Figure 2.

The following definitions are required in our GA:

$Npop$	Population size
G	The maximal number of generations
P_c	Crossover rate
P_m	Mutation rate

4.1 Chromosome representation

Our problem has two parts: the first part is to find a selection of orders for truck routes. The second is finding an optimal sequence of orders for each backhaul truck to be served. Therefore, we opt for a two-part chromosome technique to define a chromosome, which is adapted from Carter and Ragsdale (2006). As the name indicates, this encoding method divides the chromosome into two parts. The first part of length n represents a permutation of n orders, which determines the sequence of orders that each truck served. The second part comprises $m + 1$ non-negative integers of sum n, in which the first m genes refer to the number of orders assigned to each truck, and the final gene provides the number of unselected orders assigned to a dummy truck, denoted as U. Therefore, the total length of the chromosome is $n + m + 1$ in this representation. Figure 3 gives an example of a chromosome representation consisting of 16 orders and three trucks. (O_2 , O_4 , O_7 , O_{12}), (O_{14}, O_8 , O_{10} , O_6 , O_5), and (O_{11} , O_{15} , O_{16} , O_3) are respectively the sequences of orders served by truck 1, truck 2, and truck 3, while the last sequence (O_{13} , O_9 , O_1) consists of the unselected orders.

4.2 Initial population

To build an initial population of $Npop$ chromosomes, we will develop a heuristic that progressively constructs each chromosome. Each chromosome must at least be a feasible candidate solution, i.e., all truck routes in the chromosome must respect the constraints of earliest departure and latest arrival time of trucks and loading and unloading time windows of served orders. The initialization process is started by randomly classifying the trucks. Then randomly select an order and inserting it into a truck route according to the ordering of trucks, which when the order violates the constraints, it is deleted from the current route and assigned to the next one. If it is not possible to assign such an order to all trucks, the order is placed in the dummy truck. This process continues until all orders are routed, and a feasible initial population is built as shown in Table 2. Once a truck route $T_k = \langle D_k,\ O_{i_1}, \ldots, O_{i_{n_k}},\ A_k \rangle$ is generated, it is often possible to delay the departure time of truck k from the depot D_k without violating the loading and unloading time windows constraints of orders, which can reduce the total waiting time on route T_k. To do so, an algorithm is designed as shown in Table 3, which is named Algorithm 1. Having identified a waiting time at the loading and/or unloading point of an order O_{i_j}, $j \in \{1, \ldots, n_k\}$, served in route T_k, the proposed algorithm delayed the departure time of the truck k and then updated the loading and unloading times of the previous orders.After generating the initial population, we construct the corresponding two-part chromosome for each solution as shown in Figure 4.

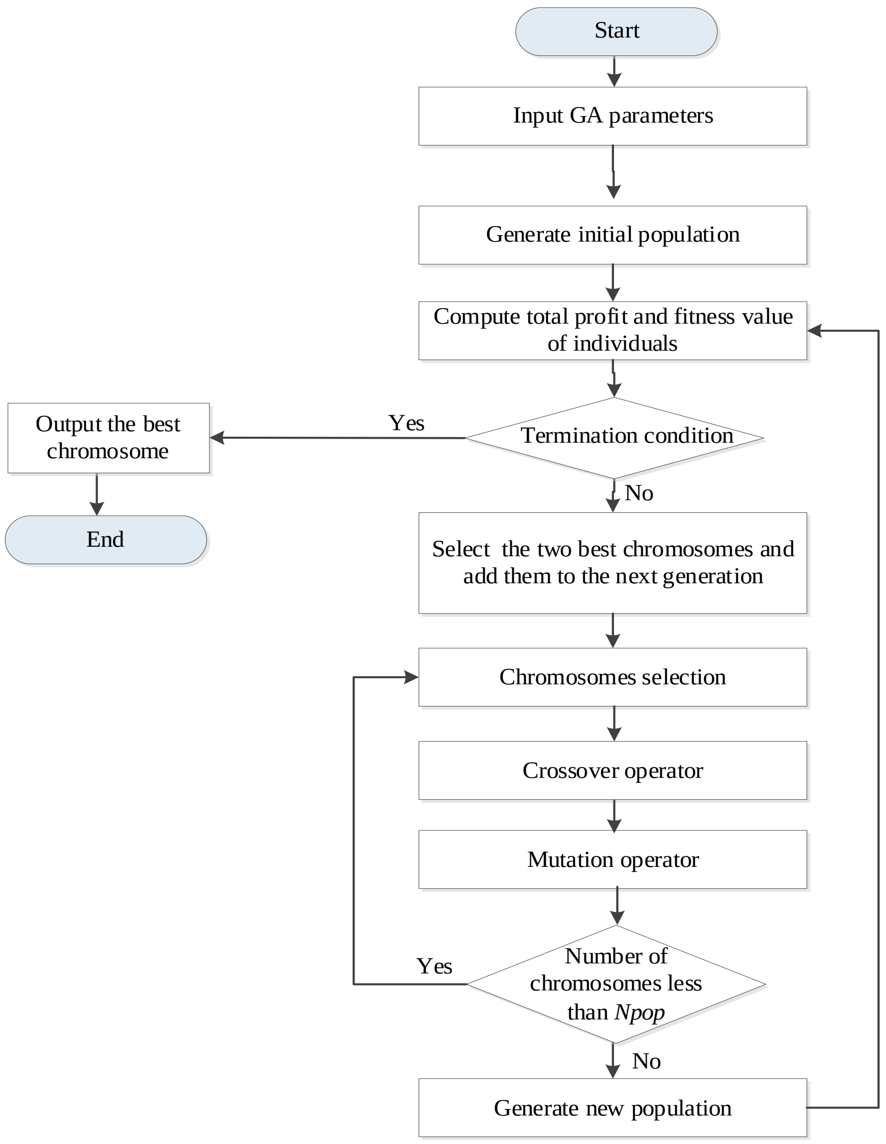

Start
Input GA parameters
Generate initial population
Compute total profit and fitness value of individuals
Termination condition
Yes
No
Output the best chromosome
End
Select the two best chromosomes and add them to the next generation
Chromosomes selection
Crossover operator
Mutation operator
Number of chromosomes less than Npop
Yes
No
Generate new population

Figure 2. Flowchart of the solution procedure

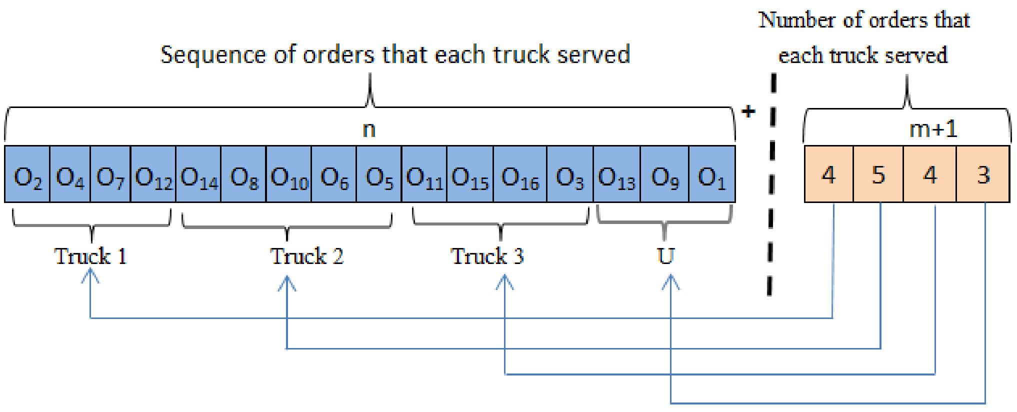

Sequence of orders that each truck served
Number of orders that each truck served
n
m+1
O2 O4 O7 O12 O14 O8 O10 O6 O5 O11 O15 O16 O3 O13 O9 O1
4 5 4 3
+
Truck 1
Truck 2
Truck 3
U

Figure 3. An example of two-part chromosome representation

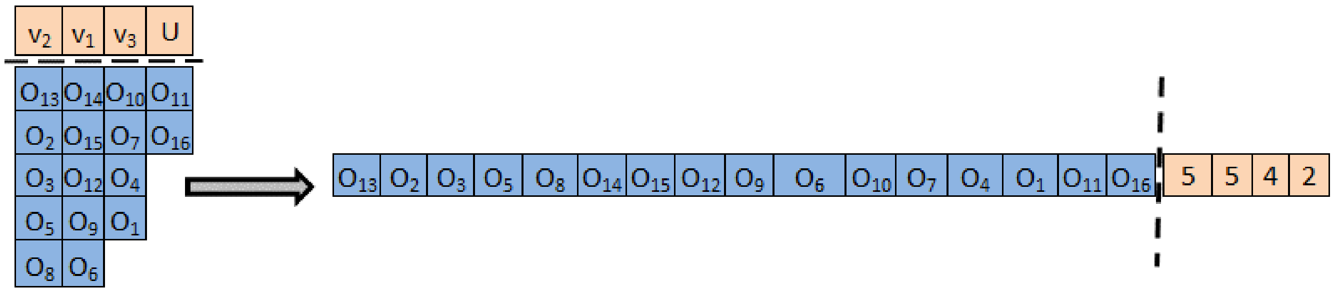

V2 V1 V3 U
O13 O14 O10 O11
O2 O15 O7 O16
O3 O12 O4
O5 O9 O1
O8 O6
O13 O2 O3 O5 O8 O14 O15 O12 O9 O6 O10 O7 O4 O1 O11 O16 5 5 4 2

