TRANSPORT
ISSN 1648-4142 / eISSN 1648-3480
2019 Volume 34 Issue 6: 692–707
https://doi.org/10.3846/transport.2019.11145

A MATHEMATICAL MODEL FOR THE CAPACITATED LOCATION-ARC ROUTING PROBLEM WITH DEADLINES AND HETEROGENEOUS FLEET

Soheila MIRZAEI-KHAFRI[1], Mahdi BASHIRI[2*], Roya SOLTANI[3], Mohammad KHALILZADEH[4]

[1, 4]*Dept of Industrial Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*
[2]*School of Strategy and Leadership, Faculty of Business and Law, Coventry University, Coventry, UK*
[3]*Dept of Industrial Engineering, Khatam University, Tehran, Iran*

Received 2 October 2017; revised 2 March 2018, 22 June 2018; accepted 13 September 2018;
first published online 26 September 2019

Abstract. This paper considers a Capacitated Location-Arc Routing Problem (CLARP) with Deadlines (CLARPD) and a fleet of capacitated heterogeneous vehicles. The proposed mixed integer programming model determines a subset of potential depots to be opened, the served roads within predefined deadlines, and the vehicles assigned to each open depot. In addition, efficient routing plans are determined to minimize total establishment and traveling costs. Since the CLARP is NP-hard, a Genetic Algorithm (GA) is presented to consider proposed operators, and a constructive heuristic to generate initial solutions. In addition, a Simulated Annealing (SA) algorithm is investigated to compare the performance of the GA. Computational experiments are carried out for several test instances. The computational results show that the proposed GA is promising. Finally, sensitivity analysis confirms that the developed model can meet arc routing timing requirements more precisely compared to the classical Capacitated Arc Routing Problem (CARP).

Keywords: capacitated location-arc routing, mixed integer programming, deadlines, genetic algorithm, simulated annealing, heterogeneous fleet.

Introduction and problem definition

Design of a distribution network is a fundamental step in building an efficient supply chain. This process requires decisions at different levels: the strategic level (e.g., location of depots); the tactical level (e.g., routing plans); and the operational level (e.g., vehicle and personnel scheduling). Several authors have warned of the incremental costs of considering these decisions separately (Frederickson 1979; Rand 1976; Salhi, Rand 1989). Location and routing decisions should be made as strategic and tactical decisions, respectively. The problem may be called the Location Routing Problem (LRP) or the Location-Arc Routing Problem (LARP), based on whether demands are located on edges or nodes of the network. To the best of the authors' knowledge, while the LRP has been much studied (Fazayeli *et al.* 2018; Prodhon, Prins 2014; Fazel Zarandi *et al.* 2013; Lopes *et al.* 2013), less attention has been paid to the LARP (Levy, Bodin 1989; Ghiani, Laporte 1999; Hashemi Doulabi, Seifi 2013; Lopes *et al.* 2014; Riquelme-Rodríguez *et al.* 2016). Ghiani and Laporte (2001) surveyed the most important applications, such as garbage collection, mail delivery and road maintenance, and solution approaches for the LARP. One of important extensions of the LRP, according to real-world requirements, is considering the servicing time window for each node, which is called the LRP with Time Window (LRPTW). Time windows ensure that each customer is visited within a specific time interval. The LRPTW includes three well-known types: (1) The LRP with hard time windows (LRPHTW), in which customers must be served within specific time windows (Wasner, Zäpfel 2004; Schittekat, Sörensen 2009; Gündüz 2011; Farham *et al.* 2018); (2) The LRP with soft time windows (LRPSTW), in which customers can be served outside of their time windows with a penalty (Nikbakhsh, Zegordi 2010; Gharavani, Setak 2015; Rabbani *et al.* 2018); (3) The LRP with one-sided hard time windows or deadlines (LRPD) (Aksen, Altinkemer 2008). In hard time windows, the vehicle can arrive at customers early. The wait times of the vehicle can be taken into account, with additional costs (Govindan *et al.* 2014) or without any cost (Setak *et al.* 2017).

*Corresponding author. E-mail: *mahdi.bashiri@coventry.ac.uk*

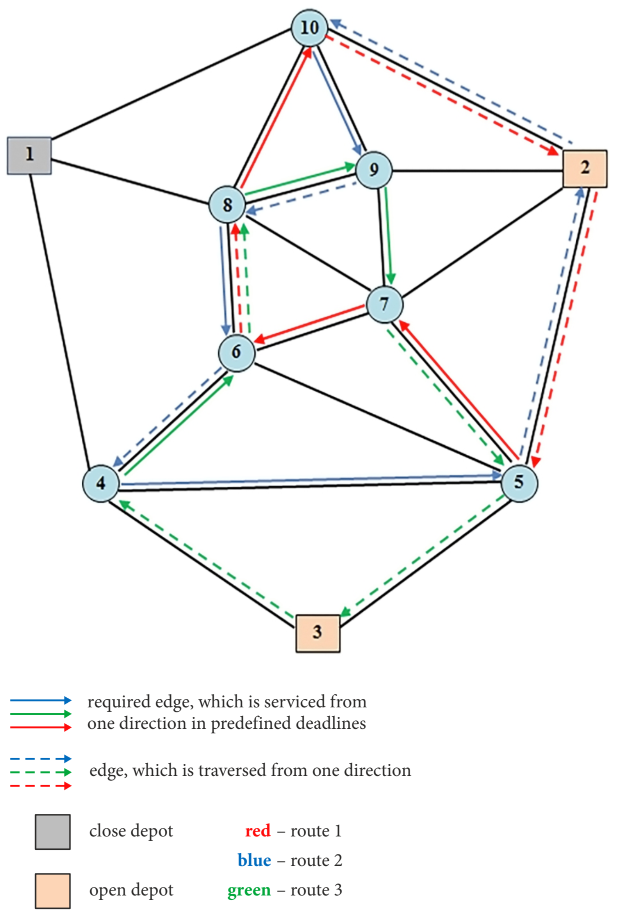
Generally, the ARP can be divided into three types: (1) the Chinese Postman Problem (CCP), in which all edges (or arcs) of a graph have positive demand and must be served; (2) the Rural Postman Problem (RPP), in which only a subset of edges (or arcs) have positive demand and must be served; (3) the Capacitated Arc Routing Problem (CARP) with vehicle capacity constraints. The LARP is an extension of the CARP in which demands are located on edges (or arcs) instead of nodes. The first formulation of the CARP with vehicle capacity constraints was introduced by Golden and Wong (1981). They proved that the CARP is Non-deterministic Polynomial-time hard (NP-hard). Therefore, the LARP is also an NP-hard problem (Nagy, Salhi 2007). A review of the literature shows that, while the LRP with various extensions related to time windows has been considered, the LARP has been overlooked. Therefore, a research gap exists with regard to application of extensions to time windows for the LARP. In the current paper, a variation of the LARP, the Capacitated Location-Arc Routing Problem with Deadlines (CLARPD), is considered, where the service at each required edge must be started before a predefined deadline.

A CLARPD is defined on an undirected graph $G(N, E)$ in which N and E are sets of nodes and edges, respectively. Edges and nodes represent two-way roads and junctions (or depots), respectively. All required roads must be serviced only once from one direction and within predefined deadlines with a heterogeneous fleet of vehicles. It's assumed that each depot has a special predefined covering radius. Thus, the first starting edges from the depots are assumed to be covered by other existing facilities, so the vehicles that start from the depots don't necessarily serve the edges that are after the depots. A feasible solution in an undirected network with 20 edges and 3 potential depots is shown in Figure 1. Note that the deadline constraint must be satisfied for edges that are served.

The aim of the present paper is to determine a subset of potential depots to be opened, to allocate roads that are served within predefined deadlines to opened depots, and to specify efficient routing plans to minimize total establishment and traveling costs. Because this is an NP-hard problem, two popular and efficient algorithms of Genetic Algorithm (GA) and Simulated Annealing (SA) are employed to solve the problem. So, the main contributions of the current study can be stated as follows:

- developing a mathematical model for the CLARPD in an undirected network, considering of the capacity for vehicles and depots, and heterogeneous fleet with deadline constraints;
- using a GA with proposed mutation and crossover operators along with a constructive heuristic to generate initial solutions;
- carrying out a sensitivity analysis on the effects of deadline characteristics on the proposed CLARPD model.

The rest of this paper is organized as follows. Section 1 provides a brief review of the literature. The problem statement and formulation of the proposed CLARPD is presented in Section 2. The solution algorithms for CLARPD are described in Section 3. Computational experiments are given in Section 4. Finally, discussion and conclusions are presented in the last section.


10
9
1
2
8
7
6
5
4
3
required edge, which is serviced from one direction in predefined deadlines
edge, which is traversed from one direction
close depot
open depot
red – route 1
blue – route 2
green – route 3
Figure 1. A sample feasible solution for a CLARPD instance

1. Literature review

The literature review is divided into two parts. The first focuses on the Arc Routing Problem with Time Window (ARPTW). Then the integrated LARP is reviewed.

The Arc Routing Problem (ARP) is an NP-hard problem. Therefore, three approaches are used to solve it: exact, heuristic and meta-heuristic algorithms. Exact approaches for the ARP have attracted less attention than the others. The first exact approach was presented by Hirabayashi *et al.* (1992) and was limited to 20 edges. Approaches such as the path-scanning heuristic (Golden *et al.* 1983), augment-merge heuristic (Golden, Wong 1981) and Ulusoy's heuristic (Ulusoy 1985) were proposed to solve the problem in larger instances. Some meta-heuristic approaches have had better results, such as the Tabu Search (TS) algorithm (Brandão, Eglese 2008), GA (Arakaki, Usberti 2018), memetic algorithm (Zhang *et al.* 2017), ant colony (Santos *et al.* 2010), and guided local search algorithm (Muyldermans, Pang 2010). For more studies on solution approaches and various aspects of the ARP, readers may refer to Wøhlk (2008).

