Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Metaheuristic algorithm for the location, routing and packing problem in the collection of recyclable waste

Juan Sebastián Herrera-Cobo[a], John Willmer Escobar[b*] and David Álvarez-Martínez[c]

[a]*Data Science Revenue Growth Management Area, Mondelez International, Cali, Colombia*
[b]*Department of Accounting and Finance, Universidad del Valle, Cali, Cali 760001, Valle del Cauca, Colombia*
[c]*Department of Industrial Engineering, Universidad de Los Andes, Bogotá, Bogotá 110111, Colombia*

C H R O N I C L E

Article history:
Received February 3 2022
Received in Revised Format June 31 2022
Accepted August 26 2022
Available online August, 26 2022

Keywords:
Location Routing
Packing
Multi-compartment Vehicle Routing Problem
Recyclable Waste
Tabu Search
GRASP

A B S T R A C T

The increasing accumulation of solid waste worldwide has made it necessary to look for alternatives that improve the operation of recyclable waste collection systems to make waste treatment more profitable and eco-friendlier. This paper introduces a new variant of the multi-compartment vehicle routing problem (MCVRP) that considers the rearrangement or relocation of collection points and packing the demand. This problem is called the location packing multi-compartment vehicle routing problem (LPMCVRP) and is developed for a waste collection system using vehicles with flexible compartments. A mathematical formulation of the problem is proposed. A two-phase metaheuristic algorithm based on a tabu search without packing considerations and a variant that integrates a tabu search and a greedy randomized adaptive search procedure (GRASP) scheme with packing constraints have been proposed. A set of instances adapted from the literature is generated to validate the proposed solution strategy. The results obtained show the efficiency of the proposed solution scheme for optimizing collection systems.

1. Introduction

In recent decades, researcher interest in "green logistics," which can be defined as the execution of logistics activities in a more environmentally friendly manner, taking into account external factors such as waste, noise, energy, and carbon emissions, has been increasing (Dukkanci et al., 2019). Currently, there is worldwide concern about the exponential increase in waste accumulation. For this reason, between 1994 and 1995, the Organization for Economic Cooperation and Development (OECD) established the first stage of the basic principles of extended producer responsibility (EPR), with which 36 countries collaborate, necessitating the development of systems to recycle most of this waste. Countries like Sweden can reuse and recycle 99% of their recyclable waste (Wheeler, 2013). Similarly, Korea has a strict waste treatment policy where garbage collection points have additional posters with consumer information and segregating items into food waste, general trash, recyclables, and bulky items. Additionally, distribution costs make up a large part of the total logistics cost in a supply chain and are essential in optimizing garbage or recyclable waste collection system (Capelle et al., 2009).

Henke et al. (2015) present a routing problem in a waste collection system where vehicles collect different waste types. These recyclable wastes cannot be mixed during transport, and therefore, the vehicle's capacity is divided into different compartments. This problem is called the multi-compartment vehicle routing problem (MCVRP), which can also be applied to liquid transport problems (Caramia & Guerriero, 2010). The MCVRP is considered NP-hard since it is a variant of the capacitated vehicle routing problem (CVRP) (Henke et al., 2015), seeking to determine the best waste management routes.

* Corresponding author
E-mail: john.wilmer.escobar@correounivalle.edu.co (J. W. Escobar)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2023 Growing Science Ltd.
doi: 10.5267/j.ijiec.2022.8.004

This paper introduces a new variant of the MCVRP that considers the rearrangement or relocation of collection points to improve the routes developed and packing decisions. This problem is called the location packing multi-compartment vehicle routing problem (LPMCVRP). Specifically, we consider that waste can be modeled as "bins," which allows packing in multi-compartment vehicles in the location and routing problem. The decisions the LPMCVRP considers are a) to where should collection points be relocated? b) which routes will be used by the vehicles? c) which products will be collected at each collection point? d) into how many compartments should the capacity of each vehicle be divided? e) how should demand be packaged inside the container of the vehicle to optimize the space used?. The proposed solution to the LPMCVRP is a metaheuristic algorithm based on a tabu search without packing considerations that integrates the algorithm with a greedy randomized adaptive search procedure (GRASP) solution strategy for packing considerations. The packing strategy has a significant impact also in other industries like package delivery.

The proposed scheme's efficiency has been validated in adapted instances of the three-dimensional vehicle packing and routing problem (3L-CVRP) and the well-known location and routing problem (LRP). The results obtained are compared with the formal solution of the mixed-integer programming (MIP) model for the LPMCVRP.

The paper is divided as follows: Section 2 reviews the literature on the problems associated with the LPMCVRP, identifying the relevance of the research and the fundamentals for defining the problem introduced thus far. Section 3 provides a detailed formulation of the LPMCVRP through a mathematical model and the proposed methodology's detail. Section 4 discusses the tests carried out and the results of the proposed algorithm. Finally, Section 5 presents the conclusions of the research and possible future studies.

2. Literature Review

The well-known Vehicle Routing Problem (VRP) principles began with mathematician William Rowan Hamilton in the XIX century. His work considers a sequence of consecutive arcs of a graph that visits each vertex, once called the Hamiltonian circuit problem. As an extension of this, the Traveling Salesman Problem (TSP) emerged, which became the basis of all modern VRP research (Dantzig & Ramser, 1959). The authors consider the problem of dispatching gasoline trucks from one terminal to different service stations. The objective is to minimize the total distance travelled by the trucks, taking into account that the capacity of each is limited and that the demand for each service station must be fully satisfied (CVRP). The CVRP has been widely studied in the literature (Bernal et al., 2018; Linfati & Escobar, 2018).

One of the well-known variants of the CVRP is the multi-depot vehicle routing problem (MDVRP), which has been extensively studied (Escobar et al., 2014a; Chávez et al., 2016; Bolanos et al., 2018; Linfati & Escobar, 2018). In the MDVRP, there are several possible starting points for the vehicles, and therefore, each customer is assigned a route from one of the available depots. The MDVRP has many applications in real-world supply chains such as food delivery, chemicals, and carbonated beverages. The first heuristics for the MDVRP were proposed by Wren & Holliday (1972), Gillett & Miller (1974), Golden et al. (1977), and Raft (1982).

In parallel to all the methodologies previously presented for the VRP, integrations of this problem with other NP-hard problems have also been developed, such as the facility location problem (LP) and product packing (PP). The LP considers the geographical location of production or storage points in a supply chain, using fixed facility costs and transportation variable costs. The integration of MDVRP with the LP is called the location-routing problem (LRP) in the literature. It considers the depots that must be opened and the routes to be developed to meet the demand, seeking the minimization of the fixed facility costs of depots and the variable costs of the distance travelled (Escobar et al., 2013; Escobar et al., 2014b; Bernal-Moyano et al., 2017).

Prins et al. (2007) propose a cooperative algorithm for the LRP that alternates between the location and routing phases by exchanging promising edges. In the first phase, routes and customers are aggregated into super-customers used to locate the depots through Lagrangian relaxation of the assignment constraints. In the second phase, the routes are improved using the granular tabu search (GTS) heuristic. At the end of each iteration, information on the most used movements is recorded for the next iteration.