Figure 4. From the solution to the two-part chromosome encoding

Table 2. Pseudo-code of the heuristic of the initial population creation

```
The heuristic for generating the initial population
1. for $i = 1$ to $Npop$ do
2.  $T_k = \langle D_k, A_k \rangle$, $U = \emptyset$, $t_{0,L}^k = D_k^{min}$, $t_{n+1,U}^k = D_k^{min} + t_{0,n+1}^k$
3.  Create the candidate list containing all orders: CL = $\{O_1, \ldots, O_n\}$
4.  Randomly classify the trucks
5.  while CL $\neq \emptyset$ do
6.    Randomly select a non-assigned order $O_j \in$ CL, and assign them
      according to the ordering of the trucks
7.    if it is possible to assign such an order $O_j$ to a truck $k$ then
8.      $T_k = \langle D_k, \ldots, O_j, A_k \rangle$
9.      if truck $k$ serves order $O_j$ as its first order then
10.       $t_{0,L}^k = D_k^{min} + max(0, L_j^{min} - D_k^{min} - t_{0,j}^k)$
11.       $t_{j,L}^k = t_{0,L}^k + t_{0,j}^k$
12.       $a_{j,L}^k = 0$
13.     else if order $O_j$ is served immediately after order $O_i$ by truck $k$ then
14.       $t_{j,L}^k = t_{i,U}^k + t_{ij} + max(0, L_j^{min} - t_{i,U}^k - t_{ij})$
15.       $a_{j,L}^k = max(0, L_j^{min} - t_{i,U}^k - t_{ij})$
16.     end if
17.     $t_{j,U}^k = t_{j,L}^k + t_j + max(0, U_j^{min} - t_{j,L}^k - t_j)$
18.   else
19.     $U = U \cup \{O_j\}$
20.   end if
21.   CL = CL \ $\{O_j\}$
22. end while
23. for k= 1 to m do
24.   if $T_k \neq \langle D_k, A_k \rangle$// there is at least one order served on route $T_k$
25.     Use the algorithm 1 to calculate the minimum waiting time and update the
        departure and arrival time of truck k
26.   end if
27. end for
28. end for
```

Table 3. Pseudocode of Algorithm 1

```
Algorithm 1
input: Route $T_k = \langle D_k, O_{i_1}, \ldots, O_{i_{n_k}}, A_k \rangle$, where $\{i_1, \ldots, i_{n_k}\} \subset \{1, \ldots, n\}$
output: Minimum waiting time before loading/unloading order $O_{i_j}$, $i_j = 1, \ldots, n_k$
1. $a_{i_1,U}^k = t_{i_1,U}^k - t_{i_1,L}^k - t_{i_1}$ // Waiting time before unloading order $O_{i_1}$ on truck k
2. if $(a_{i_1,U}^k \neq 0)$ then
3.  $\Delta U(i_1, k) = max\left(0, t_{i_1,L}^k + a_{i_1,U}^k - L_{i_1}^{max}\right)$
4.  $t_{0,L}^k = t_{0,L}^k + a_{i_1,U}^k - \Delta U(i_1, k)$
5.  $t_{i_1,L}^k = t_{i_1,L}^k + a_{i_1,U}^k - \Delta U(i_1, k)$
6. end if
7. if $(i_{n_k} > 1)$ then
8.  $j = 2$
9.  while $j \leq n_k$ and $(\forall h \in \{1, \ldots, j-1\}$, $t_{i_h,L}^k < L_{i_h}^{max}$ and $t_{i_h,U}^k < U_{i_h}^{max})$ do
10.   if $(a_{i_j,L}^k \neq 0)$ and then
11.   $\Delta L(i_j, k) = max\left(0, max\left\{t_{i_h,L}^k + a_{i_j,L}^k - L_{i_h}^{max} / 1 \leq h \leq j-1\right\} \cup \left\{t_{i_h,U}^k + a_{i_j,L}^k - U_{i_h}^{max} / 1 \leq h \leq j-1\right\}\right)$
```

```
12.     $t_{0,L}^{k} = t_{0,L}^{k} + a_{i_j,L}^{k} - \Delta L(i_j,k)$
13.     for $h$ = 1 to j – 1 do
14.       $t_{i_h,L}^{k} = t_{i_h,L}^{k} + a_{i_j,L}^{k} - \Delta L(i_j,k)$
15.       $t_{i_h,U}^{k} = t_{i_h,U}^{k} + a_{i_j,L}^{k} - \Delta L(i_j,k)$
16.     end for
17.     $a_{i_j,L}^{k} = \Delta L(i_j,k)$
18.   end if
19.   $a_{i_j,U}^{k} = t_{i_j,U}^{k} - t_{i_j,L}^{k} - t_{i_j}$ // Waiting time before unloading order $O_{i_j}$ on truck k
20.   if ($a_{i_j,U}^{k} \neq 0$ and $t_{i_j,L}^{k} < L_{i_j}^{max}$) and ($\forall h \in \{1, \ldots, j-1\}$, $t_{i_h,L}^{k} < L_{i_h}^{max}$ and $t_{i_h,U}^{k} < U_{i_h}^{max}$) then
21.     $\Delta U(i_j,k) = max\left(0, max\left\{t_{i_h,L}^{k} + a_{i_j,U}^{k} - L_{i_h}^{max} / 1 \le h \le j\right\} \cup \left\{t_{i_h,U}^{k} + a_{i_j,U}^{k} - U_{i_h}^{max} / 1 \le h \le j-1\right\}\right)$
22.     $t_{0,L}^{k} = t_{0,L}^{k} + a_{i_j,U}^{k} - \Delta U(i_j,k)$
23.     $t_{i_j,L}^{k} = t_{i_j,L}^{k} + a_{i_j,U}^{k} - \Delta U(i_j,k)$
24.     for $h$ = 1 to j – 1 do
25.       $t_{i_h,L}^{k} = t_{i_h,L}^{k} + a_{i_j,U}^{k} - \Delta U(i_j,k)$
26.       $t_{i_h,U}^{k} = t_{i_h,U}^{k} + a_{i_j,U}^{k} - \Delta U(i_j,k)$
27.     end for
28.   end if
29.   $j$ = j + 1
30.  end while
31. end if
32. $t_{n+1,U}^{k} = t_{i_{n_k},U}^{k} + t_{i_{n_k},n+1}^{k}$
```

4.3 Fitness function and chromosome selection

The fitness value provides the quality of an individual in the population. Accordingly, the chances of the selection of this individual are calculated. This paper adopts an approach based on elitism strategy and roulette wheel selection to make a new generation. The top two condidate individuals from each generation are selected to directly move on to the next generation, unaltered. Then, we use the roulette wheel selection to choose two individuals as parents to make two children and then repeat this process until $Npop$ chromosomes are generated for the next generation. Permitting meta-heuristics to admit infeasible solutions often provides a better method to efficiently give higher-quality solutions (Cordeau et al., 2004). Therefore, we allow infeasible solutions during the process of our GA by relaxing constraints on the latest arrival time for trucks and order loading and unloading time windows.

Given an individual p, let $f(p)$ denote the objective function value of p, the new objective function is $f'(p) = f(p) - M \times \sum_{k=1}^{m}\left(\sum_{i=1}^{n} max\left(0, t_{i,L}^{k} - L_{i}^{max}\right) + \sum_{i=1}^{n} max\left(0, t_{i,U}^{k} - U_{i}^{max}\right) + max\left(0, t_{n+1,U}^{k} - A_{k}^{max}\right)\right)$, which decreases the survival chance of infeasible individuals. The fitness value is needed to be non-negative, and a higher fitness value designates a better individual. Thus, the fitness value $F(p)$ of an individual p takes the following form:

$$\begin{cases} F(p) = 1 + f'(p); \;\; f'(p) > 0 \\ F(p) = \dfrac{1}{1 - f'(p)}; \;\; f'(p) \le 0 \end{cases}$$

4.4 Crossover operator

In this section, we propose a new crossover operator denoted S-TCX for the GA. The S-TCX crossover treats each truck route individually when executing the crossover operator in the first part of the chromosome. Besides, The S-TCX reproduces genes in the second part of the chromosome according to the crossover operation results performed in the first part of the chromosome. Figure 5 illustrates the process of using the S-TCX in 5 steps to generate a new child chromosome of 10 transportation demands and three trucks.