The ARPTW is a variant of the ARP where each required edge must be served within a predefined time window. The ARPTW has not been studied much in the literature. Theoretical aspects of the ARPTW were considered by Wøhlk (2005). Some of the related studies have considered the problem in a node-based structure, while others have considered it in the original arc-based structure. In the first approach, the main problem (ARPTW) is converted to the Vehicle Routing Problem with Time Windows (VRPTW) (Mullaseril 1997; Wøhlk 2005; Tagmouti *et al.* 2007; Ciancio *et al.* 2018). Mullaseril (1997) investigated a CApacitated RPP considering Split Delivery and Time Windows (CARPSDTW). Two heuristics were used to solve it: the extended path-scanning heuristic, and the modified augment-merge heuristic with a column generation approach. Wøhlk (2005) presented two integer linear programming models for a CARP with Time Windows (CARPTW) in an undirected network; both node- and arc-based structures were used. This author developed a preferable neighbour heuristic for generating first solutions. Then the best solution was selected by a set covering problem. Tagmouti *et al.* (2007) studied a CARPTW in a directed network. They assumed that the service cost on each road is a linear function of the service start time. Then, they presented a nonlinear mixed integer programming model by transforming the CARP into a Capacitated Vehicle Routing Problem (CVRP); a column generation approach was used to solve it. This model was applicable to salt spreading operations. Ciancio *et al.* (2018) considered a Mixed Capacitated General Routing Problem with Time Windows (MCGRPTW) in a mixed network. They assumed that a subset of nodes, arcs and edges with positive demand must be served in predefined time windows. Then the problem in a node-based structure was solved by a branch price and cut algorithm.

The CARPTW has some applications, such as waste collection, street sweeping, winter road gritting and spreading, etc. Eglese (1994) studied a Multiple Depots Capacitated CPP (MDCCPP) for winter gritting operation, considering different priorities for roads. A heuristic for generating initial solutions with a SA algorithm was proposed to solve it. Haghani and Qiao (2001) presented a mixed integer programming model for a CARPTW, considering a salt spreading case in Maryland. Two proposed heuristics were used to generate initial solutions. Augment, delete, insert, and merge heuristics were proposed to improve the solutions. Reghioui *et al.* (2007) presented a Greedy Randomized Adaptive Search Procedure (GRASP) with path relinking for a CARPTW in an undirected network. Two constructive methods, first a randomized path-scanning heuristic and then a heuristic based on a route first-cluster second, were used to obtain initial solutions, which is the first step for many meta-heuristics. Then, or-opt, swap, and 2-opt heuristics were proposed to obtain near optimal solutions. Johnson and Wøhlk (2009) considered various solution approaches for a CARPTW in an undirected network. They proposed a heuristic method with column generation to solve it. Afsar (2010) considered a CARP with soft time windows in an undirected network. A branch-and-price algorithm was used to solve different instances.

The LARP has been considered less often than the ARP. Levy and Bodin (1989) studied an LARP for the first time, considering a postal carrier delivery case in the United States. They presented two heuristics: location–allocation–routing and allocation–routing–location, to solve it. Ghiani and Laporte (1999) used an exact method, branch-and-cut, for solving a special case of the LARP in the context of a RPP in an undirected network. Amaya *et al.* (2007) considered a special type of CARP with Refill Points (CARPRP) that is similar to the LARP. For this purpose, two types of vehicles are used. The first type is used to service the arcs, and is called a servicing vehicle. The second type is used to refill the servicing vehicle, and is called a refilling vehicle. They presented an integer linear programming model and solved it by a cutting-plane approach. Hashemi Doulabi and Seifi (2013) studied the LARP in a mixed network. They presented two mixed integer linear programming models for single and multiple depots. A SA algorithm with an insertion heuristic was proposed to solve it. Lopes *et al.* (2014) introduced several constructive heuristics for tackling the LARP, such as extended augment-merge and extended merge, and improvement heuristics such as reverse and relocate. They combined several meta-heuristics such as GRASP, TS and Variable Neighbourhood Search (VNS). The results showed that a combination of TS and GRASP have the best results. Riquelme-Rodríguez *et al.* (2016) studied a Periodic LARP with Inventory (PLARPI) in a mixed network. A mixed integer linear programming model was presented that is applicable to controlling dust on roads around open-pit mines. An Adaptive Large Neighbourhood Search (ALNS) was proposed to solve the problem. Chen *et al.* (2017) presented a mixed integer linear programming model for road maintenance operations as an LARP. A branch-and-cut algorithm and a three-stage heuristic algorithm are applied to solve various instances. An overview of the literature is summarized in Table 1.

The main research gap in the LARPs is the lack of time windows consideration. Table 2 provides a comparison of different aspects of the developed LARP models in the literature. Table 2 merely considers different aspects of the proposed model with previous studies in context of the LARP.

2. Problem statement and formulation

The CLARPD is defined on an undirected graph $G(N, E)$ in which N and E are the sets of all nodes and edges, respectively. The set N contains a non-empty subset J of potential depot locations $(J \subseteq N)$. The edge set $E = \{(i, j) : i, j \in N, i \neq j\}$ includes two-way roads. Let $E_R \subseteq E$ be the set of required edges that must be serviced only once from one direction. Each edge (i, j) can be replaced with two arcs (i, j) and (j, i). Therefore, directed graph $G' = (N, A)$ is constructed from graph G where arc set A is defined as $A = \{(i, j), (j, i) | (i, j) \in E\}$ and $A_R \subseteq A$ is the set of required arcs. Each edge with positive demand can be deadheaded if it is passed without being served by a vehicle. For instance, arcs (k, i) and (i, j) are deadheaded in Figure 2b. For each required edge $(k, i) \in E_R$, traversal and servicing times are

Table 1. A summary of previous related studies problem type

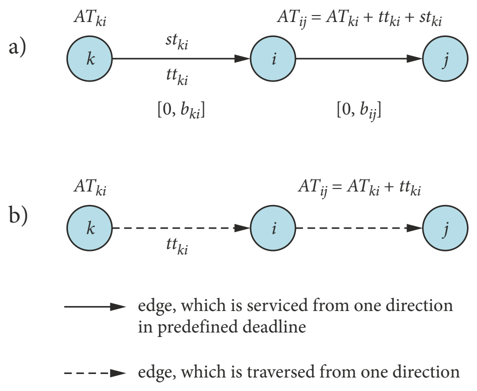
Authors	CARPTW	CARPSDTW	MDCCPPTW*	MCGRPTW	CARPRP	LARP	PLARPI	Solution approach		
								Exact	Heuristic	Meta-heuristic
Levy, Bodin (1989)	–	–	–	–	–	√	–	–	location–allocation–routing and allocation–routing–location	–
Eglese (1994)	–	–	√	–	–	–	–	–	proposed heuristic	SA
Mullaseril (1997)	–	√	–	–	–	–	–	column generation	path scanning and augment merge	–
Ghiani, Laporte (1999)	–	–	–	–	–	√	–	branch and cut	–	–
Haghani, Qiao (2001)	√	–	–	–	–	–	–	–	proposed heuristics, augment, delete, insert and merge	–
Wøhlk (2005)	√	–	–	–	–	–	–	set covering	preferable neighbour heuristic	–
Amaya *et al.* (2007)	–	–	–	–	√	–	–	–	cutting plane	–
Tagmouti *et al.* (2007)	√	–	–	–	–	–	–	column generation	–	–
Reghioui *et al.* (2007)	√	–	–	–	–	–	–	–	path scanning, route first cluster second, or-opt, swap and 2-opt	GRASP with path relinking
Johnson, Wøhlk (2009)	√	–	–	–	–	–	–	column generation	proposed heuristic	–
Afsar (2010)	√	–	–	–	–	–	–	branch and price	–	–
Hashemi Doulabi, Seifi (2013)	–	–	–	–	–	√	–	–	insertion heuristic	SA
Lopes *et al.* (2014)	–	–	–	–	–	√	–	–	augment-merge, merge, reverse and relocate	GRASP, TS and VNS
Riquelme-Rodríguez *et al.* (2016)	–	–	–	–	–	–	√	–	–	ALNS
Chen *et al.* (2017)	–	–	–	–	–	√	–	branch and cut	three-stage heuristic	–
Ciancio *et al.* (2018)	–	–	–	√	–	–	–	branch price and cut	–	–

Note: MDCCPPTW – Multiple Depots Capacitated CPP (MDCCPP) with Time Window

Table 2. Different aspects of developed LARP models in previous studies

Authors	Network			Multi depot	Capacitated depot	Capacitated vehicle	Vehicle		Hard time window		Periodic	Pick up & delivery	Split delivery	Inventory	Uncertainty	Methodology		Application
	Undirected	Directed	Mixed				Heterogeneous	Homogenous	Two sided	One-sided (deadline)						Type of model	Solution approach	
Levy, Bodin (1989)	√	–	–	√	–	–	–	√	–	–	–	–	–	–	–	–	heuristic	postal carrier delivery
Ghiani, Laporte (1999)	√	–	–	√	–	–	–	√	–	–	–	–	–	–	–	ILP	branch and cut	–
Amaya *et al.* (2007)	–	–	√	–	–	√	√	–	–	–	–	–	–	–	–	ILP	cutting plane	road marking
Hashemi Doulabi, Seifi (2013)	–	–	√	√	–	√	–	√	–	–	–	–	–	–	–	MILP	SA, insertion heuristic	–
Lopes *et al.* (2014)	√	–	–	√	√	√	–	√	–	–	–	–	–	–	–	–	GRASP, TS and VNS	–
Riquelme-Rodríguez *et al.* (2016)	–	–	√	√	–	√	–	√	–	–	√	–	–	√	–	MILP	ALNS	controlling dust on road
Chen *et al.* (2017)	–	√	–	√	–	√	–	√	–	–	–	–	–	–	–	MILP	branch and cut three-stage heuristic	road maintenance operations
Current paper	√	–	–	√	√	√	√	–	–	√	–	–	–	–	–	MILP	GA and SA	salt spreading operations