Vincent et al. (2010) developed a solution for the LRP based on the popular simulated annealing (SA) heuristic. This local search solution strategy avoids falling into local optima by accepting worse solutions with a certain probability. Kaya & Ozkok (2018) propose an algorithm based on SA for the LRP with inventory constraints. Escobar et al. (2013) propose a two-phase hybrid heuristic to solve the LRP. In the first phase, called the construction phase, initial solutions are generated with a division and grouping procedure to reduce the cost of the distance traveled. In the second phase, called the improvement phase, a modified GTS algorithm considers various diversification and intensification elements to improve the solution's quality. An improved version of this algorithm for the LRP has been proposed by Escobar et al. (2014b), in which the GTS efficiency and variable neighborhood search (VNS) are integrated. Dai et al. (2019) proposed a two-phase heuristic for three-echelon LRPs based on the savings algorithm.

Besides, the packing problem arises from the need to add a volumetric constraint on vehicles. With this, in addition to compliance with weight restrictions, the objective is to minimize the space used by the merchandise based on each product's dimensions, which creates a vital tool for improving operations, for example, in the container loading operation (Dereli & Daş, 2010). Applications of the packing problem can be found in Lodi et al. (2002), Wäscher et al. (2007), and Coffman et al. (2013). Escobar-Falcón et al. (2016) propose a hybrid two-phase algorithm for the packing and routing problem called 3L-CVRP. In the first phase, an optimization procedure based on cuts is used for the CVRP. In the second phase, the solutions obtained in the first phase are validated with a GRASP algorithm that restricts the packing of each route and optimizes the placement of packages in the container to simplify unloading. The integration of the routing problem with the packing problem generates a multicompartment delivery problem (MCDP). This new variant considers a CVRP with vehicle capacity fractioned into compartments in which different types of products that cannot be mixed are packed.

Coelho & Laporte (2015) present four MCDP categories that consider the possibility of dividing the product of one compartment among several customers and the constraint of satisfying customer demand in one or more deliveries. The variants of the problem have been solved using MIP models and branch-and-cut algorithms. The computational results for single- and multi-period cases yield exact solutions for 20 and 50 clients, respectively.

Henke et al. (2015) present a mathematical formulation and an algorithm based on VNS to solve an MCDP variant in Germany's waste collection system. However, the conditions in which the vehicle compartments must be configured are different from those found in classic applications of the MCVRP. First, the size of each compartment varies and can be determined individually for each vehicle. Second, the size of the compartments can vary only discreetly. That means that the compartment walls can be moved only to predefined positions. Last, the number of compartments can be less than or equal to the number of products. Heuristic and metaheuristic algorithms for the MCVRP have been proposed by Reed et al. (2014), Mendoza et al. (2010), El Fallahi et al. (2008), Abdulkader et al. (2015), Silvestrin & Ritt (2017), Chen & Shi (2019), Hübner & Ostermeier (2019) and Adil & Lakhbab (2020). Finally, recent studies by Farrokhi-Asl et al. (2018) and Gruler et al. (2015) have proposed algorithms for solving truck routing problems with constrained capacity and network configuration decisions.

However, despite the extensive literature on problems related to routing, location, and packing, similar studies that consider integrating the MCVRP and LRP in the collection of recyclable waste with packing considerations are not found in the literature. Similarly, none of the studies consider the relocation of the collection points with traditional packing constraints from the literature reviewed. Thus, this study greatly contributes to the operational improvement in the supply chain of emerging recycling and freight transport activities globally.

3. Proposed Methodology

3.1. Problem Definition and Formulation

A recyclable waste collection system has different operational components:

- The depots from which the vehicles depart and arrive optimize the coverage, the fixed costs of the fleet, and the adjusted storage capacity.
- The waste collection points (residential or industrial) are located to satisfy the coverage demand in a certain period and minimize the routing costs. It should be noted that this practice is common in Colombia and that the relocation of the collection points does not generate complaints in the users since, in the residential type, the changes are only from street to street and, in the industrial case, these changes occur inside an industrial complex.
- The number of vehicles and their capacity to minimize the resource idle or underutilization.
- The routes that all the vehicles must follow to meet the demand always trying to avoid unnecessary subtours.

The objective function is the minimization of the warehouse operating costs and vehicle travel costs. Additionally, current garbage collection vehicles (all waste in one large compartment) are unsuitable for a recyclable waste collection system. In this system, waste is separated as organics, recyclables, glass, and others, taking into account that waste is already arranged in garbage cans separated into these categories at the collection point. Therefore, the problem of recyclable waste collection requires vehicles with multiple compartments to avoid mixing the waste. These compartments divide the large garbage compartments with vertical walls and allow the ecological treatment of waste. In the LPMCVRP, collection point relocation is allowed; relocating to another place (potential relocation point) nearby allows the maintenance of coverage in the area and offers new possibilities for vehicle routing improvement. For example, if a collection point is moved to another street in the same block, unnecessary vehicle crossings due to lane direction can be avoided. With this consideration, the formulation of the problem becomes more attractive from a practical standpoint. The operator determines the number and the technical requirements of these potential relocation points.

The LPMCVRP can be modeled as a variant of the classic routing problem (CVRP), where the serviced customers' location can be changed to improve the route, and a single vehicle must meet the customer demand, including all the waste types. In addition to this, each customer generates different types of waste for collection, and each route starts and ends at a single depot where it can download all the waste types. When the waste is collected, each type of waste should be kept separate, and consequently, each truck capacity can be divided into different compartments before the route begins. Each truck must collect the entire customer demand; namely, the demand cannot be split. Specifically, two variants of the problem are proposed: with and without packing considerations. In considering the packing elements, the intent is to optimize the container's space and its loading and unloading according to the route established. This variant is developed to cover other problems in other industries like package delivery, creating more impact in the scientific community and making this research more applicable. Besides, this variant works as a reference since it combines location, routing, and traditional packing.

The LPMCVRP can be formally presented through the following mathematical model:

3.1.1. Sets

V $= \{0, \dots, d, \dots, d + s\}$ is a set of nodes of an undirected graph.
D $= \{1, \dots, d\}$ represents the number of depots.
S $= \{1, \dots, c, \dots, s\}$ is the set of customers (collection points) with their respective potential relocation points. In this problem, a potential relocation point is a street or space with all the requirements to set the waste relocation point.
$C \subset S$ $= \{1, \dots, c\}$ represents the initial locations of the collection points.
K $= \{1, \dots, k\}$ is the set of vehicles.
M_k $= \{1, \dots, m_k\}$ is a set indexed by k that represents the compartments of each vehicle.
P $= \{1, \dots, p\}$ is the set of types of recyclable waste.

3.1.2. Parameters

c_{ij} = Cost of traveling from node i to node j.
O = Cost of relocating a collection point (equivalent to distance).
F_k = Fixed cost of developing a route for vehicle k.
R_{ij} = Matrix which represents the potential relocation points (j) for each node i. Equals 1 if the change is possible, 0 otherwise. Example of R_{ij}: if we have two customers and we have two potential relocation points for each customer, the matrix will be 2x6 (2 for the current customer and the other 4 for the potential relocation points). In each row, the matrix must have three ones, or possible relocation points, one for the same position of i (that means that the customer is not relocated) and two for the relocation points. As shown below, the row for customer 1 has ones in the same customer position one and P1,1 and P1,2, while the second row has its ones in position two and P2,1 and P2,2.