First, we select a pair of two-part chromosomes as parents (P_1 and P_2); in this example, P_1 is used as the base chromosome to produce a child C_1. Second, we randomly choose substrings from the first part of parent P_1. In this example, the selected segments are (O_4 , O_7) for truck 1, (O_{14} , O_8 , O_{10}) for truck 2, (O_{11}, O_{15} , O_2) for truck 3 and (O_1) for the dummy truck U. Therefore, the numbers of the randomly selected genes are $(2, 3, 3 , 1)$ as shown in step 2. Next, the remaining genes are sorted according to their corresponding positions in the first part of P_2's chromosome. In this example, the remaining genes in the first part of P_1's chromosome are (O_{16} , O_{12} , O_6, O_5 , O_3 , O_{13} , O_9), and they are rearranged to (O_{13}, O_3 , O_5, O_{12} , O_9 , O_6 , O_{16}) according to the first part of P_2's chromosome. Afterwards, based on the number of unselected genes, we calculate a uniform random number, between 1 and the current value of unselected genes, to determine how many new genes will be added for each truck route in the offspring chromosome C_1. Here, the unselected genes are (O_{13}, O_3 , O_5, O_{12} , O_9 , O_6 , O_{16}) and if, for example, we iteratively generate the sequence of integers $(2, 2, 2 , 1)$, truck 1 receives genes (O_{13}, O_3), truck 2 receives (O_5, O_{12}), truck 2 receives (O_9 , O_6) and U receives (O_{16}). Therefore, for the first part of child C_1, v_1 would have (O_4, O_7, O_{13}, O_3), v_2 would have $(O_{14}, O_8, O_{10}, O_5, O_{12})$, v_3 would have $(O_{11}, O_{15}, O_2, O_9, O_6)$ and U would have (O_1, O_{16}). Finally, we build the two-part chromosome for the child C_1 by updating the information contained in its second part. In this example, the vector (4, 3, 3) is generated through the sum of the two intermediate vectors $(2, 3, 3, 1)$ and $(2, 2, 2, 1)$: $(2, 3, 3, 1) + (2, 3, 3, 1) = (4, 5, 5, 2)$.

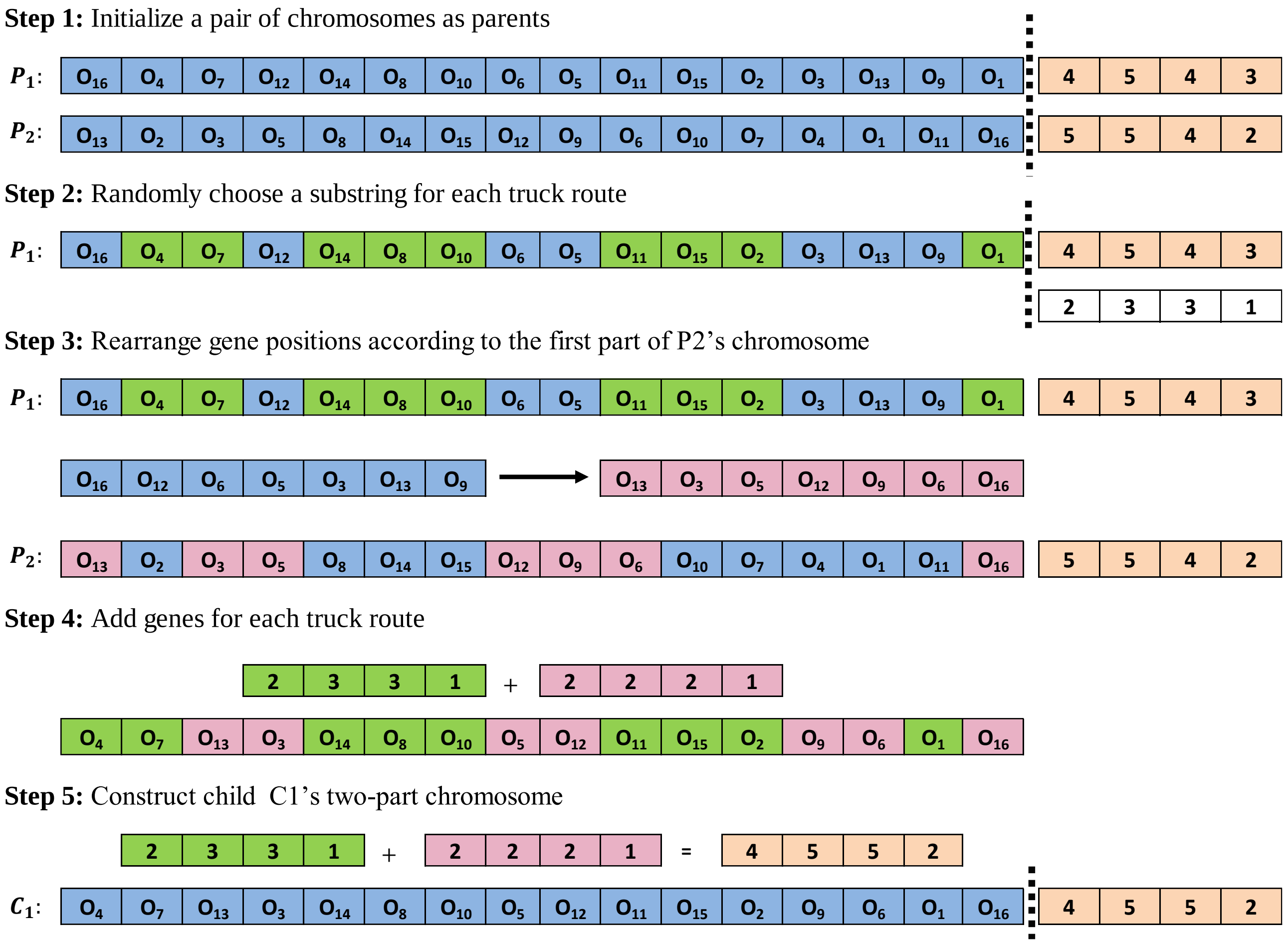

Step 1: Initialize a pair of chromosomes as parents
P_1: O16 O4 O7 O12 O14 O8 O10 O6 O5 O11 O15 O2 O3 O13 O9 O1 | 4 5 4 3
P_2: O13 O2 O3 O5 O8 O14 O15 O12 O9 O6 O10 O7 O4 O1 O11 O16 | 5 5 4 2
Step 2: Randomly choose a substring for each truck route
P_1: O16 O4 O7 O12 O14 O8 O10 O6 O5 O11 O15 O2 O3 O13 O9 O1 | 4 5 4 3
2 3 3 1
Step 3: Rearrange gene positions according to the first part of P2's chromosome
P_1: O16 O4 O7 O12 O14 O8 O10 O6 O5 O11 O15 O2 O3 O13 O9 O1 | 4 5 4 3
O16 O12 O6 O5 O3 O13 O9
O13 O3 O5 O12 O9 O6 O16
P_2: O13 O2 O3 O5 O8 O14 O15 O12 O9 O6 O10 O7 O4 O1 O11 O16 | 5 5 4 2
Step 4: Add genes for each truck route
2 3 3 1 + 2 2 2 1
O4 O7 O13 O3 O14 O8 O10 O5 O12 O11 O15 O2 O9 O6 O1 O16
Step 5: Construct child C1's two-part chromosome
2 3 3 1 + 2 2 2 1 = 4 5 5 2
C_1: O4 O7 O13 O3 O14 O8 O10 O5 O12 O11 O15 O2 O9 O6 O1 O16 | 4 5 5 2

Figure 5. Example of the S-TCX crossover

Subsequently, we use the P_2's chromosome as the base, and we perform the five steps explained above to obtain another child C_2. After each S-TCX crossover operation, we generate a pair of child chromosomes.

4.5 Mutation operator

The mutation operation slightly alters the composition of a child chromosome to diversify the population. Our GA's mutation operator is based on the swap mutation (Goldberg,1989; Carter and Ragsdale, 2006; Yuan et al., 2013), which consists of randomly swapping two genes in both parts of the two-part chromosome. The swap mutation operation is applied to the first-part and second one of the chromosome through the mutation operator's independent application (see Figure 6).

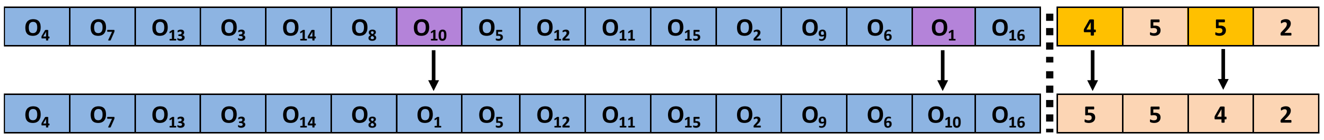

O4 O7 O13 O3 O14 O8 O10 O5 O12 O11 O15 O2 O9 O6 O1 O16 | 4 5 5 2
O4 O7 O13 O3 O14 O8 O1 O5 O12 O11 O15 O2 O9 O6 O10 O16 | 5 5 4 2

Figure 6. Example of the swap mutation

4.6 Stopping criterion

The latest GA step is to verify whether the current near-optimal solution is good enough to satisfy our expectations. When the termination condition is met, the algorithm breaks, and the current best individual is the final solution to the problem. In this research, GA ends after G generations. Note that G's value depends on the instance size.