Notes: ILP – Integer Linear Programming; MILP – Mixed Integer Linear Programming.



a)
AT_{ki}
$AT_{ij} = AT_{ki} + tt_{ki} + st_{ki}$
st_{ki}
tt_{ki}
$[0, b_{ki}]$
$[0, b_{ij}]$
b)
AT_{ki}
$AT_{ij} = AT_{ki} + tt_{ki}$
tt_{ki}
k
i
j
edge, which is serviced from one direction in predefined deadline
edge, which is traversed from one direction

Figure 2. Graphic representation of arrival time in arc routing problem

shown by tt_{ki} and st_{ki}, respectively. As shown in Figure 2 the arrival time for arc (i, j) is actually defined as the beginning node arrival time of the arc (i, j) shown by AT_{ij}. The arrival time of each arc is calculated according to its predecessor served or traversed arcs.

It is assumed that each required edge i.e., edge $(i, j) \in E_R$, is associated with two deadlines, one for each direction. The deadline of each required arc $(i, j) \in A_R$ is defined as time window $\left[0, b_{ij}\right]$, in which b_{ij} represent the latest beginning service time for arc (i, j). It is important to note that the deadline constraint corresponding to arc (i, j) in Figure 2a must be satisfied. Unlike Figure 2a, in which required arc (i, j) is serviced, in Figure 2b, required arc (i, j) is only traversed; therefore, it is not essential that the deadline be satisfied. The assumptions of current study are as follows:

- each required edge must be serviced only once, from one direction, by one vehicle;
- a specific vehicle cannot traverse each edge more than once;
- vehicles are heterogeneous;
- depots and vehicles have predefined capacities;
- each edge that is serviced by a vehicle must satisfy a deadline;
- each vehicle should start from depot m $(m = 1, ..., |J|)$ in the beginning of a time horizon and returns to the same depot.

2.1. Notations

The sets, parameters and decision variables are introduced as follows.

Sets:

N – set of nodes;
J – set of potential depot locations $(m = 1, ..., |J|)$;
A_R – set of required arcs $(A_R \subseteq A,$
$A = \{(i, j), (j, i) : (i, j) \in E\})$;
K – set of vehicles $(k = 1, ..., |K|)$.

Parameters:

D_{ij} – demand of edge (i, j);
o_m – opening cost of a depot at node $m \in J$;
ca_m – capacity of a depot at node $m \in J$;
f_k – fixed cost for vehicle k;
c_{ij} – distance between nodes i and j;
Q_k – capacity of vehicle k;
tt_{ij} – traversal time of arc (i, j);
st_{ij} – servicing time of arc (i, j);
b_{ij} – latest beginning service time or deadline for arc (i, j);
M – a sufficiently large positive number.

Variables:

x_{ij}^k – takes 1 if an arc (i, j) is traversed by vehicle k, and 0, otherwise;
y_{ij}^k – takes 1 if an arc (i, j) is serviced by vehicle k, and 0, otherwise;
z_m – takes 1 if depot m is to be opened, and 0, otherwise;
w_{ijm} – takes 1 if the arc (i, j), which is serviced by its assigned depot m, and 0, otherwise;
u_i^k – an auxiliary variable for sub-tour elimination;
AT_{ij}^k – an arrival time to arc (i, j) by vehicle k.

2.2. Mathematical programming model

The model of a CLARPD is proposed as follows:

Minimize:

$$Z = \sum_{m \in J} o_m \cdot z_m + \sum_{(i,j) \in A} \sum_{k \in K} c_{ij} \cdot x_{ij}^k + \sum_{k \in K} \sum_{(m,j) \in A} f_k \cdot x_{mj}^k \quad (1)$$

subject to:

$$\sum_{k \in K} y_{ij}^k + y_{ji}^k = 1, \ \forall (i,j), (j,i) \in A_R; \quad (2)$$

$$x_{ij}^k \ge y_{ij}^k, \ \forall (i,j) \in A_R, \ \forall k \in K; \quad (3)$$

$$\sum_{(i,j) \in A_R} d_{ij} \cdot y_{ij}^k \le Q_k, \ \forall k \in K; \quad (4)$$

$$\sum_{j:(j,i) \in A} x_{ji}^k - \sum_{j:(i,j) \in A} x_{ij}^k = 0,$$
$$\forall i \in N, \ \forall k \in K; \quad (5)$$

$$\sum_{j:(m,j) \in A} x_{mj}^k \le 1, \ \forall k \in K, \ \forall m \in J; \quad (6)$$

$$u_i^k - u_j^k + N \cdot x_{ij}^k \le N - 1, \ \forall (i,j) \in A,$$
$$\forall i, j \in N \setminus J, \ \forall k \in K; \quad (7)$$

$$\sum_{h:(m,h) \in A} x_{mh}^k + y_{ij}^k \le w_{ijm}, \ \forall (i,j) \in A_R,$$
$$\forall m \in J, \ \forall k \in K; \quad (8)$$

$$\sum_{m \in J} \left(w_{ijm} + w_{jim}\right) = 1, \ \forall (i,j), (i,j) \in A_R; \quad (9)$$

$$\sum_{(i,j) \in A_R} d_{ij} w_{ijm} \le ca_m z_m, \ \forall m \in J; \quad (10)$$

$$AT_{ij}^k + tt_{ij} \cdot x_{ij}^k + st_{ij} \cdot y_{ij}^k -$$
$$M \cdot \left(1 - x_{ij}^k - y_{ij}^k + x_{ij}^k \cdot y_{ij}^k\right) \le AT_{jl}^k,$$

$$\forall (i,j),(j,l) \in A,\ \forall i,j \in N \setminus J,$$
$$\forall l \in N,\ \forall k \in K; \tag{11}$$

$$AT_{ij}^k + tt_{ij} \cdot x_{ij}^k + st_{ij} \cdot y_{ij}^k -$$
$$M \cdot \left(1 - x_{ij}^k - y_{ij}^k + x_{ij}^k \cdot y_{ij}^k\right) \le AT_{jl}^k,$$
$$\forall (i,j),(j,l) \in A,\ \forall j,l \in N \setminus J,$$
$$\forall i \in J,\ \forall k \in K; \tag{12}$$

$$AT_{mj}^k = 0,\ \forall (i,j) \in A,\ \forall m \in J,$$
$$\forall j \in N \setminus J,\ \forall k \in K; \tag{13}$$

$$AT_{ij}^k - M\left(1 - y_{ij}^k\right) \le b_{ij},$$
$$\forall (i,j) \in A_R,\ \forall k \in K; \tag{14}$$

$$x_{ij}^k \in \{0,1\},\ \forall (i,j) \in A,\ \forall k \in K; \tag{15}$$

$$y_{ij}^k \in \{0,1\},\ \forall (i,j) \in A_R,\ \forall k \in K; \tag{16}$$

$$z_m \in \{0,1\},\ \forall m \in J; \tag{17}$$

$$w_{ijm} \in \{0,1\},\ \forall (i,j) \in A_R,\ \forall m \in J; \tag{18}$$

$$AT_{ij}^k \ge 0,\ \forall (i,j) \in A,\ \forall k \in K. \tag{19}$$

Objective function (1) minimizes the total depot opening and vehicle fixed costs and also total traveling costs. Constraint (2) guarantees that each required edge is serviced only once, by one vehicle, from one direction. Constraint (3) states that arc (i,j) can be serviced by vehicle k only if that vehicle traverses arc (i,j). Constraint (4) considers vehicle capacity. Constraint (5) shows that routes should have continuity. Constraint (6) states that each vehicle may have at most one link from a depot. Constraint (7) eliminates illegal sub-tours. Constraints (8) and (9) guarantee that an edge serviced from one direction must only be assigned to a depot if there is a route between that depot and that edge. Constraint (10) considers depot capacity. Constraints (11) and (12) calculate arrival time for arc (i,j), which is either serviced or traversed by vehicle k. Constraint (13) expresses that arrival times for arcs starting from a depot are zero. Constraint (14) guarantees service at each required edge must be started before a predefined deadline. Therefore, servicing time limits are considered using Equations (11)–(14). Constraints (15)–(19) define types of variables. Since constraints (11) and (12) have nonlinear terms because they are a product of two binary variables x_{ij}^k and y_{ij}^k, they are transformed to linear forms by replacing the binary variable zz_{ij}^k as follows:

$$zz_{ij}^k = x_{ij}^k \cdot y_{ij}^k,\ \forall (i,j) \in A,\ \forall k. \tag{20}$$

Then, linear constraints (21) and (22) should be added to the mathematical model, as follows:

$$x_{ij}^k + y_{ij}^k - zz_{ij}^k \le 1,\ \forall (i,j) \in A,\ \forall k \in K; \tag{21}$$

$$2 \cdot zz_{ij}^k - x_{ij}^k - y_{ij}^k \le 0,\ \forall (i,j) \in A,\ \forall k \in K. \tag{22}$$

3. Solution approach

LARP is classified in NP-hard problems, so it is necessary to propose an efficient solution algorithm to solve the problem in medium and large size instances. Here, two popular and efficient algorithms are employed which are GA and SA. First we describe a solution representation of the problem which is essential step in designing of an algorithm then related operators and descriptions about the algorithm is presented in following subsections.