R_{ij}	1	2	P1,1	P1,2	P2,1	P2,2
1	1	0	1	1	0	0
2	0	1	0	0	1	1

Q_k = Capacity of vehicle k.
d_{jp} = Demand for waste p of customer j.
q_{mk} = Relative size of compartment m in vehicle k.

3.1.3. Variables

The following decision variables are considered in the construction of the model:

x_{ijk} = binary variable, equals 1 if the vehicle k goes from node i to node j, 0 otherwise.
y_{ij} = binary variable, equals 1 if node i is relocated to node j, 0 otherwise.
u_{pkm} = binary variable, equals 1 if size q_m is assigned to vehicle k for waste type p, 0 otherwise.
s_{kj} = binary variable, equals 1 if the demand of point j is serviced with vehicle k, 0 otherwise.
f_{ij} = binary variable, equals 1 if node j starts its route at depot i, 0 otherwise.

3.1.4. Objective function

$$min\ Z = \sum_{i \in C} \sum_{j \in S \setminus i \neq j} O y_{ij} + \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} c_{ij} x_{ijk} + \sum_{i \in D} \sum_{j \in S} \sum_{k \in K} F_k x_{ijk} \tag{1}$$

3.1.5. Constraints

The constraints applied are as follows:

$$y_{ij} \leq R_{ij} \quad \forall\, i \in C, j \in S \tag{2}$$

$$\sum_{j \in S} y_{ij} = 1 \quad \forall\, i \in C \tag{3}$$

$$\sum_{i \in S} y_{ij} = 1 \quad \forall\, j \in S \tag{4}$$

$$\sum_{i \in V} \sum_{k \in K} x_{ijk} = \sum_{i \in V} y_{ij} \quad \forall\, j \in S \tag{5}$$

$$\sum_{i \in D} \sum_{j \in S} x_{ijk} \leq 1 \quad \forall\, k \in K \tag{6}$$

$$\sum_{i \in D} \sum_{j \in S} x_{jik} \leq 1 \quad \forall\, k \in K \tag{7}$$

$$\sum_{u \in S} x_{iuk} + \sum_{u \in V \setminus \{i\}} x_{ujk} \leq 1 + f_{ij} \quad \forall\, i \in D, j \in S, k \in K \tag{8}$$

$$\sum_{i \in A} \sum_{j \in A} x_{ijk} \leq |A| - 1 \quad \forall\, A \in S, k \in K \tag{9}$$

$$\sum_{j \in V} x_{ijk} - \sum_{j \in V} x_{jik} = 0 \quad \forall\, i \in V, k \in K \tag{10}$$

$$\sum_{p \in P} \sum_{m \in M_k} u_{pkm} \leq m_k \quad \forall\, k \in K \tag{11}$$

$$\sum_{j \in S} \sum_{p \in P} d_{jp} s_{kj} \leq Q_k \sum_{m \in M_k} \sum_{p \in P} q_{mk} u_{pkm} \quad \forall\, k \in K \tag{12}$$

$$\sum_{i \in V} x_{ijk} = s_{kj} \quad \forall\, j \in S, k \in K \tag{13}$$

$$\sum_{m \in M_k} u_{pkm} \leq 1 \quad \forall\, p \in P, k \in K \tag{14}$$

$$x_{ijk}, y_{ij}, u_{pkm}, s_{kj}, f_{ij} \in \{0,1\} \tag{15}$$

Eq. (1) is the objective function considering minimizing the node relocation costs, costs of the travelled distance, and fixed costs associated with the maintenance of the vehicle and the cost of the driver service. In addition, the constraints of this model are implicitly divided into three groups: relocation constraints, routing constraints, and packing and demand fulfilment constraints.

- *Relocation constraints*: Eq. (2) guarantees that the relocation of the collection points occurs only at allowed points. Eq. (3) and Eq. (4) restrict relocations to one per point and each relocation point to being assigned only once. Eq. (5) allows movements only to points that have been relocated (including the initial node). This fact means that, for the model, a point will always be relocated, but this is not considered in the objective function when it is relocated to its current location.
- *Routing constraints*: Eq. (6) and Eq. (7) force each vehicle to start and finish its route at a depot, and Eq. (8) guarantees the connection of a route to a depot. Eq. (9) ensures flow continuity, and Eq. (10) ensures the elimination of subtours.
- *Packing and demand fulfillment constraints*: Eq. (11) prevents the number of compartments from being greater than that allowed, Eq. (12) prevents the capacity of the vehicle from being exceeded, Eq. (13) requires that the demand of each point is collected by a single vehicle, and Eq. (14) forces the configuration of one compartment per type of recyclable waste. Note that the parameter Q_k for the model variant without packing considerations is specified as a parameter without any relation to the type of load. This aspect implies that d_{jp} assumes that all bins have the same density of material with different demands, namely, that each customer has different types of bins with different densities but that all the bins have the same density.

For the model with packing constraints, we consider that the sets of recyclable waste generate the total weight of the demand of the customer $j \in s$. Each set of recyclable waste has a length w_{jp}, height h_{jp} and width l_{jp}. The load of each vehicle has a length W_k, height H_k and width L_k. Thus, the new value of the demand parameter is $d_{jp} = \sum_{p \in P} w_{jp}\, h_{jp} l_{jp} \quad \forall j \in C$, and the new capacity of each vehicle is $Q_k = W_k H_k L_k$.

3.2. Metaheuristic algorithm

The proposed algorithm (Pseudocode 1) is divided into two phases: construction and improvement phases. A giant tour is generated in the construction phase with the Lin-Kernighan-Helsgaun (LKH) algorithm (Helsgaun, 2000), including all the current collection points (potential relocation points not included). Subsequently, some candidate clusters can be generated with the application of the well-known savings algorithm. In this manner, the set-partitioning problem is solved using a mixed-integer linear programming model (lines 1, 2, and 3). With the chosen clusters (or routes), this initial solution is then improved upon through a local search with four different operators, two inter-route and two intra-route, as shown in Figures 1-5. With this, the construction phase (lines 4, 5, and 6) ends.

A tabu search is applied with these four operators (Figs. 1-4). The two-opt operator works in a classical way where two nodes are selected, where the route crosses over itself, and reorder it to reduce the traveled distance. The relocation operator selects one of the potential relocation points and moves the customer there, so the distance between the previous node and the next node is reduced. The swap operator exchanges two customers of different routes, and the insertion operator adds a customer to an existing route (without exchange).

These operators allow the exploration of infeasible solutions based on excess capacity and make decisions about the relocation. Such infeasible solutions are penalized with a dynamic factor, which is adjusted during the search. Low values allow the algorithm to examine feasible solutions, while high values allow exploring infeasible solutions. In each iteration, the type of operator is chosen first with a probability of 50% for inter-route and 50% for intra-route. Once the type of operator is chosen, each operator's improvement is estimated concerning the value of the objective function of the current solution (lines 13 to 22). This estimation is done considering the distance cost improvement and the penalization for excessing capacity. The first part is the difference between the added and subtracted edges, and the second part is the excessed demand multiplied by a demand/distance factor (presented in Section 4).