5. Experimental tests

This section introduces and discusses the computational findings of our mathematical model and GA on newly generated instances. First, we describe the new problem instances used to evaluate the proposed solution approaches. Afterward~~s~~, we illustrate the computational findings on the generated instances.

5.1 Generation of instances and parameter setting

We have generated new SFTMDVRPTW instances in analogy to real-life data based on Solomon's VRPTW benchmark datasets[*]. The latter is divided into three instance categories: R, C, RC. They comprise one depot and 100 points with given coordinates in a $[0, 100]^2$ square. We designed 38 SFTMDVRPTW instances partitioned in small size ($n \in \{16, 20\}$), medium-size ($n \in \{24, 30\}$) and large-size instances ($n \in \{50, 75\}$). There are between 2 and 7 trucks. First, we generate the coordinates of all points. After that, to generate a set of orders, we randomly decide whether a point is regarded as a loading or unloading point. To generate trucks' departure and arrival points, we divide the $[0, 100]^2$ square into four quarters. We randomly located the departure point in a quarter and the arrival point in another one so that the distance between them is greater than 60. Traveling times and costs (loaded/empty) are calculated using the Euclidean distance function; the unitary service revenue p of each order is equal to \$6/mile; the unitary operational cost c (loaded/empty) is equal to \$1/mile, the average truck speed S_T (loaded/empty) has been set to 1 mile/min (or 60 mile/hour) for all trucks, the processing duration time t_i of an order O_i is taken as the sum of the service time to load at the point L_i, the service time to unload at the point U_i, and the loaded travel time between L_i and U_i, the width of the time windows, denoted by $widthtw$, ranges from 2 to 4h.

We set the format of each instance name as follows: $SFTx\_Sourcexx\_n\_m$. Consider the instance $SFT1\_R25\_20\_2$ as an example. There are 20 transportation demands and two trucks. The code $SFT1$ refers to the instance's number. The code $R25$ means this instance is created by just using the 25 first points of the original one: category R. Table 4 represents brief instance information; it specifies the number of orders, number of trucks, service time s for loading/unloading an order, earliest departure and latest arrival times permitted for each truck, $widthtw$, and detailed values of all parameters of the problem. These newly generated test instances can be downloaded via link[†]. Parameters in GA are as follows: $Npop = 200, P_c = 0.9, P_m = 0.1, G = 500$ (for small-sized instances), $G = 2000$ (for medium- and large-sized instances). In the MILP formulation, the constant M is set to 100000.

5.2 Evaluation of GA

The proposed GA is implemented in the C++ language and run on a PC with Intel(R) Core(TM) i5 Processor (2.4 GHz) and 4 GB RAM. The MILP model was implemented in OPL and solved by CPLEX 12.10 to obtain the best feasible solution (the lower bound) and the upper bound. A comparison between GA and CPLEX will be evaluated on computation effectiveness and efficiency. For each instance, the CPLEX is run for 2 hour-time limit, and the GA is run 10 times, and the best objective value is reported.

In Table 5 we present the results obtained from CPLEX and GA. The first column is the instances' names. The second and third columns are the lower bound LB and and the upper bound UB provided by CPLEX. The fourth column (Gap1) is the gap between LB and UB, which is calculated as: $100 \times (UB - LB) / UB$. The fifth and ninth columns are the CPU time of CPLEX and GA, respectively. The sixth and tenth columns show the number of orders not selected in the solutions found in CPLEX and GA, respectively. The seventh column is the best objective found in GA. The solutions in bold are as good as the optimal solutions obtained from CPLEX. The eighth column (Gap2) is the gap between the best objective and the best integer, calculated by $100 \times (UB - Best\ objective) / UB$.

Table 4. The parameter structure for the generated instances

Instance ID	n	m	s (min)	D_k^{min} (min)	A_k^{max} (min)	$widthtw$ (min)	Parameters	
							Notation	Value
$SFT1-2\_C25\_16\_2$	16	2	30	{0,60}	600	180	$d(.,.)$ ($mile$)	Euclidean distance between any two points
$SFT3-5\_C25\_16\_2$	16	2	40	{0,60}	720	240	p_i (\$)	$p \times d(L_i, U_i)$
$SFT1-2\_C50\_24\_3$	24	3	30	{0,60,120}	600	180	c_i (\$)	$c \times d(L_i, U_i)$

[*] website: http://web.cba.neu.edu/~msolomon/problems.htm

[†] (https://drive.google.com/file/d/1SR5Jk3G4hwwlpe-SCeZwetKmF0558PnJ/view?usp=sharing)

$SFT3-5\_C50\_24\_3$	24	3	40	{0,60,120}	720	240	c^w ($\$/min$)	0.42
$SFT1-2\_R25\_20\_2$	20	2	10	{0,60}	480	120	c_{ij} ($)	$c \times d(U_i, L_j)$
$SFT3-5\_R25\_20\_2$	20	2	20	{0,60}	720	180	$c_{0,i}^k$ ($)	$c \times d(D_k, L_i)$
$SFT1-2\_R50\_30\_3$	30	3	10	{0,60,120}	480	120	$c_{i,n+1}^k$ ($)	$c \times d(U_i, A_k)$
$SFT3-5\_R50\_30\_3$	30	3	20	{0,60,120}	720	180	t_i (min)	$d(L_i, U_i) / S_T + 2 * s$
$SFT1-2\_RC25\_20\_2$	20	2	10	{0,60}	480	120	t_{ij} (min)	$d(U_i, L_j) / S_T$
$SFT3-5\_RC25\_20\_2$	20	2	20	{0,60}	720	180	$t_{0,i}^k$ (min)	$d(D_k, L_i) / S_T$
$SFT1-2\_RC50\_30\_3$	30	3	10	{0,60,120}	480	120	$t_{i,n+1}^k$ (min)	$d(U_i, A_k) / S_T$
$SFT3-5\_RC50\_30\_3$	30	3	20	{0,60,120}	720	180		
$SFT1-2\_R100\_50\_5$	50	5	10	$0:30:120$	480	120	L_i^{min} (min)	$\min_k\{t_{0,i}^k\} + (A_k^{max} - \min_k\{t_{0,i}^k\} - t_i - \max_k\{t_{i,n+1}^k\}) \times U(0,1)$
$SFT3-4\_R100\_50\_5$	50	5	20	$0:30:120$	720	180	L_i^{max} (min)	$L_i^{min} + widthtw$
$FT1-2\_R100\_75\_7$	75	7	10	$0:30:180$	480	120	U_i^{min} (min)	$L_i^{min} + t_i$
$FT3-4\_R100\_75\_7$	75	7	20	$0:30:180$	720	180	U_i^{max} (min)	$U_i^{min} + widthtw$

5.2.1 Experiments on small and medium size instances

As observed on Table 5, the Gap1 and Gap2 for 28 out of 30 small- and medium-sized instances equals zero. This means that both the CPLEX solver and our GA found optimal solutions. The average running time of CPLEX and GA for all instances is about 682 s and 65 s, respectively. Therefore, our GA algorithm supersedes CPLEX in terms of running time, the longest CPU time being only 135.45 s compared with the limit of 7200 s reached by CPLEX. Figures 7-9 illustrate a comparison between CPU time and the number of unselected orders in the solutions provided by CPLEX and GA for SFT instances of types C, R and RC, respectively.

For the C-type instances with 16 orders, when $(widthtw, A_k^{max}) = (180, 600)$ $(SFT1-2\_C25\_16\_2)$ and $(widthtw, A_k^{max}) = (240, 720)$ $(SFTL3-5\_C25\_16\_2)$, the CPU times of both solution approaches are approximately the same, which doesn't exceed 25 s, and the number of unselected orders is the same in the two cases. When the number of orders increases to 24, the same can be said when $(widthtw, A_k^{max}) = (240, 720)$ $(SFT3-5\_C50\_24\_3)$ with a slight difference in CPU time that doesn't exceed 67 s, and the number of unselected orders is the same, that is 3. For the case $(widthtw, A_k^{max}) = (180, 600)$, the CPU time of CPLEX grows from 4.72 s when the number of unselected orders is 1 $(SFT1\_C50\_24\_3)$ to 717.92 s when the number of unselected orders is 5 $(SFT2_C50_24_3)$while GA has consumed nearly the same CPU time (around 1 min) to solve these instances to optimality.