3.1. Solution representation

In the current paper, a solution representation is composed of $|K|$ vectors. Each vector corresponds to a vehicle and is made up of two parts. The first determines the sequence of traversing/servicing arcs traversed by a vehicle starting from an opened depot. The second includes binary values to determine the status of traversing (0)/servicing (1) arcs corresponding to the tour in part 1. Suppose that n_1, n_2, …, n_k are the number of nodes to be visited by vehicles 1, 2, …, k. Correspondingly, the second part for each vehicle will contain $n_k - 3$ binary values. Therefore, each chromosome is made up of $2 \cdot (n_1 + n_2 + \ldots + n_k) - 3 \cdot |K|$ genes. To clarify the encoding, consider the undirected network illustrated in Figure 1. It includes 9 required edges with three vehicles, and three potential depots. The encoding of a solution including 3 vehicles is shown in Figure 3. V_{11}, V_{21} and V_{31} are the tour plans of vehicles 1 to 3, respectively, as chromosome 1. In the following, the pseudocode of the constructive heuristic to generate initial solutions is depicted in Figure 4.

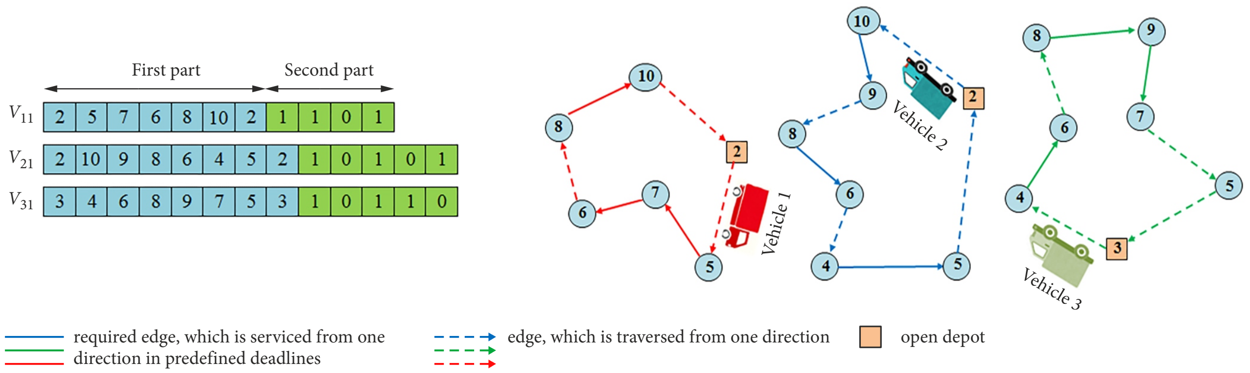

First part
Second part
V11 2 5 7 6 8 10 2 1 1 0 1
V21 2 10 9 8 6 4 5 2 1 0 1 0 1
V31 3 4 6 8 9 7 5 3 1 0 1 1 0
Vehicle 1
Vehicle 2
Vehicle 3
required edge, which is serviced from one direction in predefined deadlines
edge, which is traversed from one direction
open depot

Figure 3. An illustrative example of solution representation for the CLARPD

```
Algorithm 1: Constructive heuristics (Num_Vehicle, Required_ Edges, Popsize)
# First part
  For (k=1: Popsize)
      Active_Depots ← Randomly selects a set from potential depots based on a randomly defined number of depots
      For (j=1: Num_Vehicle)
      Selected _Depot ← Randomly selects a depot among Active_Depots
      Assign jth vehicle to the Selected_Depot in a random way.
      PopTour(k,j) ← Tour finder (Selected_Depot)
      End for
# Second part
  Set Exist_Postion_Edge=Exist_Vehicle=0;
  For (m=1: size(Required_ Edge))
     For (j=1: Num_Vehicle)
        if (Required_ Edge m exists in one of directions in PopTour (k,j) )
          Exist_Vehicle (j)=1;
          Exist_Positon_Edge (j) ← Position of Required_ Edge m in PopTour (k,j)
  End if
     End for
        if (Exist_Vehicle≠∅)
          Selected_Vehicle ← Randomly selects a vehicle among vehicles with the value of one of the vector Exist_Vehicle
          Postion ← Exist_Postion_Edge (Selected_Vehicle)
          Corresponding gen in second part of PopTour (k, Selected_Vehicle) considering Postion ← 1
        End if
     End for
End for
Algorithm 2: Tour finder (Selected _Depot)
Step1: Set Tour=[], Current_ Node=Selected _Depot;
Step2: In the graph remove all arcs that enter or exit to/from depots except selected depot.
Step3: In the graph specify all possible nodes that have a link to the Current_Node. If these nodes do not exist in the
       Tour, they can be added to Tour. The indices of these nodes are put in a set named Qualified-Nodes.
Step4: if (Qualified_Index≠∅), then randomly select a node from this set and calls it Selected_Node.
Step5: if (Selected_Node≠Selected _Depot), then remove the edge linking the current node to the Selected_Node in the
       graph and update it, then add the Selected_node to the end of the Tour vector, also set the Current_node=Selected_node
       and then go to Step3.
       else End.
```

Figure 4. Pseudocode of the constructive heuristics

3.2. Genetic algorithm

GA is a population-based search approach that arises from a natural selection process. It was introduced by Holland (1992). GA has been more popular due to its contribution in obtaining good solutions for complicated optimization problems in a reasonable amount of time (Cheng, Wang 2009). It has seen in various applications of combinatorial optimization problems, including different types of vehicle routing problem, especially where time windows are included (Baker, Ayechew 2003). In addition, Bräysy and Gendreau (2005) declare that during the past few years, numerous papers have been published on solving the VRPTW efficiently by the GA. Main reasons of using the GA in this paper can be stated as following:

- because of existing of three main operators (selection, crossover and mutation) in the GA, it searches solution space and this makes the algorithm converge towards high quality solutions within a few generations by well tuning of its parameters (Wang, Deng 2018);
- it should be noted that while the GA is inherently discrete, some algorithms, like Particle Swarm Optimization (PSO) are inherently continuous and should be redesigned to consider discrete design variables (Hassan *et al.* 2005);
- our investigation of solution approaches for routing problems revealed that there are lots of published papers on the topic that have used GA, SA and PSO. This encouraged us to use two algorithms, GA and SA, in the CLARPD problem.

Unlike conventional search approaches, a GA begins with a set of solutions or individuals, called the initial population. First, chromosomes are selected by the roulette wheel method, in which the probability of choosing an individual is directly proportional to its fitness value. Then pairs of selected chromosomes are crossed over. The resulting chromosomes are called offspring. Eventually, the offspring are modified by a mutation operator. It should be noted that each generated chromosome (solution) during the algorithm is considered to meet all constraints, like the servicing time limitation. In the case of a violation of a constraint, a penalty value is added to the objective function value based on the violation amount. This consideration leads the final chromosomes to serve their assigned edges in the predefined time limitations and achieve other constraints limitations. The pseudocode of the proposed GA algorithm is depicted in Figure 5. Additional details about the fitness function and these operators are provided in the next subsections.

3.2.1. Selection

There are different methods for selecting parents in an algorithm, such as tournament selection, fitness proportionate selection, and rank selection. In this paper, a roulette wheel selection operator is applied to create the next generation. In this approach, the probability of selecting an individual is directly proportional to its fitness value.

3.2.2. Crossover operator

For two selected parents, each time, the crossover operator starts with the first vehicle for both chromosomes and continues until the last vehicle. Two crossover operators are presented and analysed, single-point and heuristic crossovers. These proposed crossovers are performed for all vehicles of two parents to generate two offspring. The operators are described in more details as follows.

The single-point crossover is used only for cases where similar depots exist for the k-th vehicle in both parents. The proposed single-point crossover is performed for two vectors, V_{k1} and V_{k2}, corresponding to the k-th vehicle with the same depot of the first and second selected chromosomes. During this operator, tour plans of the k-th vehicle for both parents are considered. The identical nodes in the first part of both chromosomes for the k-th vehicle are extracted. One is randomly selected, and then the right side of the second parent tour is joined to the left side of the first parent tour. The same procedure is applied to generate the second offspring. Accordingly, second parts of offspring are inherited from their parents. During the crossover operation, some duplicated nodes may appear in the solution, so a repair approach is used to fix it. Suppose that there are two vectors, namely V_{11} and V_{12}, corresponding to the first vehicle of chromosomes 1 and 2, which are shown in Figure 6.