Pseudocode 1. Algorithm LR

```
Input: V, demand, costs, maxIt, operators, prob
Output: routes
1 GiantTour = LKH(C ∈ V)
2 Clusters = SweepAlgorithm(GiantTour, demand)
3 S* = Set Partitioning Problem (Clusters, costs)
4 foreach op ∈ operators do
5       foreach r ∈ S* do
6             S* = r.apply(op)
7 BestS = S*
8 InitialS = S*
9 checkIt = maxIt/10;
10 it = 0
11 for it ≤ maxIt do
12      n = random(0, 1)
13      if n ≤ 0,5 then
14            foreach op1 ∈ InterOperators do
15                  estimate(S*, op1)
16            S* = S*.apply(bestOp1)
17      else
18            foreach op2 ∈ IntraOperators do
19                  estimate(S*, op2)
20            S* = S*.apply(bestOp2)
21      if S* ≤ BestS then
22            BestS = S*
23      if it == checkIt then
24            if S* > InitialS then
25                  S* = InitialS
26      checkIt+ = (maxIt/10)
27      n2 = random(0, 1)
28      if n2 ≤ prob then
29            S* = S*.apply(NewRoute)
30      update(prob)
31      it = it + 1
32 return BestS
```

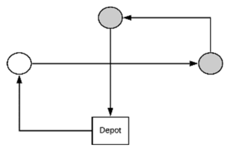

Fig. 1. Two-opt operator

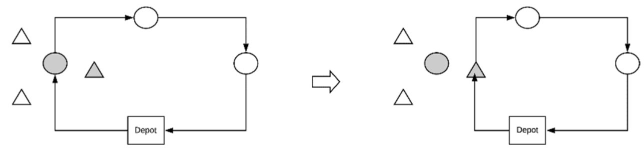

Fig. 2. Relocation operator

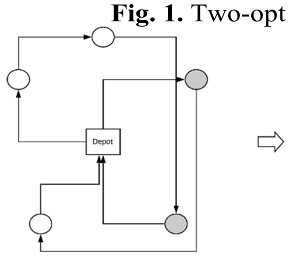

Fig. 3. Swap operator

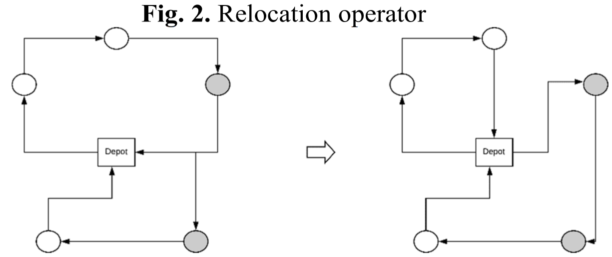

Fig. 4. Insertion operator

Source: Owner

Two strategies were implemented to make the tabu search more flexible and avoid continually falling into local optima or cycles. The first is to verify the current solution of the tabu regarding the initial solution in every tenth of the maximum iterations. If this solution has a greater objective function, the tabu's current solution is reset to the initial solution without deleting the tabu list so that what was saved in the last steps is not lost (lines 23 to 26). The second strategy uses a fifth operator (shown in Figure 5) that creates a new route with a randomly selected customer based on an initial probability, which increases each time this operator improves the solution during the search (lines 27 to 30).

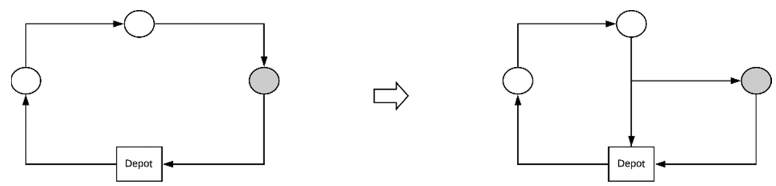

Fig. 5. New-route operator
Source: Owner

The variant of the problem includes packing considerations, using a GRASP algorithm based on a version adapted from the idea proposed by Escobar-Falcón et al. (2016) and Martínez et al. (2015) for the 3L-CVRP is proposed. The GRASP algorithm is based on the representation of maximum free spaces, allowing viable solutions to be obtained by controlling the generation and improvement of these spaces in the construction phase. This algorithm satisfies item orientation constraints, load-bearing strength, weight limits, load stability, and multi-drop constraints (loads with multiple destinations). The GRASP algorithm is an iterative procedure that combines a construction phase and an improvement phase. In the construction phase, a solution is built step by step, adding elements. The improvement phase seeks to apply a local search scheme through transitions on the built solutions. The proposed constructive strategy allows random selection of the residual (free) space and items in each iteration. The improvement phase partially deconstructs the built solution and refills it with a criterion focus on minimizing the residual spaces.

The procedure that validates the packing feasibility of a route is presented in Algorithm 2. Only one algorithm is defined to describe constructive and local search procedures; both procedures are repeated in an iterative scheme defined by the maximum number of iterations stipulated (line 1). In the construction, the packing pattern is generated from zero (line 2). The list of maximal spaces *S* is initialized with a free space equal to the original container space (line 3). In each iteration, we try to pack the boxes of each node, following the order given in the route to validate, thus *nextNode* is initialized in zero so that it references the first position of the route (line 4). In this way, the current node boxes are extracted from list *B* of boxes to be packed (line 5). The construction will only be completed until all free spaces are used or all nodes in the route are already packed (line 6). For this, the algorithm will repeat the container filling process as long as there are remaining boxes in the *Boxes* list (line 7). The filling process consists of first select an available space from the list *S*, choosing the free space furthest from the container base (line 8); ties are broken by selecting the space with the highest volume. Then, all the box layers that fit within the selected space are generated (line 9). The layers are built with boxes with several copies; a layer consists of a rectangular array of boxes positioned in rows or columns, six different layers (*XY, XZ, YX, YZ, ZX,* and *ZY*) can be obtained by combining the three axes. For all possible rotations of the box, its different layers must be built. Items with only one copy are also considered as layers. Having the layer list (*Cs*), the selection of the best ones (highest volume) must be made (lines 10 and 11). In this work, a dynamic selection process was established to adjust the number of candidate layers according to their historical performance. This way, a balance can be made between small RCL sizes, resulting in excessively greedy behavior, and large RCL sizes, which can rapidly compromise efficiency. The winning layer's selection is made by randomly choosing one layer from the RCL (line 11). This layer must be placed within the space to build the packing pattern so that the list of remaining boxes and the available maximal spaces can be updated (lines 12-14). The maximal spaces can be too small to pack any item, so it will be necessary to eliminate from the *Boxes* list all elements that cannot be fit into any space from S (line 14). When the *Boxes* list is exhausted, the S list must be updated to eliminate all the free spaces that will be unreachable for the cargo of the next node (line 15). The next node-set must be assigned, and the loading process will be repeated (lines 16 and 17). The solution built is checked to know if all the demanded cargo has been placed inside the container (line 18); if this occurs, the algorithm ends up returning that the route is feasible by packing (line 19); otherwise, the solution is tried to be improved, for this purpose a filtering strategy is implemented. (line 20).

