Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

# A specialized genetic algorithm for the fuel consumption heterogeneous fleet vehicle routing problem with bidimensional packing constraints

**Luis Miguel Escobar-Falcón[a*], David Álvarez-Martínez[b], John Wilmer-Escobar[c] and Mauricio Granada-Echeverri[d]**

[a]*Universidad Libre Seccional Pereira – Integra S.A., Colombia*
[b]*Universidad de Los Andes, Colombia*
[c]*Department of Accounting and Finance, Universidad del Valle, Colombia*
[d]*Universidad Tecnológica de Pereira, Colombia*

CHRONICLE

*Article history*:
Received July 14 2020
Received in Revised Format
November 24 2020
Accepted November 24 2020
Available online
November, 24 2020

Keywords:
*2L-FHFVRP*
*2L-HFVRP*
*Elitist Genetic Algorithm*
*GRASP*
*Sequential Loading*

ABSTRACT

The vehicle routing problem combined with loading of goods, considering the reduction of fuel consumption, aims at finding the set of routes that will serve the demands of the customers, arguing that the fuel consumption is directly related to the weight of the load in the paths that compose the routes. This study integrates the Fuel Consumption Heterogeneous Vehicle Routing Problem with Two-Dimensional Loading Constraints (2L-FHFVRP). To reduce fuel consumption taking the associated environmental impact into account is a classical VRP variant that has gained increasing attention in the last decade. The objective of this problem is to design the delivery routes to satisfy the customers' demands with the lowest possible fuel consumption, which depends on the distances of the paths, the assigned vehicles, the loading/unloading pattern and the load weight. In the vehicle routing problem literature, the approximate algorithms have had great success, especially the evolutionary ones, which appear in previous works with quite a sophisticated structure, obtaining excellent results, but that are difficult to implement and adapt to other variants such as the one proposed here. In this study, we present a specialized genetic algorithm to solve the design of routes, keeping its main characteristic: the easy implementation. By contrast, the loading of goods restriction is validated by means of a GRASP algorithm, which has been widely employed for solving packing problems. With a view of confirming the performance of the proposed methodology, we provide a computational study that uses all the available benchmark instances, allowing to illustrate the savings achieved in fuel consumption. In addition, the methodology suggested can be adapted to the version of solely minimizing the total distance traveled for serving the customers (without the fuel consumption) and it is compared to the best works presented in the literature. The computational results show that the methodology manages to be adequately adapted to this version and it is capable of finding improved solutions for some benchmark instances. As for future work, we propose to adjust the methodology to consider the three-dimensional loading problem so that it adapts to more real-life conditions of the industry.

## 1. Introduction

The delivery of goods to geographically dispersed customers plays a crucial role in the management of distribution and logistics, since it stands for a significant proportion of the operating expenses of the companies, mainly because of the high consumption and cost of fuel required for the whole operation. The delivery problem which has attracted the greatest attention is known as the vehicle routing problem (VRP) and it describes the distribution of goods from a main depot to a set of

* Corresponding author Tel.: +57 3217938145
E-mail: luism.escobarf@unilibre.edu.co (L. M. Escobar-Falcón)

2021 Growing Science Ltd.
doi: 10.5267/j.ijiec.2020.11.003

customers served by vehicles. The VRP was first introduced by Dantzig and Ramser (1959), which dealt with the delivery of fuel by trucks from a bulk terminal to a large number of service stations. Since the publication of this work, a great deal of considerations have been taken into account to tackle other industry problems. The original version of the problem merely considered to minimize the total distance traveled, but several studies have demonstrated that this objective is not aligned with the concerns of companies today. One of the most studied variants in the last decade is the one which reduces fuel consumption, given that not only does it represent a decrease in considerable costs for the company but it also begins to address the concerns of having a cleaner operation, that is, generating the least possible environmental impact Toth and Vigo (2014). It is known that fuel consumption is directly related to greenhouse gas emissions (GHG), especially the CO2 ones Demir et al. (2014). Conversely, due to advancements in more sophisticated computational systems and the development of more potent optimization techniques, it has been possible to solve problems that are closer to real-world conditions in the industry. One example is the solution of the integrated problem of loading and vehicle routing, which had been considered separately for more than 50 years. In the majority of these works, the loading was simplified in such a way that its geometric shape was ignored and only one dimension (in general, the volumetric one) was considered. In practice, should solve transport and loading problems at the same time, especially if the loading is not trivial (a single type of product, bulk or liquid, usually). For example, when an item is stacked on top of others, all these aspects should be carefully considered: the stacking stability, the frailty and how easy the unloading is for the operators. Thus, the solution for each individual problem turns out to be less practical due to the fact that, the solution may become infeasible when carrying out the task. For example, for an established route, the actual loading and unloading operations do not allow the packing of the entire customers' demand.