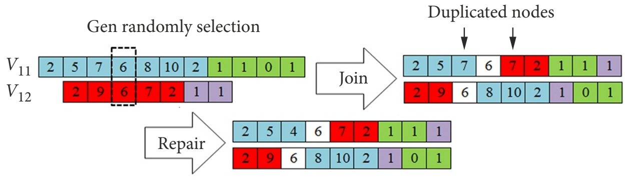


Figure 6. Example of the proposed single-point crossover operator

```
Initialize Popsize, Maxgen, Crossover probability ($P_C$), Mutation probability($P_M$), Erate and Drate
Generate initial population by constructive heuristic.
Set initial gen to 1;
While (gen<Maxgen)
Calculate the fitness value for the initial population.
Select parents from the current population in size {(1-Erate-Drate)* Popsize} based on the Roulette wheel selection approach.
  For i=1:2:(1-Erate-Drate)*Popsize
      if (rand<$P_C$)
        Select two parents $P_1$ and $P_2$.
        If they have the same depot, then do the single-point crossover operator to create offspring $O_1$ and $O_2$.
        If they have different depots, then do proposed heuristic crossover to create offspring $O_1$ and $O_2$.
        Add offspring to the new population.
      End if
  End for
For i=1:(1-Erate-Drate)*Popsize
      if (rand<$P_M$)
        Select two chromosomes from among the generated offspring and do a mutation operator on each of them.
        Add them to the new population.
      End if
  End for
%Elitism
     Transfer the elite chromosomes of the current generation to the next generation in size of Erate*Popsize.
     %Diversification
     For the remaining population for the next generation select chromosomes from the current population randomly to
     also appear in the next generation.
   gen=gen+1;
End while
Report the best solution found among the population of all generations.
```

Figure 5. Pseudocode of the proposed GA

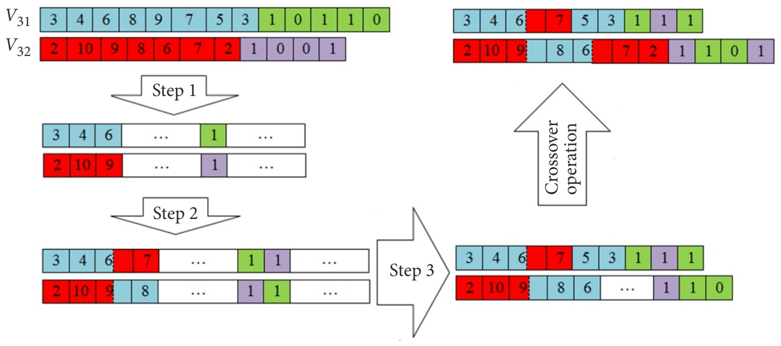


Figure 7. Example of the proposed heuristic crossover operator

A crossover point is selected for the first identical node in the first part of the chromosomes (node 6). Then two offspring are generated, but one needs some repair. Finally, the second parts of the new offspring should be defined by inheritance from their parents, as illustrated in Figure 6.

As mentioned before, the first crossover operator needs to have the same depots to be performed. However some cases may involve different depots, so a heuristic crossover is proposed that does not require depot uniqueness. The algorithm starts with the first parent, and the offspring inherits the first parent's genes until the same node is observed in both parents. Then the left/right gene of the same node is transferred to the offspring. This scheme is repeated until completion of a tour. The second part of the offspring is inherited from their parents, as described before. To clarify, the crossover steps are described in an example. Suppose that there are two vectors: V_{31} and V_{32}, corresponding to the third vehicle of chromosomes 1 and 2, which are shown in Figure 7.

3.2.3. Mutation operator

The proposed mutation operator is performed on generated offspring chromosomes by a heuristic method. During the proposed mutation, a gene of a chromosome is randomly selected. Then the left side of the first part remains unchanged, while the right side of a new chromosome is reconstructed by a constructive heuristic (# first part) approach. Then the values of the second part of the solution are defined according to the previous chromosome, before the selected gene and the value of the other remaining genes are randomly assigned. This approach ensures the feasibility of the solution. An illustrative example for the proposed mutation operator is shown in Figure 8.

3.3. Simulated annealing

The SA starts with an initial solution at a given temperature, called the initial temperature. Then, it searches the neighbourhood of the current solution in each iteration. If the new solution is not worse than the current solution, then the current solution will be replaced by the new solution. Otherwise, the new solution may be accepted with the Boltzmann probability function $\exp\left(-\frac{\Delta F}{C_B \cdot T}\right)$ in which parameters ΔF, C_B and T are the difference of objective function values for the mentioned solutions, the Boltzmann constant and temperature. The acceptance of the worse solution leads to preventing getting trapped in the local optimum. This iterative procedure is continued until the maximum iterations are met. Finally, the temperature is reduced using a cooling strategy (i.e., $T = R_C \cdot T_0$) and the algorithm is continued to search the neighbourhood of the current temperature until the stopping criterion (final temperature) is met. Note that at the beginning of the search, when the temperature is high, new solutions are easily accepted. However, near the end of search process, the acceptance probability of new solutions decreases. In this paper, we use the mutation operator as the neighbourhood search mechanism for the SA.

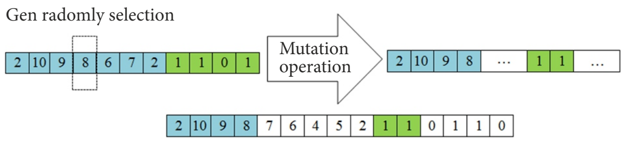


Figure 8. Example of the proposed mutation operator

3.4. Fitness function evaluation

The fitness function evaluates the quality of the solutions. This function includes total traveling and establishment costs, as well as total penalties for constraint violations, including vehicle or depot capacities, deadlines, and servicing of required edges. The algorithms try to find solutions that have both optimality and feasibility. As mentioned previously, a penalty approach was utilized for constraints violations. Total penalty for a solution x is calculated by following equation:

$$PENALTY(x) = \alpha \cdot \sum_{i=1}^{4} \text{cons\_violation}_i(x), \tag{23}$$

where: α is a constant parameter that reflects per unit cost of a constraint violation (mentioned parameter was tuned like other parameters of the algorithms and the tuning results have been reported Figures 11 and 12).

4. Computational experiments

Numerical experiments and the results obtained are discussed in this section. First, the numerical instances are introduced. Then sensitivity analyses are performed and the results are reported to show the validity of the proposed model. The proposed CLARPD model is compared with the classical LARP model to show the effectiveness of the proposed model. The proposed algorithm tuning results are also presented. Finally the performance of the considered algorithms is compared.

There are no benchmark instances in the context of LARP. So some sets of instances are generated randomly, which are shown in Table 3. Other instances were taken from the CARP literature (*https://www.uv.es/belengue/carp.html*) with some modifications, as defined in Table 4.

Computational experiments results for two aforementioned groups of small and large size instances are reported in this section. Randomly generated instances are described more here. Ten instances are randomly generated for each group, and the coordination of the considered nodes are randomly located in the interval of [90, 140]. A density index of 0.7 is considered to generate edges in the graph. Some of the edges are randomly determined as required edges. Demand of each required edge is randomly generated by a uniform distribution [100, 600]. The vehicles have capacities of 3000, 2000, 1500 and 1000 units of weight and fixed costs of 300, 200, 150 and 100, respectively. Traversal and service times are uniformly drawn from interval [0, 2]. Deadlines are distributed uniformly in interval [1, 40]. Other properties of instances are summarized in Table 3.

Table 3. Properties of randomly generated instances

Problem size	Problem	*N*	*E*	*R*	Number of depots	Depots capacity	Fixed cost for potential depots
Small	#S, 8, 15, 9, 2	8	15	9	2	3000	1000, 2000
	#S, 8, 15, 10, 2	8	15	10	2		
	#S, 8, 16, 8, 2	8	16	8	2		
	#S, 8, 18, 10 ,2	8	18	10	2		
	#S, 8, 19, 11, 2	8	19	11	2		
	#S, 8, 20, 9, 2	8	20	9	2		
	#S, 8, 20, 13,2	8	20	13	2		
	#S, 10, 19, 9, 3	10	19	9	3		1000, 2000, 3000
	#S, 10, 20, 8, 3	10	20	8	3		
	#S, 10, 23, 8, 3	10	23	8	3		
Large	#L, 11, 25, 15, 3	11	25	15	3		
	#L, 11, 34, 18, 3	11	34	18	3		
	#L, 12, 43, 26, 3	12	43	26	3	5000	
	#L, 15, 54, 31, 4	15	54	31	4	5000	1000, 2000, 3000, 4000
	#L, 17, 70, 33, 5	17	70	33	5	6000	1000, 2000, 3000, 4000, 5000
	#L, 20, 89, 47, 5	20	89	47	5	6000	
	#L, 25, 116, 51, 8	25	116	51	8	5000	1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000
	#L, 25, 135, 88, 8	25	135	88	8	7000	
	#L, 25, 164, 83, 8	25	164	83	8	8000	
	#L, 30, 212, 108, 10	30	212	108	10	9000	1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000

Table 4. Properties of modified instances based on the CARP literature

Problem size	Problem	*N*	*E*	*R*	Number of depots	Depots capacity	Fixed cost for potential depots	Vehicles capacity	Fixed cost for vehicles
Small	Kshs–1	8	15	8	2	400	200	100, 150	60, 90
	Gdb–4	11	19	11	2	10	20	5, 7	1, 2
	Gdb–20	11	22	9	2	40	60	20, 27	5, 10
	Gdb–1	12	22	14	2	10	20	5, 7	1, 2
	Gdb–10	12	25	12	2	15	25	7, 10	2, 4
	Gdb–5	13	26	14	3	10	20	5, 7	1, 2
Large	Gdb–22	11	44	22	2	70	100	20, 27	5, 10
	Bccm–1B	24	39	20	3	150	150	30, 40	10, 20
	Bccm–5A	34	65	32	4	200	200	30, 40, 50	10, 20, 30
	Bccm–4A	41	69	42	4	200	200	30, 40, 50	10, 20, 30

4.1. Sensitivity analysis

First, the effect of servicing deadlines on the routing scheme is considered. It was analysed for the #S, 8, 15, 9, 2. Deadlines were investigated in the following cases, which are shown in Table 5: Case0 (Ordinary), Case1 (Less tight), Case2 (Tighter) and Case3 (Even tighter).

In cases with tighter deadlines, ignoring arc deadlines causes a significant number of un-served required arcs, which has an increasing trend, as shown in Figure 9. This shows the necessity of considering deadlines in the proposed model.