Table 5. Comparison of results between GA and CPLEX

Data Set	CPLEX					GA			
	LB	UB	Gap1 (%)	CPU (s)	# U	Best objective	Gap2 (%)	CPU (s)	# U
$SFT1\_C25\_16\_2$	520	520	0.00	23.88	1	**520**[1]	0.00	18.16	1
$SFT2\_C25\_16\_2$	810	810	0.00	2.29	1	**810**	0.00	23.06	1
$SFT3\_C25\_16\_2$	819	819	0.00	5.44	2	**819**	0.00	22.52	2
$SFT4\_C25\_16\_2$	766	766	0.00	4.34	2	**766**	0.00	24.05	2
$SFT5\_C25\_16\_2$	828	828	0.00	4.01	2	**828**	0.00	24.07	2
$SFT1\_C50\_24\_3$	894	894	0.00	4.72	1	**894**	0.00	57.64	1
$SFT2\_C50\_24\_3$	1100	1100	0.00	717.92	5	**1100**	0.00	54.25	5
$SFT3\_C50\_24\_3$	1067	1067	0.00	66.08	3	**1067**	0.00	56.39	3
$SFT4\_C50\_24\_3$	1215	1215	0.00	14.07	3	**1215**	0.00	57.69	3
$SFT5\_C50\_24\_3$	1375	1375	0.00	66.18	3	**1375**	0.00	54.80	3
$SFT1\_R25\_20\_2$	2130	2130	0.00	53.88	2	**2130**	0.00	35.58	2
$SFT2\_R25\_20\_2$	1346	1346	0.00	2.05	0	**1346**	0.00	34.13	0
$SFT3\_R25\_20\_2$	2331	2331	0.00	3.88	1	**2331**	0.00	35.46	1
$SFT4\_R25\_20\_2$	2176	2176	0.00	2.76	1	**2176**	0.00	29.23	1
$SFT5\_R25\_20\_2$	2488.2	2488.2	0.00	2.38	1	**2488.2**	0.00	28.14	1
$SFT1\_R50\_30\_3$	3120	3367.15	7.34	7200[2]	**4**	**3120**	7.3 4	120.00	4
$SFT2\_R50\_30\_3$	2346	2346	0.00	3.25	0	**2346**	0.00	131.02	0
$SFT3\_R50\_30\_3$	3494	3494	0.00	855.70	1	**3494**	0.00	133.21	1
$SFT4\_R50\_30\_3$	3398	3398	0.00	490.12	1	**3398**	0.00	134.46	1

$SFT5\_R50\_30\_3$	3742	3795.92	1.42	7200	3	**3742**	1.42	135.45	3
$SFT1\_RC25\_20\_2$	1857	1857	0.00	2.05	1	**1857**	0.00	28.46	1
$SFT2\_RC25\_20\_2$	1538	1538	0.00	1.80	0	**1538**	0.00	34.21	0
$SFT3\_RC25\_20\_2$	2246	2246	0.00	3.70	1	**2246**	0.00	34.43	1
$SFT4\_RC25\_20\_2$	1859	1859	0.00	3.02	0	**1859**	0.00	35.79	0
$SFT5\_RC25\_20\_2$	2458	2458	0.00	12.42	2	**2458**	0.00	33.68	2
$SFT1\_RC50\_30\_3$	2882	2882	0.00	66.97	2	**2882**	0.00	119.86	2
$SFT2\_RC50\_30\_3$	2889	2889	0.00	110.90	4	**2889**	0.00	118.22	4
$SFT3\_RC50\_30\_3$	4193	4193	0.00	1237.85	4	**4193**	0.00	117.25	4
$SFT4\_RC50\_30\_3$	3852	3852	0.00	10.03	4	**3852**	0.00	1١7.٢٢	4
$SFT5\_RC50\_30\_3$	4056	4056	0.00	2289.19	4	**4056**	0.00	125.65	4
$SFT1\_R100\_50\_5$	4586	4586	0.00	587,63	1	**4586**	0.00	318.29	1
$SFT2\_R100\_50\_5$	4412	4498,18	1.92	7200	1	**4428**	1.56	320.83	1
$SFT3\_R100\_50\_5$	4637	4637	0.00	88,91	0	**4637**	0.00	371.51	0
$SFT4\_R100\_50\_5$	4506	4580.19	1.62	7200	1	**4507**	1.59	357.80	1
$SFT1\_R100\_75\_7$	8656[a]	8959,82[a]	3,39	3451.89	5	8595	4.07	509.20	5
$SFT2\_R100\_75\_7$	8647[a]	8960,34[a]	3,50	2753.54	**5**	8559	4.48	490.93	5
$SFT3\_R100\_75\_7$	8726[a]	8983,60[a]	2,87	5375,05	**4**	8704	3.11	521.93	4
$SFT4\_R100\_75\_7$	8800[a]	8964,87[a]	1,84	6319,94	**3**	**8800**	1.84	523.58	3

[1] The solutions given by GA are as good as that obtained from CPLEX
[2] CPLEX is run for a 7200 s time limit
[a] The ‘‘out of memory’’ values.

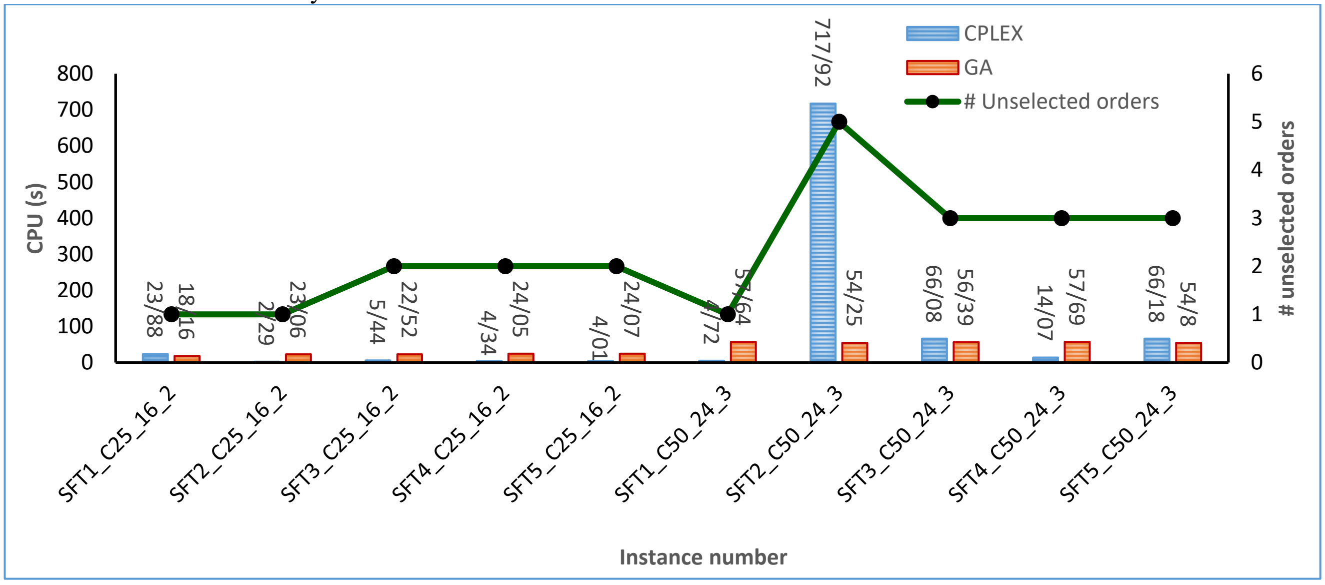


Figure 7. Computation time vs. Number of unselected orders for small and medium C-type instances

For the R-type instances with 20 orders, when $(widthtw, A_k^{max}) = (120, 480)$, we have two different instances, the model solves the instance $SFT2\_R25\_20\_2$ in a fast CPU time that is 2.05 s, in which all orders are selected. As to the instance $SFT1\_R25\_20\_2$, there are two unselected orders, and the CPU rises to 54 s. When $(widthtw, A_k^{max}) = (180, 720)$ $(SFT3 - 5\_R25\_20\_2)$, the CPU time and the number of unselected orders of CPLEX are the same in all instances, where the first doesn't exceed 4 s and the latter is 1. The CPU times consumed by GA to solve these five instances are nearly the same, which doesn't exceed 36 s, no matter the $(widthtw, A_k^{max})$ values. For $n = 30$, the instance $SFT2\_R50\_30\_3$ is solved optimally by CPLEX in less CPU time, and all orders are selected; it's a matter of bound. That is, in the first iterations, CPLEX finds the best bound. Maybe other parameters impact the quality of the solution and the CPU time of our model. The instances $SFT1\_R50\_30\_3$ and $SFT5\_R50\_30\_3$ are not solved optimally by CPLEX within 2 h in which the numbers of unselected orders in the LB_{MILP} are 4 and 3, respectively. Even though the number of unselected orders is 1 for $SFT3\_R50\_30\_3$ and $SFT4\_R50\_30\_3$, CPU time explodes to some extent; perhaps the FTL instances of class R are the most difficult ones to be solved by CPLEX. In contrast , the GA stopped within 131 s for each of the five instances.