Pseudocode 2. GRASP algorithm to validate the packing feasibility of a route

```
Parameters: totalIterations
Input: R: Route; O: Original container space; B: List of boxes of each node
Output: Feasible or Unfeasible Packing
1 for i=1 to totalIterations do
2     P= {}
3     S = O
4     nextNode = 0
5         Boxes = B_R[nextNode]
6         while S ≠ {} and nextNode < R.size() do
7             while Boxes ≠ {} do
8                 FreeSpace = SelectSpace(S)
9                 List C_s = GenerateLayersList(FreeSpace, Boxes)
10                List RCL = BuildRCL(C_s)
11                Layer L = SelectLayerRandomly(RCL)
12                Packing Pattern P = LocateLayer-Space(L, FreeSpace,P)
13            List S = UpdateListofMaximalSpaces(L, S)
14            Boxes = UpdateListofRemainingBoxes(L, Boxes)
15        List S = UpdateListofSpacesByNodeChanging(S)
16        nextNode = nextNode + 1
17        Boxes = B_R[nextNode]
18    if all B_c are packed in P
19        return Feasible
20    if f(P)>threshold then
21        Packing Pattern P = ImprovePattern(P,k)
22        threshold = f(P)
23        else
24            threshold = threshold·(3 ∗ i - 5 ∗ totalIterations - 2)/(10 - 10 ∗ totalIterations)
25    if all B_c are packed in P
26        return Feasible
27 return Infeasible
```

The filtering strategy (line 20) consists of applying the improvement phase only if the constructive phase's solution is promising. Filtering is crucial for computational efficiency as it avoids local search operations on solutions below a certain performance threshold. The threshold starts with the first solution's value and is updated at any iteration in which a new solution outperforms the threshold (lines 21 and 22). If a solution's objective does not reach the threshold, the local search procedure is not applied, and the threshold is decreased. As good solutions appear, the threshold becomes bigger (in a maximization context), leading to the rejection of many solutions and compromising diversity in the search. Therefore, the threshold is decreased following a scheme that depends on the current iteration (i) and the parameter *totalIterations*, allowing large reductions at the beginning of the search process and making small reductions as the iterations are exhausted. (line 24).

The function *ImprovePattern* modifies the packing pattern from the built solution removing 30% of boxes in the local search. Removal consists of eliminating item by item from the container door backward. In this way, the "half-empty" container, together with the removed and no packed items, must enter again on the packing process described in the construction phase, but now from the layer list (*Cs*) are selected the layers with best-fit to the free spaces (minimizing the residual spaces that will be generated after placing the layer). Later, the improved solution is checked to know if all the demanded cargo has been placed inside the container (line 25); if this occurs, the algorithm ends up returning that the route is feasible by packing (line 26); otherwise, another iteration of the GRASP algorithm will be executed. The constructive and improvement phases are executed a *totalIterations* number of times (line 1). The algorithm returns *Unfeasible* if none of the iterations has managed to pack all the cargo demanded in the route (line 27).

In particular, the proposed algorithm is different from concerning the traditional tabu search in only two ways: 1) routes that do not fit in the container of the vehicle (validated with the GRASP algorithm) are eliminated before entering the set-partitioning problem, and 2) the local search and the tabu search do not consider solutions that do not comply with the packing. Specifically, line 6 of Pseudocode 1 is modified by applying the operators before entering the improvement phase to validate the best estimate with the GRASP Algorithm (*apply*(*op, GRASP* ())); additionally, in line 29, the new route (*estimate*(*S*∗*, op, GRASP* ())) operator is applied. The proposed approach's main strength is that the computational effort is focused mainly on solving the LRP with the tabu metaheuristic. In contrast, a high-performance GRASP approach solves the loading problem. The GRASP approach is calibrated according to the characteristics of the items delivered, namely, the accumulated demand of customers covered by each route. Mainly, if the solution is feasible, the procedure must verify the packing of the demand for each route using the GRASP algorithm, which determines if it is possible to pack the bins in the vehicles with a loading pattern without reordering. Figure 6 shows an illustrative example of a GRASP solution for a route where the bottom bins should be those of the last customer to maintain chronological loading and unloading and facilitate the vehicle's operation.

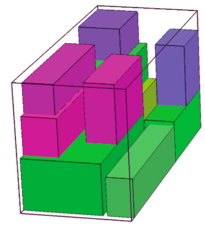

Fig. 6. Example of a packing solution using the GRASP algorithm
Source: Owner

4. Tests and result analysis

To validate the algorithm, 27 instances of the well-known 3L-CVRP, available in the library published in http://or.dei.unibo.it/instances/three-dimensional-capacitated-vehicle-routing-problem-3l-cvrp, are modified. The test instances include a range of 15 customers to 100 customers with volume information for the first variant of the proposed algorithm and bin dimensions for the second variant (including packing). Three random relocation nodes are added to each instance in a radius around each node; this parameter varies between 2 and 4 depending on the nodes' proximity to prevent the relocation points from exceeding the current nodes, thus preserving the coverage criteria for the areas served. Pseudocode 3 shows the procedure for generating the relocation points.

Pseudocode 3. Procedure for generating N random points around a collection point

```
Input: numPoints, radius, x, y
Output: Random points
result = vector()
i = 0
for i < numPoints do
    point = vector()
    x1 = 0
    y1 = 0
    angle = rand()%361;
    if angle ≥ 0 and angle ≤ 90 then
        x1 = radius ∗ cos(angle ∗ PI/180,0)
        point.add(int(x + x1))
        y1 = radius ∗ sin(angle ∗ PI/180,0)
        point.add(int(y + y1))
    else
        if angle > 90 and angle < 180 then
            angle = 180 − angle
            x1 = radius cos(angle ∗ PI/180,0)
            point.add(int(x − x1))
            y1 = radius ∗ sin(angle ∗ PI/180,0)
            point.add(int(y + y1))
    else
        if angle ≥ 180 and angle ≤ 270 then
            angle = 270 − angle
            x1 = radius ∗ sin(angle ∗ PI/180,0)
            point.add(int(x − x1))
            y1 = radius ∗ cos(angle ∗ PI/180,0)
            point.add(int(y − y1))
    else
        angle = 360 − angle
        x1 = radius ∗ cos(angle ∗ PI/180,0)
        point.add(int(x + x1))
        y1 = radius ∗ sin(angle ∗ PI/180,0)
        point.add(int(y − y1))
    result.add(point)
return result
```

The modified instances and results obtained are available in Herrera (2020a). The code for the first and second variants are found in Herrera (2020b) and Herrera (2020c), respectively. The proposed algorithm is coded in C ++ with the "CodeBlocks" development environment, in Linux, and using the VRPH libraries (Groër et al., 2010) and Lin-Kernighan Heuristic – LKH (Helsgaun, 2000). The tests are run on an Acer Spin 5 computer with 8 GB RAM and an Intel(R) Core (TM) i5 7200 CPU @ 2.50 GHz. The mathematical model described in Section 3.1 is used as a reference point in validating results using solver CPLEX 12.4. Python is used in the Spyder development environment, available in the Anaconda package, to generate the graphs.