Table 5. Values of deadlines for different arcs in Cases0, Cases1, Cases2, Cases3 on instance #S, 8, 15, 9, 2

Arcs deadlines / Cases	(3, –)	(4, –)	(5, –)	(6, –)	(7, –)	(8, –)
Case0	3	5	8	6	4	9
Case1	3	2	8	6	4	9
Case2	3	2	8	2.4	1.5	9
Case3	3	2	6.5	2.4	1.5	5.4

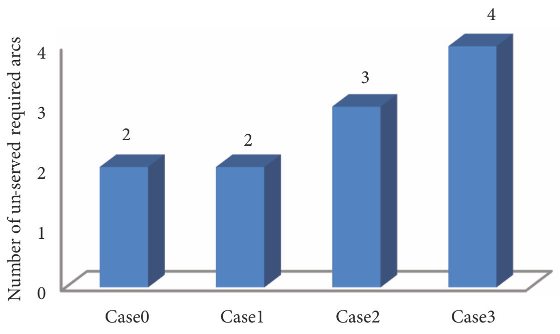
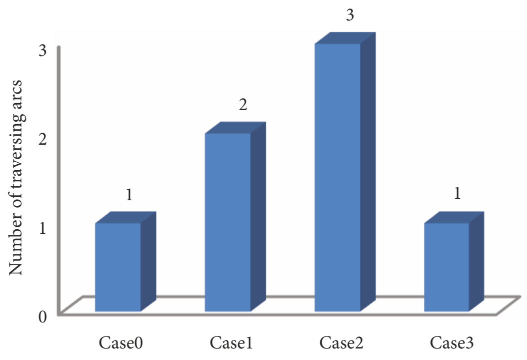


Figure 9. Effects of the tighter the deadlines on number of un-served required arcs

Figure 10 shows the effect of tightening deadlines on the number of traversing arcs and total costs. Tightening deadlines increases the number of traversing arcs to meet the predefined deadlines; consequently, total costs increase. In the last case of tightening deadlines, the number of traversing arcs decreases because of an increase in the number of vehicles from 3 to 4. Increasing the number of traversed arcs in the case of tighter deadlines is reasonable. A vehicle must traverse more arcs to achieve the arc deadlines. As an example, suppose that in the salt spreading case, there is an arc where salt needs to be spread within a tight deadline because of regional weather conditions. In this case, it is obvious that the salt spreader vehicle should traverse more arcs to serve the mentioned arc before its deadline, while the cost will increase, too.

4.2. Comparison of the proposed model with the classical LARP

As shown in Table 6, it is evident that the proposed model can serve all demands of all required arcs within their corresponding deadlines. It can be seen that the servicing scheme determined by the LARP model cannot serve all demands of arcs within their corresponding deadlines. This comparison confirms effectiveness and necessity of the proposed model. According to the performed analysis, in the case of tight deadlines, the mentioned models behave differently, while in the case of loose deadlines, both models show similar behaviour. This means that in cases like salt spreading, involving freezing weather and snow, it is more reasonable to use the CLARPD model to ensure safe roads.



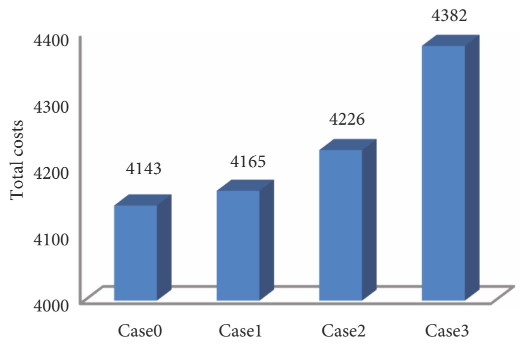


Figure 10. Effects of the tightening deadlines on the number of traversing arcs and total costs

Table 6. Comparison effects of the tighter the deadlines of the proposed model with classical LARP

Problem	Number of un-served required arcs in the CLARPD	Number of un-served required arcs in the LARP			
		Ordinary	Less tight	Tighter	Even tighter
Gdb–1	0	1	2	3	5
Gdb–4	0	3	3	4	4
Gdb–20	0	1	2	3	4
Gdb–10	0	1	2	4	6

4.3. Parameter setting for the proposed meta-heuristics

The performances of the proposed meta-heuristics depend on the values of their parameters. Population size (*Popsize*), number of generations (*Maxgen*), crossover probability P_C, mutation probability P_M, and penalty for each unit of constraint violation were considered to be tuned for the GA. Furthermore, the parameters of the proposed SA that were considered to be tuned were as follows: initial temperature T_0, final temperature T_F, cooling schedule R_C, maximum iterations at each temperature I_M, and penalty for each unit of constraint violation. In this paper, the Taguchi method is applied for setting parameters. There, the three different levels for each parameter to be considered are shown in Table 7. The result of this experimental design are shown in Figures 11 and 12.

The CLARPD for small sizes was solved by the CPLEX solver in GAMS 23.4.3. It was solved by the proposed GA and SA algorithms, too coded in MATLAB version 7.14.0.739 (R2012Aa) for large-size instances. All computations were run in an *IntelR Core i*5 2.5 GHz PC with 6 GB of memory. Its results are compared with the results of an exact solution, as reported in Table 8, and Figures 13 and 14. The current paper uses the Relative Percentage Deviation (RPD) that is calculated as $\frac{OFV(Sol)-OFV(Best)}{OFV(Best)}$, where *Sol* is the final solution found by the algorithms used to investigate the quality of the solutions obtained from the considered algorithms.

Table 7. GA and SA parameters and their levels

GA		SA	
Parameters	Values	Parameters	Values
Popsize	{50, 100, 200}	T_0	{200, 500, 1000}
Maxgen	{100, 200, 300}	R_C	{0.8, 0.85, 0.9}
P_C	{0.8, 0.85, 0.9}	I_M	{50, 100, 200}
P_M	{0.05, 0.1, 0.2}	T_F	{0.01, 0.1, 0.2}
Penalty	{2940, 3810, 4300}	*Penalty*	{2230, 2940, 3810}
Erate	{0.1, 0.15, 0.2}		
Drate	{0.1, 0.15, 0.2}		

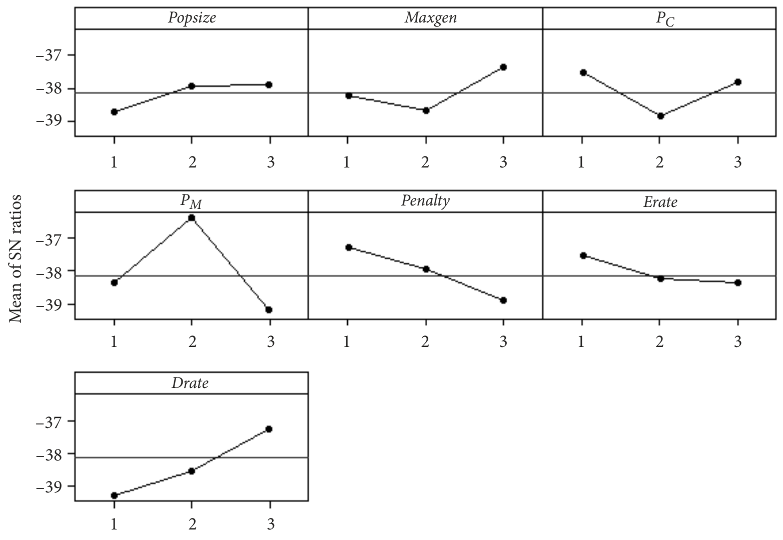


Figure 11. Main effects plot for SN ratio of the GA

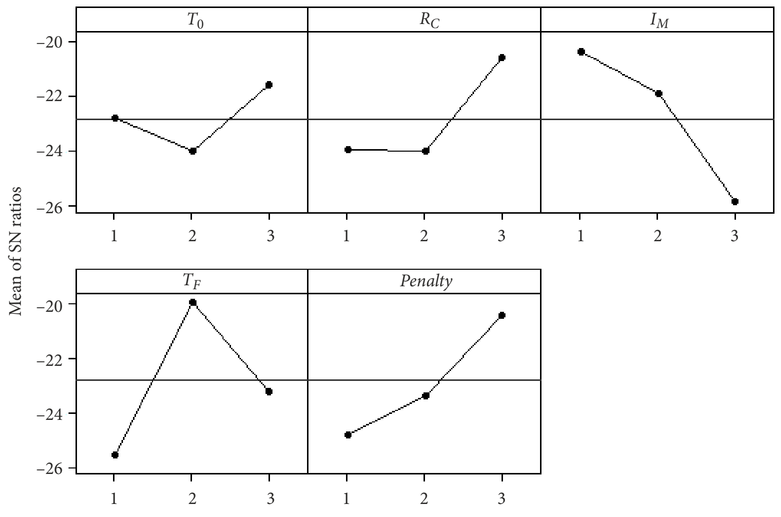


Figure 12. Main effects plot for SN ratio of the SA

Table 8. Performance comparison of GA and SA algorithms with an exact solution in small-sized instances

Problem number	Problem	GAMS			GA (20 times run)			SA		
		OFV	CPU time [s]	Absolute gap	OFV[a]	CPU time [s]	RPD	OFV[a]	CPU time [s]	RPD
1	#S, 8, 15, 9, 2	4143*	421.5	0	4143*	126.3	0	4202	6.5	1.42
2	#S, 8, 15, 10,2	3907*	742	0	3907*	199.5	0	3955	7.9	1.23
3	#S, 8, 16, 8, 2	3808*	1112.9	0	3808*	126.8	0	3926	10.4	3.1
4	#S, 8, 18, 10, 2	3729*	6113.1	0	3729*	122.2	0	3801	8.8	1.93
5	#S, 8, 19, 11, 2	3871*	113480	0	3880	87.6	0.23	4053	9.8	4.7
6	#S, 8, 20, 9, 2	3723*	20349.6	0	3737	129.1	0.38	3755	8.9	0.86
7	#S, 8, 20, 13, 2	3797*	40720	0	3797*	204.3	0	3889	7.4	2.42
8	#S, 10, 19, 9, 3	3816*	1362.3	0	3816*	164.5	0	4756	12.5	24.63
9	#S, 10, 20, 8, 3	3704*	2930.1	0	3706	185.3	0.05	3826	9.2	3.29
10	#S, 10, 23, 8, 3	3657*	5226.7	0	3675	251.2	0.49	3885	10.1	6.23
11	Kshs–1	6936*	39.725	0	6936*	331.6	0	7104	6.7	2.42
12	Gdb–4	249*	184.83	0	249*	268.5	0	271	5.3	8.83
13	Gdb–20	207*	563.1	0	207*	102.4	0	224	7.9	8.21
14	Gdb–1	270*	616.1	0	270*	98.6	0	356	6	31.85
15	Gdb–10	226*	1930	0	231	106.2	2.21	238	6.7	5.31
16	Gdb–5	273*	17103.6	0	280	447	2.56	314	8.7	15.02

Notes: *optimal solution; [a]objective function value.