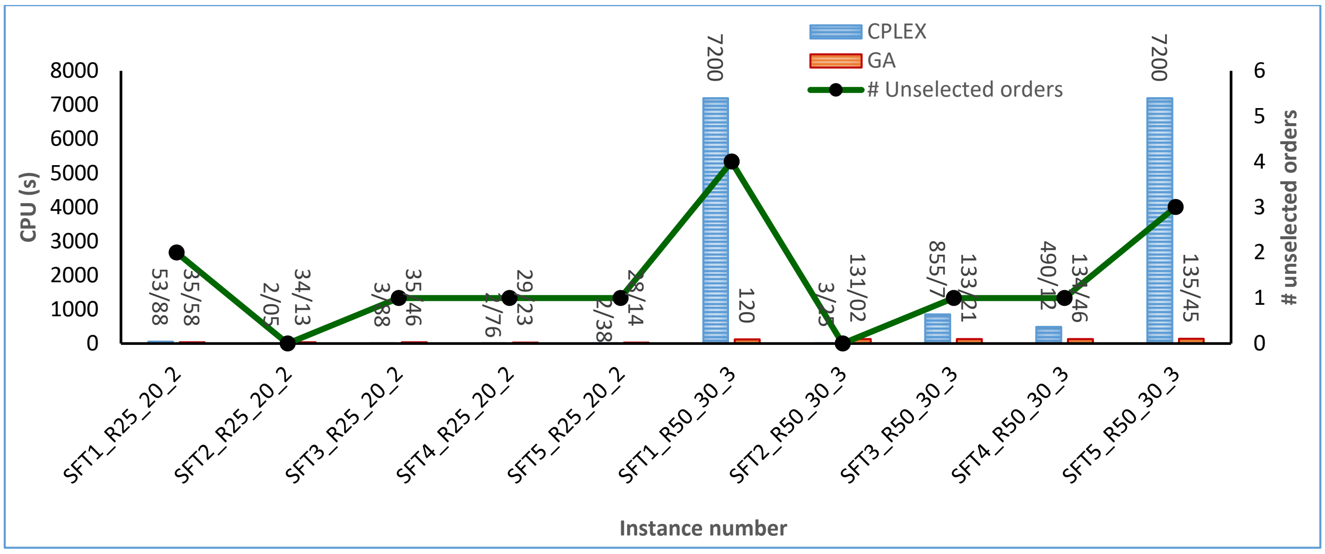


Figure 8. Computation time vs. Number of unselected orders for small and medium R-type instances

For the RC-type instances with 20 orders *($SFT1-5\_RC25\_20\_2$)*, the CPU time in these instances is close and doesn't surpass 13 s for CPLEX vs. 36 for GA, and the number of unselected orders is between 0 and 2. The CPU, in this case, is the same in the two classes C and R with the slightest difference depending on the number of unselected orders. For $n = 30$ *($SFT1-5\_RC50\_30\_3$)*, both solution approaches were able to solve the five instances to optimality. When $(widthtw, A_k^{max}) = (120, 480)$, we have two instances *($SFT1-2\_RC50\_30\_3$)*, the CPU time and the number of unselected orders of CPLEX are doubled from 66.97 s to 110.9 s and from 2 to 4, respectively. When $(widthtw, A_k^{max}) = (180, 720)$ *($SFT3-5\_RC50\_30\_3$)*, it is noted that though the instances have the exact same numbers of unselected orders, there is a big difference between them concerning the CPU time of CPLEX, which is due to the randomly generated data. We observe that in the case of 4 unselected orders, the CPU time in the class RC doesn't surpass 2289 s, whereas in class R, CPLEX failed to obtain optimal solutions, despite a larger time limit of 2 h. Therefore, the R-type instances are the hardest to solve, while the C-type instances are the easiest. For this reason, we have generated eight large R-type instances (up to 75 orders) to test the model and evaluate the performance of the proposed GA.

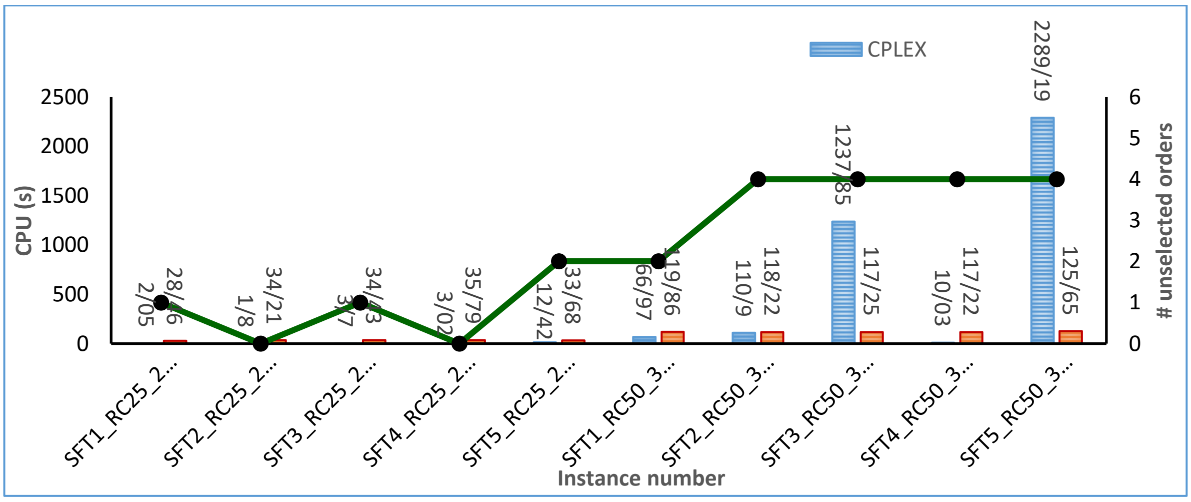


Figure 9. Computation time vs. Number of unselected orders for small and medium RC-type instances

5.2.2 Experiments on realistic-sized instances

It is difficult for CPLEX to produce optimal solutions for most large-sized instances within a 2 h time limit. For these instances, the average gap between GA and CPLEX is 2.08%. Therefore, GA is able to produce satisfying solutions for proposed model in a relatively short time.

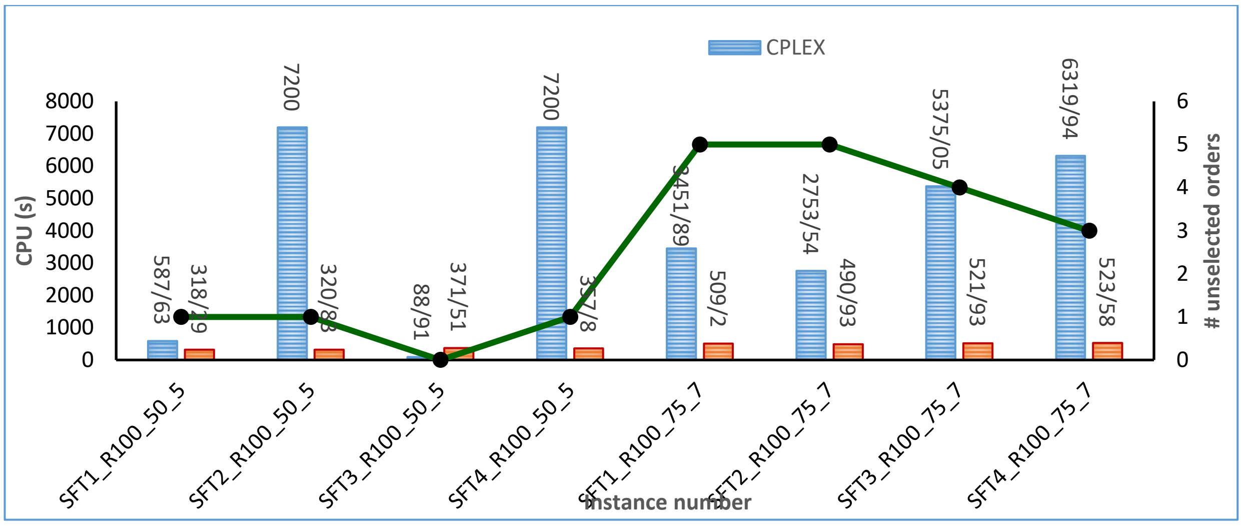


Figure 10. Computation time vs. Number of unselected orders for large R-type instances

As expected, we notice that the CPU time goes up substantially with the increase of the number of orders, especially when more orders can't be selected for delivery in the backhaul routes (see Table 5 and Figure 10). The selective aspect may be due to the following factors: the instance size, some orders are not profitable to serve, the TWs of demands, and the time intervals $\left[D_k^{min},\ A_k^{max}\right]$ during which the trucks are available.