Integrated vehicle routing and loading problems such as Capacitated Vehicle Routing Problem with Two and Three Dimensional Loading Constraints (2L-CVRP and 3L-CVRP) have been studied and obtained satisfactory results by Iori et al. (2007) and Gendrau et al. (2006). The majority of previous works did not consider a heterogeneous vehicle fleet, since it is not common in practice. However, a heterogeneous fleet offers the flexibility to design a more profitable distribution plan and it respects the fleet upgrades throughout time.

This work concerns a practical VRP application. In this problem, the depot has a heterogeneous vehicle fleet, the customers' demands are composed of a set of rectangular items that represent non-stackable objects in practice, with known lengths, widths and weights. We then seek to select a set of vehicles and determine the route of each vehicle to serve all customers, aiming at minimizing fuel consumption used on the routes added to fixed and variable costs of the selected vehicles. Fuel consumption emerges as a factor on the sum of edges that form a trip made by the vehicle, associating the cargo transported in each edge with the loading capacity of the vehicle. While the fixed costs are associated with the vehicle selection, the variable costs depend on the edges traveled (distance between customers) to cover the route with the type of vehicle selected. The types of vehicles are heterogeneous regarding the capacity, the space and the fixed and variable costs of the loading. The number of units of each type of vehicle is unlimited. One route is only deemed to be feasible when there is a viable loading (unloading) plan for the demanded items by the customers on this route, that is, the unloading of goods from a customer upon arriving at its destination should be done without reordering of the remaining boxes, which is known in the literature as sequential loading constraint or LIFO (Last-in First-out) or multi-drop.

The 2L-FHFVRP is also of great interest in the theoretical research. Considering that it is a combination of two well-known NP-hard problems: the Heterogeneous Fleet Vehicle Routing Problem (HFVRP) and the Two-dimensional Bin Packing Problem (2D-BPP). Iori et al. (2007) shows that the integrated problem is at least as complex as one of these separately, i.e., the 2L-FHFVRP is NP-hard, too.

To solve this problem, we present an elitist-based genetic algorithm Chu and Beasley (1996). We adapted and extended the chromosome representation (solution) from a data structure proposed by Liu et al. (2009). The data structure consists in a one-dimensional array to register the order of customers to be served and employs a directed graph to generate a set of possible routes given by that order.

The initial population makes use of the heuristic solution associated to the TSP over the original nodes. The tournament selection is carried out, where two winning chromosomes are recombined through five classical operators. Next, we compute the fitness associated with each of these elements, including the validation of the two-dimensional packing constraints, which penalize the offspring that do not satisfy them. Lastly, we present the new individuals to the population. This procedure is repeated until the maximum number of generations criterion is met. Due to the elitist structure of the algorithm, two populations are required during the optimization process.

In particular, the algorithm that checks the two-dimensional packing constraints is Greedy Randomized Adaptive Search Procedure (GRASP) based procedure which, in its construction phase, uses the maximum space concept to represent the packed items and the largest remaining available spaces, pseudo-randomly chooses among criteria such as grouping items by customers or taking all items to the rear of the vehicle. By allowing infeasible solutions, the proposed methodology manages to stand out from previous work which merely dealt with feasible search spaces.

The devised encoding allows to reunite the assignments of all types of vehicles required by the chromosome in the direct graph and dissociates the packing metaheuristic by calling it only when there is a solution to the shortest route problem (which defines the sequences of customers to be visited). To demonstrate the efficiency of the proposed methodology, we solved all the benchmark instances, achieving solutions in a reasonable CPU time. In addition, the genetic algorithm was compared with previous works on the 2L-HFVRP (without considering fuel consumption) and the results demonstrate that the proposed algorithm has an acceptable performance in relation to the best works published in the literature.

The remainder of this paper is organized as follows. Section 3 formally describes the problem. Section 4 provides a brief literature review on the problem. Section 5 presents the proposed methodology, detailing both the data structure and the optimization algorithms. Section 6 contains the performed computational study. Finally, Section 7 offers concluding remarks and future work of this study.