Table 1 summarizes the instances' information and specifies the instance name, number of collection points, and relocation costs. For all instances, a "demand/distance" conversion factor has to be calculated to translate the excess capacity values into distance values. For example, with a value of 0.5, an excess volume of ten units represents five distance units in the tabu. This value is given by Equation (16). Additionally, two different relocation cost values are taken to determine the impact on the number of relocated points. These values are defined after running tests with the algorithm.

$$FDD = \frac{100 \times Average\ Distance}{Average\ Demand} \tag{16}$$

Table 1
Information of Instances generated for LPMCVRP

Instance	File	Name of Instance	Nodes	Relocation Cost	Demand /
1	testInput	E016-03m.dat	15	2	0.424933
2	testInput2	E016-05m.dat	15	5	0.557267
3	testInput3	E021-04m.dat	20	2	0.619176
4	testInput4	E021-06m.dat	20	5	0.622930
5	testInput5	E022-04g.dat	21	2	0.561785
6	testInput6	E022-06m.dat	21	5	0.747585
7	testInput7	E023-03g.dat	22	2	0.989185
8	testInput8	E023-05s.dat	22	5	0.999746
9	testInput9	E026-08m.dat	25	2	0.530341
10	testInput10	E030-03g.dat	29	2	0.873708
11	testInput11	E030-04s.dat	29	5	0.906092
12	testInput12	E031-09h.dat	30	2	0.506725
13	testInput13	E033-03n.dat	32	5	0.319762
14	testInput14	E033-04g.dat	32	2	0.588160
15	testInput15	E033-05s.dat	32	5	0.618347
16	testInput16	E036-11h.dat	35	2	0.628680
17	testInput17	E041-14h.dat	40	5	0.573641
18	testInput18	E045-04f.dat	44	2	0.887064
19	testInput19	E051-05e.dat	50	5	0.597996
20	testInput20	E072-04f.dat	71	2	0.286026
21	testInput21	E076-07s.dat	75	5	0.608014
22	testInput22	E076-08s.dat	75	2	0.561503
23	testInput23	E076-10e.dat	75	5	0.587890
24	testInput24	E076-14s.dat	75	2	0.633750
25	testInput25	E101-08e.dat	100	5	0.632894
26	testInput26	E101-10c.dat	100	2	0.652071
27	testInput27	E101-14s.dat	100	5	0.571864

Source: Owner

Table 2 shows the results obtained with Algorithm 1 after five repetitions. The best solution is reported for each instance, taking into account a route start cost of zero. Table 2 shows the values of the objective function of the construction and improvement (tabu search) phases for the algorithm without packing considerations. The average solution time is 54.13 seconds having a maximum of 146.5 seconds with 100 customers. The application of the algorithm in two computers of a problem solved in 10 regions (each one with one depot) have 100 customers, that means 1000 customers, the total solution time would be approx.—forty-eight hours (two days), which seems rational for operations planning context.

The number of iterations is parameterized by Equation (17):

$$It = \frac{1{,}000{,}000}{(Number\ of\ Relocation\ Points)^2} \tag{17}$$

Table 2
Performance of algorithm without packing constraints

Instance	Phase I - Constructive	Phase II - Improvement	CPU Time [s]	Relocated Nodes
1	269.32	235.84	15.20	7
2	242.32	242.32	14.80	0
3	335.46	310.65	20.80	10
4	331.89	245.80	22.20	4
5	396.48	349.40	23.80	11
6	335.71	333.81	22.10	0
7	627.17	590.05	23.60	11
8	629.00	600.89	22.90	2
9	449.49	433.02	28.70	8
10	635.03	600.81	34.90	11
11	635.88	590.45	31.80	4
12	443.29	336.49	35.00	17
13	2254.70	2135.30	34.90	16
14	1092.42	1009.95	37.40	13
15	1028.52	991.56	36.80	3
16	452.73	413.23	37.50	13
17	540.64	498.25	47.40	5
18	1011.55	946.51	48.80	10
19	644.17	616.96	55.60	2
20	462.42	446.75	94.80	4
21	889.23	854.95	90.20	2
22	960.50	737.46	102.50	28
23	938.05	882.03	95.00	5
24	909.90	837.82	91.60	33
25	1185.42	1130.30	127.10	7
26	1210.75	1138.06	146.50	16
27	1219.63	1178.51	119.80	5

Source: Owner

Table 3
Performance of MIP and the gap respect to the proposed approach without packing

Instance	Objective Function MIP	Solution Approach	CPU MIP [s]	Delta [%]
1	247.27	235.84	3000	4.83
2	238.67*	242.32	624	-1.51
3	312.26	310.65	1800	0.52
4	304.28	245.80	3600	23.79
5	338.83	349.40	2400	-3.02
6	338.17	333.81	3600	1.31
7	609.23	590.05	1800	3.25
8	600.89	600.89	3600	0.00
9	425.05	433.02	2400	-1.84
10	685.26	600.81	1800	12.37
11	587.03	590.45	3000	-0.58
12	430.73	336.49	3600	28.01
13	2154.14	2135.30	3000	0.88
14	1038.68	1009.95	3000	2.84
15	1039.99	991.56	3600	4.88
16	477.55	413.23	3600	15.56
17	617.46	498.25	3600	23.93
18	1119.44	946.51	3600	18.27
19	791.62	616.96	3600	28.31
20	-	-	-	-
21	-	-	-	-
22	-	-	-	-
23	-	-	-	-
24	-	-	-	-
25	-	-	-	-
26	-	-	-	-
27	-	-	-	-

Source: Owner
*Optimal Solution (Gap = 0.00%)

Table 3 shows the objective function value (Objective) and computing time of the MIP (CPU MIP). The only solution optimal value (Objective) is, for instance, 2. For the other values of the Objective, the MIP cannot obtain the optimal solution (Gap MIP = 0.00%). Also, the delta between the MIP solution (Objective) and the value of phase II of the algorithm (Solution) is calculated. The overall average delta is 8.52%, and, in instances with more than 50 customers, the MIP cannot find the feasible solution, while the algorithm found solutions within 108 seconds on average in these instances. These results suggest the better overall performance of the algorithm regarding time and objective function. It is essential to stand out that in these kinds of problems where the size of the instance is large, the solution time is such important as the objective function. Finally, Table 4 shows the results of the problem considering the packing constraints using the GRASP algorithm. In this case, the number of iterations is parameterized by Equation (18):

$$It2 = \frac{10{,}000}{Number\ of\ Relocation\ Points \times 2} \tag{18}$$

In this case, there are execution times longer than 200 seconds. There are no feasible solutions in eight instances since the packing of the demand of one of the points exceeds the vehicle's capacity.

Table 4

Performance of algorithm with packing constraints

Instance	Phase I - Constructive	Phase II - Improvement Tabu + GRASP	CPU Time [s]
1	344.77	288.91	200.03
2	446.87	346.31	637.85
3	405.66	336.11	1180.12
4	459.14	423.1	1894.81
5	602.53	515.41	669.84
6	528.75	499.4	1131.22
7	1048.16	834.8	1009.33
8	-	-	-
9	727.28	555.56	1147.23
10	-	-	-
11	1266.51	1028.13	1541.85
12	658.29	503.86	616.82
13	3394.21	2555.73	774.64
14	2027.47	1869.38	2548.88
15	2665.46	1839.39	1040.1
16	828.43	661.06	3161.92
17	1006.12	739.03	1352.98
18	1611.44	1448.73	2182.34
19	-	-	-
20	-	-	-
21	-	-	-
22	-	-	-
23	1834.43	1533.57	646.05
24	1325.99	1112.83	2341.73
25	2049.26	1741.34	4179.71
26	-	-	-
27	-	-	-

Source: Owner

This variant implies higher solution times, and it could not work in actual sizes problems like the first shown in this section where a city has 1000 customers and ten regions. The combination with the GRASP increases the time within the proposed approach since it is applied during the whole LPMCVRP.