5.3 Sensitivity analyses

To improve the result analysis of the proposed MILP model, we provide some sensitivity analyses considering the following parameters: the unit revenue and unit traveling cost. We changed one of these parameters while fixing the other and solved only the two difficult SFT instances of types R and RC with 30 orders; $SFT1\_R50\_30\_3$, $SFT5\_R50\_30\_3$, $SFT2\_RC50\_30\_3$, and $SFT5\_RC50\_30\_3$. For each one of the four instances, we run the CPLEX with no time limit imposed.

5.3.1. The effect of changes on parameter $\boldsymbol{p}$

In this section, we analyze the effect of the unit revenue p on the results by decreasing and increasing its value. To this end, the original value $p = 6$ is assigned values from the set $\{2, 4, 6, 8, 10\}$. Table 6 summarizes the results.

Table 6. Sensitivity analysis considering price per unit distance

Instance	p	c	LB_{MILP}	UB_{MILP}	Gap (%)	CPU(s)	# Nodes	# U	CPLEX Status
$SFT1\_R50\_30\_3$	2	1	**440**	**440**	**0.00**	**53.54**	**6155**	**4**	**Optimal**
	4	1	**1776***	**1781.72**	**0.00**	**19317.92**	**3231160**	**4**	**Integer optimal, tolerance**
	6	1	(3120)	(3294.04)	(5.28)	(16466.15)	(2967072)	(4)	Out of memory
	8	1	4463	4805.26	7.67	1465.21	1551109	**4**	Out of memory
	10	1	5801	6218.22	7.19	13805.20	2708482	**4**	Out of memory
$SFT5\_R50\_30\_3$	2	1	**578**	**578**	**0.00**	**0.81**	**29**	**5**	**Optimal**
	4	1	**2158**	**2158**	**0.00**	**196.50**	**26442**	**3**	**Integer optimal, tolerance**
	6	1	3742	3742*	0.00	9969.81	1381457	3	integer optimal, tolerance
	8	1	5332	5484.33	2.86	15199.88	2817924	3	Out of memory
	10	1	6919	7161.00	3.50	13755.17	2789270	3	Out of memory
$SFT2\_RC50\_30\_3$	2	1	**402**	**402**	**0.00**	**0.76**	**77**	**4**	**Integer optimal, tolerance**
	4	1	**1646**	**1646**	**0.00**	**4.76**	**1424**	**4**	**Integer optimal, tolerance**
	6	1	2889	2889*	0.00	110.90	20437	4	Integer optimal, tolerance
	8	1	4133	4133	0.00	405.35	53769	4	Integer optimal, tolerance
	10	1	5382	5382	0.00	923.43	174731	4	Integer optimal, tolerance
	2	1	**630**	**630**	**0.00**	**25.99**	**5169**	**4**	**Integer optimal, tolerance**

$SFT5\_RC50\_30/3$	4	1	**2341**	**2341**	**0.00**	**1228.98**	**168194**	**4**	**Integer optimal, tolerance**
	6	1	4056	4056*	0.00	2289.19	298813	4	Integer optimal, tolerance
	8	1	5772	5772	0.00	28456.47	3886803	4	Integer optimal, tolerance
	10	1	7460	7719.90	3.48	14923.43	3142671	4	Out of memory

It is concluded that by increasing the unit revenue, more transportation demands are satisfied; however, for much higher values of the mentioned parameter, CPLEX reports "out of memory" status. In contrast, when the unit revenue decreases, the CPLEX solves all four instances optimally, the CPU time decreases, and the number of unselected orders increases.

5.3.2. The effect of changes on parameter c

Here, we analyze the effect of the unit traveling cost c on the results by decreasing and increasing its value. To this end, the original value $c = 1$ is assigned values from the set $\{0.5, 1, 1.5, 2\}$.

Table 7. Sensitivity analysis considering traveling cost per unit distance

Instance	p	c	LB_{MILP}	UB_{MILP}	Gap (%)	CPU(s)	# Nodes	# U	CPLEX Status
$SFT1\_R50\_30\_3$	6	0.5	3566	3832.10	7.46	13374.37	2712834	**4**	Out of memory
	6	1	(3120)	(3294.04)	(5.58)	(16466.15)	(2967072)	(4)	Out of memory
	6	1.5	**2657**	**2801.92**	**5.45**	**1504.40**	**209684**	**4**	**Out of memory**
	6	2	**2212**	**2212**	**0.00**	**2194.54**	**210232**	**4**	**Integer optimal, tolerance**
$SFT5\_R50\_30\_3$	6	0.5	4241	4409.10	3.96	15896.30	2599243	3	Out of memory
	6	1	3742	3742*	0.00	9969.81	1381457	3	integer optimal, tolerance
	6	1.5	**3226**	**3226.00**	**0.00**	**330.41**	**26965**	**3**	**Integer optimal, tolerance**
	6	2	**2728**	**2728**	**0.00**	**7.32**	**1405**	**3**	**Integer optimal, tolerance**
$SFT2\_RC50\_30\_3$	6	0.5	3305	3305	0.00	4948.20	605198	4	Integer optimal, tolerance
	6	1	2889	2889*	0.00	110.90	20437	4	Integer optimal, tolerance
	6	1.5	**2455**	**2455**	**0.00**	**4.07**	**1141**	**4**	**Integer optimal, tolerance**
	6	2	**2035**	**2035**	**0.00**	**2.98**	**742**	**4**	**Optimal**
$SFT5\_RC50\_30\_3$	6	0.5	4597	4724.33	2.77	18857.67	3694247	4	Out of memory
	6	1	4056	4056*	0.00	2289.19	298813	4	Integer optimal, tolerance
	6	1.5	**3501**	**3501**	**0.00**	**976.96**	**160278**	**4**	**Integer optimal, tolerance**
	6	2	**2961**	**2961**	**0.00**	**350.47**	**81139**	**4**	**Integer optimal, tolerance**

The results reported in Table 7 indicate that when the unit traveling cost decreased from 1 to 0.5, CPLEX reports an "out of memory" status for most instances ($SFT1\_R50\_30\_3$, $SFT5\_R50\_30\_3$, and $SFT5\_RC50\_30\_3$). The instance $SFT2\_RC50\_30\_3$ is solved optimally, but the CPU time increased from 110.90 to 4948.20 s. Whereas by increasing the unit traveling cost to 2, the CPLEX is able to solve all four instances to optimality. For the instances $SFT5\_R50\_30\_3$, $SFT2\_RC50\_30\_3$, and $SFT5\_RC50\_30\_3$, which have been solved optimally in the original value $c = 1$, the CPU time decreased, respectively, from 9969.81 s to 7.32 s, from 110.90 s to 2.98 s and from 2289.19 s to 350.47 s.

6. Conclusion

In this paper, we have studied an order selection and routing problem with full truckload, multiple depots and time windows in the context of an empty return scenario, namely, SFTMDVRPTW. We have proposed a mathematical formulation of the problem as a MILP model. The objective function is to maximize the total profit, which equals the total revenue from the selected orders minus the overall cost (cost of moving loaded trucks, cost of moving empty trucks, and cost of waiting). CPLEX 12.10 has been used to solve 38 newly generated instances of up to 75 orders. The results demonstrate that the proposed MILP model provides plausible solutions for all small- and medium-sized instances; save for two instances, all other instances of up to 30 orders have been solved to optimality within a reasonable time. However, as the problem is NP-Hard, relatively large instances cannot be solved in this way. Therefore, a two-part chromosome representation and a new crossover using GA are proposed for solving the SFTMDVRPTW. The proposed GA is evaluated on the 38 generated instances and compared with CPLEX. The computational results indicate that the GA can produce satisfying solutions in a relatively short time. We remark that the proposed MILP model is strongly impacted by the selective aspect as well as the time window width of orders. Some sensitivity analyses have been performed to prove the validity of the proposed model and assess the effect of some parameters of the model on the solution quality and CPU time.

Our future research projects will study various facets of the SFTMDVRPTW, namely multi-objective SFTMDVRPTW. In addition, we will develop an efficient meta-heuristic algorithm for the problem, focusing on the hybridization of genetic algorithm, ant colony optimization and tabu search. Another research direction is to consider an SFTMDVRPTW under dynamic traffic environments.

References

Aras N., Aksen D. and Tuğrul Tekin M. (2011). Selective multi-depot vehicle routing problem with pricing. *Transportation Research Part C: Emerging Technologies*, Vol. 19(5), pp. 866-884.

Arunapuram S., Mathur K. and Solow D. (2003). Vehicle Routing and Scheduling with Full Truckloads. *Transportation Science*, Vol. 37(2), pp. 170-182.

Ávila T., Corberán Á., Plana I. and Sanchis J. M. (2017). Formulations and exact algorithms for the distance-constrained generalized directed rural postman problem. *EURO Journal on Computational Optimization*, Vol. 5(3), pp. 339-365.