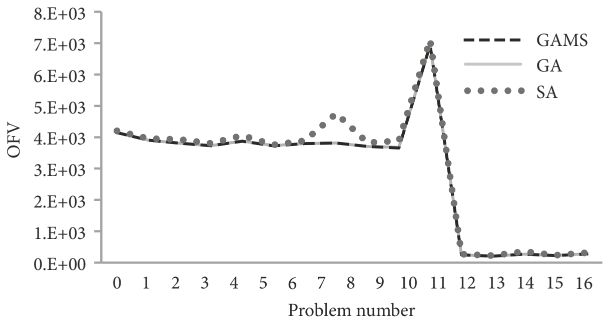


Figure 13. Computational results for small-sized instances

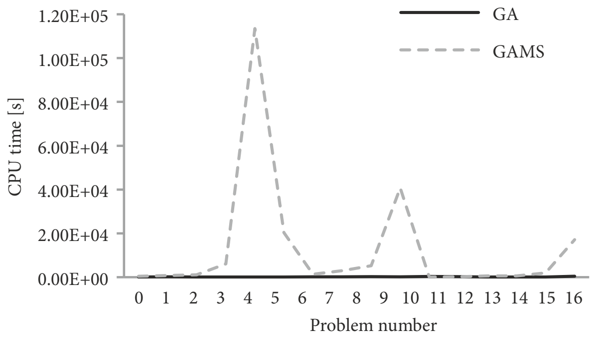


Figure 14. Comparison of computational times for the GAMS and the GA algorithm in small-sized instances

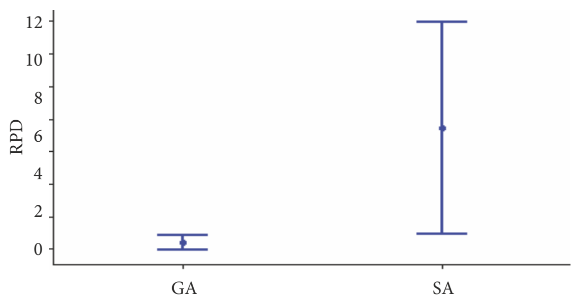


Figure 15. Mean and interval plots for the RPD of the GA and the SA algorithms

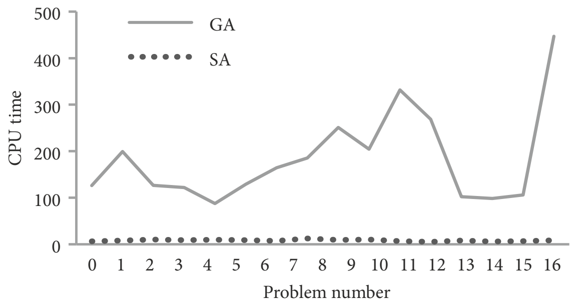


Figure 16. Comparison of computational times for the GA and the SA algorithms in small-sized instances

To examine the efficiency of the proposed algorithm, a 95% confidence interval graph of the RPD is depicted in Figure 15. It can be concluded that the RPD of the GA algorithm is much less than that of the SA. Thus, the GA algorithm is better than the SA algorithm in terms of solution quality. However, the SA requires less computational time, as illustrated in Figure 16. The result for large instances is shown in Table 9. As can be seen, the SA also shows weak performance compared to the GA in large instances.

Table 9. The results of GA and SA approaches for large-sized instances

Problem number	Problem	GA (20 times run)		SA	
		OFV[a]	CPU time [s]	OFV[a]	CPU time [s]
1	L, 11, 25, 15, 3	6004	166	7723	5.5
2	L, 11, 34, 18, 3	5152	317.3	6715	6.3
3	L, 12, 43, 26, 3	5738	319	9092	13.7
4	L, 15, 54, 31, 4	10020	491.2	12393	29
5	L, 17, 70, 33, 5	11692	594.20	18603	23.1
6	L, 20, 89, 47, 5	15100	1408.4	22585	27.2
7	L, 25, 116, 51, 8	31574	1124.2	58932	56
8	L, 25, 135, 88, 8	42240	1694	71924	60.4
9	L, 25, 164, 83, 8	30759	7254	58090	63.7
10	L, 30, 212, 108, 10	50165	10309.3	84046	84.1
11	Gdb-22	448	1080	616	20.1
12	Bccm-1B	689	2280	2758	40.2
13	Bccm-5A	1324	6668	3007	108.4
14	Bccm-4A	2047	8423.4	6951	174.2

Note: [a]objective function value.

Conclusions

This paper considers a location-arc routing problem in an undirected network. During the past decades, studies of more general types of the LARP have appeared. Therefore, we develop a mathematical programming model in which depots and vehicles have a limited capacity and roads should be serviced within predefined deadlines by a heterogeneous fleet.

The developed model can be used in real-life applications such as salt spreading operations. It can be used by planners for locating salt storage depots and specifying efficient routing plans for salt spreader trucks so as to minimize total costs, while some arcs need to be served within their deadline to prevent roads from icing up and to ensure safe roads.

Some sensitivity analyses were performed considering the effect of related parameters; the results illustrate the necessity and validity of the proposed CLARPD model. Then the proposed CLARPD model was compared with the classical LARP. The results confirm that the proposed model can serve all demands of all required arcs within their corresponding deadlines, while the servicing scheme determined by the classical LARP model cannot serve all demands of arcs within their corresponding deadlines. Because of the complexity of the problem, a GA with proposed operators and a SA algorithm, along with a constructive heuristic for initial solutions, are proposed. The proposed algorithms were tested on two group instances: benchmark sets from the CARP literature and random generated instances in small and large sizes. To confirm the validity of the model, the results of the GA and SA algorithms were compared with the result of the CPLEX solver. It was shown that the GA performs more efficiently.

This study has the following limitations:

- numerous surveys of node routing problems have been done, while reviews of arc routing problems are quite rare;
- because of nature of the considered problem, existing sub-tour elimination schemes cannot be used directly;
- inability of the CPLEX solver to solve the problem in large instances.

Consideration of sustainable development in LARP problems and network user preferences and interests could be appropriate directions for future study. Investigation of other servicing time limitations, such as soft and hard time windows for the proposed model, is another direction for future research. Finally, another study area could be developing more efficient solution algorithms, such as other greedy algorithms, and developing new math-heuristics and comparing them with existing meta-heuristics.

References

Afsar, H. M. 2010. A branch-and-price algorithm for capacitated arc routing problem with flexible time windows, *Electronic Notes in Discrete Mathematics* 36: 319–326. https://doi.org/10.1016/j.endm.2010.05.041

Aksen, D.; Altinkemer, K. 2008. A location-routing problem for the conversion to the "click-and-mortar" retailing: the static case, *European Journal of Operational Research* 186(2): 554–575. https://doi.org/10.1016/j.ejor.2007.01.048

Amaya, A.; Langevin, A.; Trépanier, M. 2007. The capacitated arc routing problem with refill points, *Operations Research Letters* 35(1): 45–53. https://doi.org/10.1016/j.orl.2005.12.009

Arakaki, R. K.; Usberti, F. L. 2018. Hybrid genetic algorithm for the open capacitated arc routing problem, *Computers & Operations Research* 90: 221–231. https://doi.org/10.1016/j.cor.2017.09.020

Baker, B. M.; Ayechew, M. A. 2003. A genetic algorithm for the vehicle routing problem, *Computers & Operations Research* 30(5): 787–800. https://doi.org/10.1016/S0305-0548(02)00051-5

Brandão, J.; Eglese, R. 2008. A deterministic tabu search algorithm for the capacitated arc routing problem, *Computers & Operations Research* 35(4): 1112–1126. https://doi.org/10.1016/j.cor.2006.07.007

Bräysy, O.; Gendreau, M. 2005. Vehicle routing problem with time windows, part II: metaheuristics, *Transportation Science* 39(1): 119–139. https://doi.org/10.1287/trsc.1030.0057

Chen, L.; Chen, B.; Bui, Q. T.; Hà, M. H. 2017. Designing service sectors for daily maintenance operations in a road network, *International Journal of Production Research* 55(8): 2251–2265. https://doi.org/10.1080/00207543.2016.1233363

Cheng, C.-B.; Wang, K.-P. 2009. Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm, *Expert Systems with Applications* 36(4): 7758–7763. https://doi.org/10.1016/j.eswa.2008.09.001