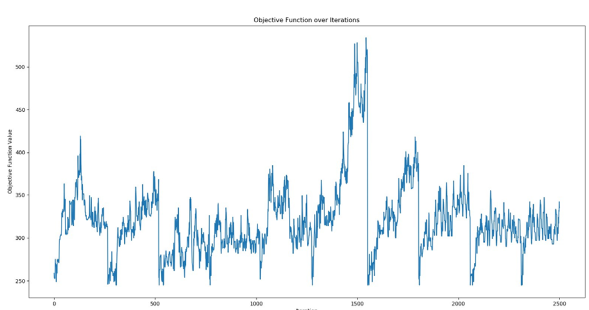

Fig. 7. Behavior of the objective function for the algorithm without packing considerations (Instance 4)

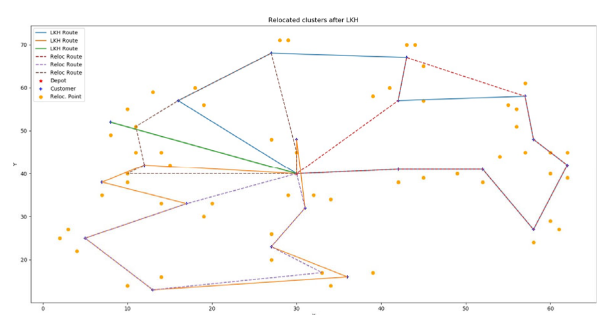

Fig. 8. Final solution of the algorithm without packing considerations (Instance 4)

Source: Owner

Figs. 7-10 show examples of the solutions' behavior in the improvement phase and the final solutions of instance 4 for the algorithms without and with packing constraints. Fig. 7 and Fig. 9 show the behavior of the proposed algorithm's objective function with and without packing constraints. Every certain number of iterations, the effect of the strategy of the flexibilization of the tabu list resetting is visible. Fig. 9, which corresponds to the case with packing constraints, shows random behavior for the first half of the iterations, and subsequently, a clear pattern of what could be a local optimum is observed. This behavior is recurrent in smaller instances (less than or equal to 25 nodes).

Fig. 8 and Fig. 10 present the graph of final solutions with two variants where the solid lines are the output from the LKH and then pointed lines are the final ones. The improvement generated by the operators are shown on the graphs where new clusters are built, and relocations are done to fix route crosses and reduce total traveled distance. On the other hand, the packing changes completely the final solution, causing different clusters and relocations in the same instance 4, confirming that this problem increases the complexity very much and that the first variant is more applicable in the real size problems.

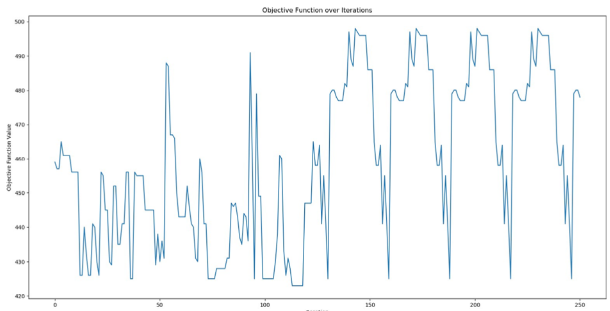

Fig. 9. Behavior of the objective function for the algorithm with packing considerations (Instance 4)

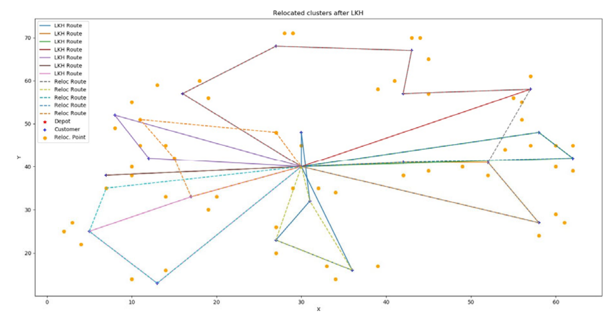

Fig. 10. Final solution of the algorithm with packing considerations (Instance 4)

Source: Owner

5. Conclusions and future work

This work introduces a new variant of the MCVRP that considers the rearrangement or relocation of collection points and the demand's packing. The problem is called the LPMCVRP and is applied in a recyclable waste collection system in vehicles with flexible compartments. A two-phase metaheuristic algorithm is proposed based on a tabu search without packing considerations and a variant that integrates the tabu search and the GRASP solution strategy considering packing restrictions to solve the LPMCVRP.

A large tour is generated in the construction phase using the LKH heuristic with all the collection points. Subsequently, the scanning algorithm can define candidate clusters, which are assigned by solving a set-partitioning problem. Finally, the solution is improved with a local search with two inter-route and two intra-route operators. In the improvement phase, a modified tabu search is executed with the operators mentioned above and two flexibilization strategies that, to a great extent, prevent the exploration of solutions that, for many iterations, lead to worse solutions and deviate from local optima. A GRASP algorithm that validates the packing of the bin demand with volumetric considerations is introduced.

Due to all the above, the problem presented in this study makes a significant contribution to the literature since formal definitions of a problem where relocation, routing, and packing are considered have not been proposed for recyclable waste collection systems and cargo transportation. In addition to this, a set of instances is generated for future investigations of the problem.

As future work, the implementation of better strategies for exploring new edges in the tabu search is proposed to make the metaheuristics more efficient and the generation of instances with several more collection points to approximate the behavior of real cases. Another future direction is the reduction of the computational time through the use of granularity strategies such as those proposed in Escobar et al. (2013), Escobar et al. (2014a), Escobar et al. (2014b), Escobar & Linfati (2012), and Linfati et al. (2014). Besides, the development of an automated decision-making system (decision support system) that visualizes the progress of the algorithm, as well as the routes, relocations, and packing; that allows the input parameters of the model, such as the current location of the collection points and the relocation costs, to be changed; and that allows the generation of new relocation points is proposed.

References

Abdulkader, M. M., Gajpal, Y., & ElMekkawy, T. Y. (2015). Hybridized ant colony algorithm for the multi compartment vehicle routing problem. *Applied Soft Computing, 37*, 196-203.

Adil, N., & Lakhbab, H. (2020). A Discrete Bat Algorithm for the Multi-Compartment Vehicle Routing Problem. *In 2020 IEEE 6th International Conference on Optimization and Applications (ICOA)*, pp. 1-5.

Bernal-Moyano, J. A., Escobar, J. W., Marín-Moreno, C., Linfati, R., & Gatica, G. (2017). A comparison of trajectory granular based algorithms for the location-routing problem with heterogeneous fleet (LRPH). *Dyna, 84*(200), 193-201.

Bernal, J., Escobar, J. W., Paz, J. C., Linfati, R., & Gatica, G. (2018). A probabilistic granular tabu search for the distance constrained capacitated vehicle routing problem. *International Journal of Industrial and Systems Engineering, 29*(4), 453 – 477.