Ball M. O., Golden B. L., Assad A. A. and Bodin L. D. (1983). Planning for truck fleet size in the presence of a common-carrier option. *Decision Sciences*, Vol. 14(1), pp. 103-120.

Baker B. M. and Ayechew M. A. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, Vol. 30(5), pp. 787-800.

Braekers K., Caris A. and Janssens G. K. (2013). Integrated planning of loaded and empty container movements. *OR Spectrum*, Vol. 35(2), pp. 457-478.

Braekers K., Caris A. and Janssens G. K. (2014). Bi-objective optimization of drayage operations in the service area of intermodal terminals. *Transportation Research Part E: Logistics and Transportation Review*, Vol. 65(1), pp. 50-69.

Berger J. and Barkaoui M. (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Computers & operations research*, Vol. 31(12), pp. 2037-2053.

Bettinelli A., Ceselli A. and Righini G. (2011). A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*,Vol. 19(5), pp. 723-740.

Bolaños R. I., Escobar J. W. and Echeverri M. G. (2018). A metaheuristic algorithm for the multi-depot vehicle routing problem with heterogeneous fleet. *International Journal of Industrial Engineering Computations*, Vol. 9(4), pp. 461–478.

Bouziyane B., Dkhissi B. and cherkaoui M. (2018). Solving a dynamic vehicle routing problem with soft time windows based on static problem resolution by a hybrid approach. *International Journal f Supply and Operations Management*, Vol. 5(2), pp. 134-151.

Caris A. and Janssens G. K. (2009). A local search heuristic for the pre-and end-haulage of intermodal container terminals. *Computers and Operations Research*, Vol. 36(10), pp. 2763-2772.

Carter A.E. and Ragsdale C.T. (2006). A new approach to solving the multiple traveling salesperson problem using genetic algorithms. *European Journal of Operational Research*, Vol.175 (1), pp. 246–257.

Cordeau J.-F., Laporte G. and Mercier A. (2004). An improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows. *Journal of the Operational Research Society*, Vol. 55, pp. 542-546.

Currie R. H. and Salhi S. (2003). Exact and heuristic methods for a full-load, multi-terminal, vehicle scheduling problem with backhauling and time windows. *Journal of the Operational Research Society*, Vol. 54(4), pp. 390-400.

Currie R. H. and Salhi S. (2004). A tabu search heuristic for a full-load, multi-terminal, vehicle scheduling problem with backhauling and time windows. *Journal of Mathematical Modelling and Algorithms*, Vol. 3(3), pp. 225-243.

Desrosiers J., Laporte G., Sauve M., Soumis F. and Taillefer S. (1988). Vehicle routing with full loads. *Computers and Operations Research*, Vol 15(3), pp. 219-226.

Dimitriou L. (2021). Optimal competitive pricing in European port container terminals: A game-theoretical framework. *Transportation Research Interdisciplinary Perspectives*, Vol. 9, pp. 1–12.

EL Bouyahyiouy K. and Bellabdaoui A. (2016). A new crossover to solve the full truckload vehicle routing problem using genetic algorithm. *3rd International Conference on Logistics Operations Management (GOL)*, 7731675, pp. 1–6.

EL Bouyahyiouy K. and Bellabdaoui A. (2017). An ant colony optimization algorithm for solving the full truckload vehicle routing problem with profit. *International Colloquium on Logistics and Supply Chain Management: Competitiveness and Innovation in Automobile and Aeronautics Industries (LOGISTIQUA)*,7962888, pp. 142-147.

Goldberg D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley Longman, Boston, MA.

Golden B. L. and Wong R. T. (1981). Capacitated arc routing problems. *Networks*, Vol. 11(3), pp. 305-315.

Grimault A., Bostel N. and Lehuédé F. (2017). An adaptive large neighborhood search for the full truckload pickup and delivery problem with resource synchronization. *Computers and Operations Research*, Vol. 88, pp. 1-14.

Gronalt M., Hartl R. F. and Reimann M. (2003). New savings based algorithms for time constrained pickup and delivery of full truckloads. *European Journal of Operational Research*, Vol. 151(3), pp. 520-535.

Gronalt M. and Hirsch P. (2007). Log-truck scheduling with a tabu search strategy.*Operations Research/ Computer Science Interfaces Series*,Vol. 39, pp. 65-88.

Holland J. H. (1975). Adaptation in natural and artificial systems, *Ann Arbor, MI*: University of Michigan Press

Ho W., Ho G. T., Ji P. and Lau H. C (2008). A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering Applications of Artificial Intelligence*, Vol. 21(4), pp. 548-557.

Imai A., Nishimura E. and Current J. (2007). A Lagrangian relaxation-based heuristic for the vehicle routing with full container load. *European Journal of Operational Research*, Vol. 176(1), pp. 87-105.

Jula H., Dessouky M., Ioannou P. and Chassiakos A. (2005). Container movement by trucks in metropolitan networks: Modeling and optimization. *Transportation Research Part E: Logistics and Transportation Review*, Vol. 41(3), pp. 235-259.

Lahyani R., Gouguenheim A. -L. and Coelho L. C. (2019). A hybrid adaptive large neighbourhood search for multi-depot open vehicle routing problems. *International Journal of Production Research*, Vol. 57(22), pp. 6963-6976.

Lahyani R., Khemakhem M. and Semet F. (2017). A unified matheuristic for solving multi-constrained traveling salesman problems with profits. *EURO Journal on Computational Optimization*, Vol. 5(3), pp. 393-422.

Lenstra J. K. and Rinnooy-Kan A. H. G. (1981). Complexity of Vehicle Routing and Scheduling Problems. *Networks*, vol. 11(2), pp. 221- 227.

Liu R., Jiang Z., Fung R. Y. K., Chen F. and Liu X. (2010a). Two-phase heuristic algorithms for full truckloads multi-depot capacitated vehicle routing problem in carrier collaboration. *Computers and Operations Research*, Vol. 37(5), pp. 950-959.

Liu R., Jiang Z., Liu X. and Chen F. (2010b). Task selection and routing problems in collaborative truckload transportation. *Transportation Research Part E: Logistics and Transportation Review*, Vol. 46(6), pp. 1071-1085.

Lysgaard J., Letchford A. N. and Eglese R. W. (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, Vol. 100(2), pp. 423-445.

Mirabi M., Shokri N. and Sadeghieh A. (2016). Modeling and Solving the Multi-depot Vehicle Routing Problem with Time Window by Considering the Flexible End Depot in Each Route. *International Journal of Supply and Operations Management*, Vol. 3(3), pp. 1373-1390.

Nossack J. and Pesch E. (2013). A truck scheduling problem arising in intermodal container transportation. *European Journal of Operational Research*, Vol. 230(3), pp. 666-680.

Powell W. B., Sheffi Y., Nickerson K. S., Butterbaugh K. and Atherton S. (1988). Maximizing profits for North American van lines' truckload division: A new framework for pricing and operations. *Interfaces*, Vol. 18 (1), pp. 21-41.

Rincon-Garcia N., Waterson B. J and Cherrett T. J. (2017). A hybrid metaheuristic for the time-dependent vehicle routing problem with hard time windows. *International Journal of Industrial Engineering Computations*, Vol. 8(1), pp. 141–160.

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

Uchoa E., Pecin D., Pessoa A., Poggi M., Vidal T. and Subramanian A. (2017). New benchmark instances for the Capacitated Vehicle Routing Problem. *European Journal of Operational Research*, Vol. 257(3), pp. 845-858.

Venkateshan P. and Mathur K. (2011). An efficient column-generation-based algorithm for solving a pickup-and-delivery problem. *Computers and Operations Research*, Vol. 38(12), pp. 1647-1655.

Wang X. and Regan A. C. (2002). Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, Vol. 36(2), pp. 97-112.

Xue N., Bai R., Qu R. and Aickelin U. (2021). A hybrid pricing and cutting approach for the multi-shift full truckload vehicle routing problem. *European Journal of Operational Research*, Vol. 292(2), pp. 500-514.

Yuan S., Skinner B., Huang S. and Liu D. (2013). A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms. *European Journal of Operational Research*, Vol. 228(1), pp. 72-82.

Yu B., Yang Z. Z. andYao B. Z. (2011). A hybrid algorithm for vehicle routing problem with time windows. *Expert Systems with Applications*, Vol. 38(1), pp. 435-441.

Zhang R., Yun W. Y. and Moon I. (2009). A reactive tabu search algorithm for the multi-depot container truck transportation problem. *Transportation Research Part E: Logistics and Transportation Review*, Vol. 45(6), pp. 904-914.

Zhang R., Yun W. Y. and Kopfer H. (2010). Heuristic-based truck scheduling for inland container transportation. *OR Spectrum*, Vol. 32(3), pp. 787-808.

Zolfagharinia H. and Haughton M. (2016). Effective truckload dispatch decision methods with incomplete advance load information *European Journal of Operational Research*, Vol. 252(1), pp. 103-121.