Ciancio, C.; Laganá, D.; Vocaturo, F. 2018. Branch-price-and-cut for the mixed capacitated general routing problem with time windows, *European Journal of Operational Research* 267(1): 187–199. https://doi.org/10.1016/j.ejor.2017.11.039

Eglese, R. W. 1994. Routeing winter gritting vehicles, *Discrete Applied Mathematics* 48(3): 231–244. https://doi.org/10.1016/0166-218X(92)00003-5

Farham, M. S.; Süral, H.; Iyigun, C. 2018. A column generation approach for the location-routing problem with time windows, *Computers & Operations Research* 90: 249–263. https://doi.org/10.1016/j.cor.2017.09.010

Fazayeli, S.; Eydi, A.; Kamalabadi, I. N. 2018. Location-routing problem in multimodal transportation network with time windows and fuzzy demands: presenting a two-part genetic algorithm, *Computers & Industrial Engineering* 119: 233–246. https://doi.org/10.1016/j.cie.2018.03.041

Fazel Zarandi, M. H.; Hemmati, A.; Davari, S.; Turksen, I. B. 2013. Capacitated location-routing problem with time windows under uncertainty, *Knowledge-Based Systems* 37: 480–489. https://doi.org/10.1016/j.knosys.2012.09.007

Frederickson, G. N. 1979. Approximation algorithms for some postman problems, *Journal of the ACM* 26(3): 538–554. https://doi.org/10.1145/322139.322150

Gharavani, M.; Setak, M. 2015. A capacitated location routing problem with semi soft time windows, *Advanced Computational Techniques in Electromagnetics* 1: 26–40. https://doi.org/10.5899/2015/acte-00197

Ghiani, G.; Laporte, G. 1999. Eulerian location problems, *Networks* 34(4): 291–302. https://doi.org/10.1002/(SICI)1097-0037(199912)34:4%3C291::AID-NET9%3E3.0.CO;2-4

Ghiani, G.; Laporte, G. 2001. Location-arc routing problems, *OPSEARCH* 38(2): 151–159. https://doi.org/10.1007/BF03399222

Golden, B. L.; Wong, R. T. 1981. Capacitated arc routing problems, *Networks* 11(3): 305–315. https://doi.org/10.1002/net.3230110308

Golden, B. L.; Dearmon, J. S.; Baker, E. K. 1983. Computational experiments with algorithms for a class of routing problems, *Computers & Operations Research* 10(1): 47–59. https://doi.org/10.1016/0305-0548(83)90026-6

Govindan, K.; Jafarian, A.; Khodaverdi, R.; Devika, K. 2014. Two-echelon multiple-vehicle location–routing problem with time windows for optimization of sustainable supply chain network of perishable food, *International Journal of Production Economics* 152: 9–28. https://doi.org/10.1016/j.ijpe.2013.12.028

Gündüz, H. I. 2011. The single-stage location-routing problem with time windows, *Lecture Notes in Computer Science* 6971: 44–58. https://doi.org/10.1007/978-3-642-24264-9_4

Haghani, A.; Qiao, H. 2001. Decision support system for snow emergency vehicle routing: algorithms and application, *Transportation Research Record: Journal of the Transportation Research Board* 1771: 172–178. https://doi.org/10.3141/1771-22

Hashemi Doulabi, S. H.; Seifi, A. 2013. Lower and upper bounds for location-arc routing problems with vehicle capacity constraints, *European Journal of Operational Research* 224(1): 189–208. https://doi.org/10.1016/j.ejor.2012.06.015

Hassan, R.; Cohanim, B.; De Weck, O.; Venter, G. 2005. A comparison of particle swarm optimization and the genetic algorithm, in *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 18–21 April 2005, Austin, Texas, US. https://doi.org/10.2514/6.2005-1897

Hirabayashi, R.; Saruwatari, Y.; Nishida, N. 1992. Tour construction algorithm for the capacitated arc routing problems, *Asia-Pacific Journal of Operational Research* 9(2): 155–175.

Holland, J. H. 1992. *Adaptation in Natural and Artificial Systems: an Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. A Bradford Book. 232 p.

Johnson, E. L.; Wøhlk, S. 2009. *Solving the Capacitated Arc Routing Problem with Time Windows using Column Generation*. Working Paper L-2008-0. Centre for Operations Research Applications in Logistics (CORAL), University of Aarhus, Denmark. 19 p. Available from Internet: https://pure.au.dk/portal/files/3875/L_2008_09.pdf

Levy, L.; Bodin, L. 1989. The arc oriented location routing problem, *INFOR: Information Systems and Operational Research* 27(1): 74–94. https://doi.org/10.1080/03155986.1989.11732083

Lopes, R. B.; Ferreira, C.; Santos, B. S.; Barreto, S. 2013. A taxonomical analysis, current methods and objectives on location-routing problems, *International Transactions in Operational Research* 20(6): 795–822. https://doi.org/10.1111/itor.12032

Lopes, R. B.; Plastria, F.; Ferreira, C.; Santos, B. S. 2014. Location-arc routing problem: Heuristic approaches and test instances, *Computers & Operations Research* 43: 309–317. https://doi.org/10.1016/j.cor.2013.10.003

Mullaseril, P. A. 1997. *Capacitated Rural Postman Problem with Time Windows and Split Delivery*. PhD Dissertation. University of Arizona, Tucson, US. 160 p. Available from Internet: https://repository.arizona.edu/handle/10150/282300

Muyldermans, L.; Pang, G. 2010. A guided local search procedure for the multi-compartment capacitated arc routing problem, *Computers & Operations Research* 37(9): 1662–1673. https://doi.org/10.1016/j.cor.2009.12.014

Nagy, G.; Salhi, S. 2007. Location-routing: Issues, models and methods, *European Journal of Operational Research* 177(2): 649–672. https://doi.org/10.1016/j.ejor.2006.04.004

Nikbakhsh, E.; Zegordi, S. 2010. A heuristic algorithm and a lower bound for the two-echelon location-routing problem with soft time window constraints, *Scientia Iranica* 17(1): 36–47.

Prodhon, C.; Prins, C. 2014. A survey of recent research on location-routing problems, *European Journal of Operational Research* 238(1): 1–17. https://doi.org/10.1016/j.ejor.2014.01.005

Rabbani, M.; Navazi, F.; Farrokhi-Asl, H.; Balali, M. H. 2018. A sustainable transportation-location-routing problem with soft time windows for distribution systems, *Uncertain Supply Chain Management* 6(3): 229–254. https://doi.org/10.5267/j.uscm.2017.12.002

Rand, G. K. 1976. Methodological choices in depot location studies, *Journal of the Operational Research Society* 27(1): 241–249. https://doi.org/10.1057/jors.1976.39

Reghioui, M.; Prins, C.; Labadi, N. 2007. GRASP with path relinking for the capacitated arc routing problem with time windows, *Lecture Notes in Computer Science* 4448: 722–731. https://doi.org/10.1007/978-3-540-71805-5_78

Riquelme-Rodríguez, J. P.; Gamache, M.; Langevin, A. 2016. Location arc routing problem with inventory constraints, *Computers & Operations Research* 76: 84–94. https://doi.org/10.1016/j.cor.2016.06.012

Salhi, S.; Rand, G. K. 1989. The effect of ignoring routes when locating depots, *European Journal of Operational Research* 39(2): 150–156. https://doi.org/10.1016/0377-2217(89)90188-4

Santos, L.; Coutinho-Rodrigues, J.; Current, J. R. 2010. An improved ant colony optimization based algorithm for the capacitated arc routing problem, *Transportation Research Part B: Methodological* 44(2): 246–266. https://doi.org/10.1016/j.trb.2009.07.004

Schittekat, P.; Sörensen, K. 2009. OR practice – supporting 3PL decisions in the automotive industry by generating diverse solutions to a large-scale location-routing problem, *Operations Research* 57(5): 1058–1067. https://doi.org/10.1287/opre.1080.0633

Setak, M.; Azizi, V.; Karimi, H.; Jalili, S. 2017. Pickup and delivery supply chain network with semi soft time windows: metaheuristic approach, *International Journal of Management Science and Engineering Management* 12(2): 89–95. https://doi.org/10.1080/17509653.2015.1136247

Tagmouti, M.; Gendreau, M.; Potvin, J.-Y. 2007. Arc routing problems with time-dependent service costs, *European Journal of Operational Research* 181(1): 30–39. https://doi.org/10.1016/j.ejor.2006.06.028

Ulusoy, G. 1985. The fleet size and mix problem for capacitated arc routing, *European Journal of Operational Research* 22(3): 329–337. https://doi.org/10.1016/0377-2217(85)90252-8

Wang, J.; Deng, W. 2018. Optimizing capacity of signalized road network with reversible lanes, *Transport* 33(1): 1–11. https://doi.org/10.3846/16484142.2014.994227

Wasner, M.; Zäpfel, G. 2004. An integrated multi-depot hub-location vehicle routing model for network planning of parcel service, *International Journal of Production Economics* 90(3): 403–419. https://doi.org/10.1016/j.ijpe.2003.12.002

Wøhlk, S. 2008. A decade of capacitated arc routing, *Operations Research/Computer Science Interfaces* 43: 29–48. https://doi.org/10.1007/978-0-387-77778-8_2

Wøhlk, S. 2005. *Contributions to Arc Routing*. PhD Dissertation. University of Southern Denmark, Denmark. 269 p.

Zhang, Y.; Mei, Y.; Tang, K.; Jiang, K. 2017. Memetic algorithm with route decomposing for periodic capacitated arc routing problem, *Applied Soft Computing* 52: 1130–1142. https://doi.org/10.1016/j.asoc.2016.09.017