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

Capelle, T., Cortés, C. E., Gendreau, M., Rey, P. A., & Rousseau, L. M. (2019). A column generation approach for location-routing problems with pickup and delivery. *European Journal of Operational Research, 272*(1), 121 – 131.

Caramia, M., & Guerriero, F. (2010). A milk collection problem with incompatibility constraints. *Interfaces, 40*(2), 130-143.

Chávez, J., Escobar, J., & Echeverri, M. (2016). A multi-objective Pareto ant colony algorithm for the Multi-Depot Vehicle Routing problem with Backhauls. *International Journal of Industrial Engineering Computations, 7*(1), 35 – 48.

Chen, J., & Shi, J. (2019). A multi-compartment vehicle routing problem with time windows for urban distribution–A comparison study on particle swarm optimization algorithms. *Computers & Industrial Engineering, 133*, 95-106.

Coelho, L. C., & Laporte, G. (2015). Classification, models and exact algorithms for multi-compartment delivery problems. *European Journal of Operational Research, 242*(3), 854-864.

Coffman, E. G., Csirik, J., Galambos, G., Martello, S., & Vigo, D. (2013). Bin packing approximation algorithms: survey and classification. *In Handbook of combinatorial optimization, pp. 455-531*. Springer New York.

Dai, Z., Aqlan, F., Gao, K., & Zhou, Y. (2019). A two-phase method for multi-echelon location-routing problems in supply chains. *Expert Systems with Applications, 115*, 618-634.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science, 6*(1), 80 – 91.

Dereli, T., & Daş, G. S. (2010). Development of a decision support system for solving container loading problems. *Transport, 25*(2), 138-147.

Dukkanci, O., Kara, B. Y., & Bektaş, T. (2019). The green location-routing problem. *Computers & Operations Research, 105*, 187 – 202.

El Fallahi, A., Prins, C., & Calvo, R. W. (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research, 35*(5), 1725-1741.

Escobar, J. W., & Linfati, R. (2012). Un algoritmo metaheurístico basado en recocido simulado con espacio de búsqueda granular para el problema de localización y ruteo con restricciones de capacidad. *Revista Ingenierías Universidad de Medellín, 11*(21), 139-150.

Escobar, J. W., Linfati, R., & Toth, P. (2013). A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. *Computers & Operations Research, 40*(1), 70 – 79.

Escobar, J. W., Linfati, R., Toth, P., & Baldoquin, M. G. (2014a). A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem. *Journal of Heuristics, 20*(5), 483 – 509.

Escobar, J. W., Linfati, R., Baldoquin, M. G., & Toth, P. (2014b). A Granular Variable Tabu Neighborhood Search for the capacitated location-routing problem. *Transportation Research Part B: Methodological, 67*, 344 – 356.

Escobar, J. W., Linfati, R., & Adarme-Jaimes, W. (2015). A hybrid metaheuristic algorithm for the capacitated location routing problem. *Dyna, 82*(189), 243-251.

Escobar-Falcón, L. M., Álvarez-Martínez, D., Granada-Echeverri, M., Escobar, J. W., & Romero-Lázaro, R. A. (2016). A matheuristic algorithm for the three-dimensional loading capacitated vehicle routing problem (3L-CVRP). *Revista Facultad de Ingeniería Universidad de Antioquia, 78*, 09-20.

Farrokhi-Asl, H., Makui, A., Jabbarzadeh, A., & Barzinpour, F. (2020). Solving a multi-objective sustainable waste collection problem considering a new collection network. *Operational Research*, *20*(4), 1977-2015.

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research, 22*(2), 340 – 349.

Golden, B. L., Magnanti, T. L., & Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks, 7*(2), 113 –148.

Groër, C., Golden, B., & Wasil, E. (2010). A library of local search heuristics for the vehicle routing problem. *Mathematical Programming Computation, 2*(2), 79-101.

Gruler, A., Juan, A. A., Contreras-Bolton, C., & Gatica, G. (2015). A biased-randomized heuristic for the waste collection problem in smart cities. *In International Forum for Interdisciplinary Mathematics, pp. 255-263*. Springer, Cham.

Helsgaun, K. (2017), An Extension of the Lin-Kernighan-Helsgaun TSP Solver for Constrained Traveling Salesman and Vehicle Routing Problems. Technical Report, Roskilde University

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research, 126*(1), 106-130.

Henke, T., Speranza, M. G., & Wäscher, G. (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research, 246*(3), 730 – 743.

Herrera, J. S. (2020a). Instances and Results of LRP Algorithm. url: https://github.com/sebashc3712/Instances_and_Results_of_LRP_Algorithm (visited 27-06-2020).

Herrera, J. S. (2020b). LRP Algorithm with GRASP. url: https://github.com/sebashc3712/LRP_ Algorithm (visitado 27-06-2020).

Herrera, J. S. (2020c). LRP Algorithm without GRASP. url: https://github.com/sebashc3712/LR_ Algorithm (visitado 27-06-2020).

Hübner, A., & Ostermeier, M. (2019). A multi-compartment vehicle routing problem with loading and unloading costs. *Transportation Science, 53*(1), 282-300.

Kaya, O., & Ozkok, D. (2018). A network design problem with location, inventory and routing decisions. *In Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 139-140.

Linfati, R., Escobar, J. W., & Cuevas, B. (2014). An algorithm based on granular tabu search for the problem of balancing public bikes by using multiple vehicles. *Dyna, 81*(186), 284-294.

Linfati, R., & Escobar, J. W. (2018). Reoptimization Heuristic for the Capacitated Vehicle Routing Problem. *Journal of Advanced Transportation*, 1 – 8.

Lodi, A., Martello, S., & Monaci, M. (2002). Two-dimensional packing problems: A survey. *European Journal of Operational Research, 141*(2), 241-252.

Martínez, D. A., Alvarez-Valdes, R., & Parreño, F. (2015). A grasp algorithm for the container loading problem with multi-drop constraints. *Pesquisa Operacional, 35*(1), 1-24.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., & Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research, 37*(11), 1886-1898.

Prins, C., Prodhon, C., Ruiz, A., Soriano, P., & Wolfler Calvo, R. (2007). Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic. *Transportation Science, 41*(4), 470 – 483.

Raft, O. M. (1982). A modular algorithm for an extended vehicle scheduling problem. *European Journal of Operational Research, 11*(1), 67 – 76.

Reed, M., Yiannakou, A., & Evering, R. (2014). An ant colony algorithm for the multi-compartment vehicle routing problem. *Applied Soft Computing, 15,* 169-176.

Silvestrin, P. V., & Ritt, M. (2017). An iterated tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research, 81*, 192-202.

Vincent, F. Y., Lin, S. W., Lee, W., & Ting, C. J. (2010). A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering, 58*(2), 288 – 299.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research, 183*(3), 1109-1130.

Wheeler, K. (2013). The Largest Environmental Movement: Recycling and Consumption Work in Sweden.

Wren, A., & Holliday, A. (1972). Computer scheduling of vehicles from one or more depots to a number of delivery points. *Journal of the Operational Research Society, 23*(3), 333 – 344.