## 2. Problem Description

The heterogeneous fleet vehicle routing problem with two-dimensional packing constraints consists in serving, from a depot, the demands of various customers who are geographically dispersed, in order to minimize fuel consumption added to fixed and variable costs. It is also important to consider the load capacity or the maximum weight each one of the employed vehicles support and place the demand (items) in such a way that, when visiting the customers, it is not necessary to rearrange the corresponding load to subsequent customers.

The problem can be formally described as follows. The 2L-FHFVRP is defined over a complete graph $G = (V, E)$, where $V = \{0, 1, 2, \dots, n\}$ is a set of vertices (vertix 0 represents the depot and the remaining ones are the customers) and $E = \{e_{ij}: i, j \in V\}$ the set of edges. Each edge $e_{ij} \in E$ has an associated distance. A fleet of $M$ different types of vehicles is located in a depot and a very large number of vehicles for each type is assumed. Each vehicle has an associated capacity $Q_t$, a fixed cost $F_t$, a variable cost $V_t$, a length $L_t$, a width $A_t$ and a weight $W_t$ (where $t = 1,2, \dots, M$). It is assumed that a vehicle with greater capacity has both higher costs and fuel consumption. Thus, if $Q_1 \leq Q_2 \leq \cdots \leq Q_M$, so $F_1 \leq F_2 \leq \cdots \leq F_M$ and $V_1 \leq V_2 \leq \cdots \leq V_M$. The variable costs of transportation for each edge $e_{ij} \in E$ for each vehicle $t$ are $C_{ijt} = V_t * d_{ij}$. Therefore, the transportation costs (not only the fixed but also the variable ones) of a route $r$ for the vehicle type $t$ are $C_r = F_t + \sum_{(i,j)\in R} C_{ijt}$, where $R$ represents the set of edges that form the route $r$ ($R = \{(0, i), (i, j), \dots, (k, 0)\}$). The cost of fuel consumption for each arc $(i, j)$ incurred by a vehicle type $t$ transporting a load with weight $q$ is $C_{fc} = fc(q) \cdot C_{ijt}$.
It is worth mentioning that, for the same arc, the fuel consumption differs depending on the amount of load transported and the vehicle that transports it. Similarly to Zhang et al. (2015), $fc(q)$ is defined as $\frac{q}{Q_t}$, where $q$ is the sum of weights of the customers' demands being transported. Therefore, the objective function of the problem is presented in Equation (1), which consists of both transport and fuel consumption costs.

$$Z = C_r + C_{fc} \tag{1}$$

In contrast, each customer $c$ (where, $c = 1, 2, \dots, n$) requires a set of $B_c$ rectangular items, with a total number of items $|B_c|$ and a total weight equal to $D_c$. Each item $I_{cb} \in B_c$ ($b = 1,2, \dots, |B_c|$) has a specific length $l_{cb}$ and a width $w_{cb}$. In the 2L-FHFVRP, the vehicle routes and packing of the goods must satisfy the following constraints:

- All routes must start and end at the depot.
- All items of a customer must be loaded in the same vehicle, that is, the customers must be served only once.
- All items must be loaded with their sides parallel to the walls of the vehicle.
- The capacity, length and width of the vehicle cannot be exceeded by the load.
- The load cannot overlap each other.
- Each item can be positioned inside the vehicle, so that, upon arrival at its destination, the vehicle can be unloaded without the need to reorganize the items of the other customers.

In this way, the 2L-FHFVRP seeks to select a set of vehicles to perform the routes respecting the assignment of customers to them so as to minimize Eq. (1), subject to the aforementioned set of constraints.

## 3. State of the Art

A starting point for the vehicle routing problem considering packing constraints is the state of the art review carried out by Iori and Martello (2013), which was improved by Pollaris et al. (2015). After studying the previous work, it is necessary to highlight the relevance of solving the vehicle routing problem combined with the loading of the goods, aiming at optimizing real-life objectives of the industry. To this end, the authors recommend reading Côté et al. (2017). To the best of our knowledge there are no studies on the 2L-FHFVRP prior to the publication date of this work, we decided to focus the literature review on the 2L-HFVRP, whose only difference is the objective function to be dealt. The first work reported in the literature presents a simulated annealing algorithm, which obtains reasonable results and was put forward by Leung et al. (2013). In this study, the 2L-CVRP and the HFVRP are also solved in a satisfactory way. On the other hand, Domínguez et al. (2016a) and Domínguez et al. (2016b) suggested ILS-based algorithms that manage to outperform the best results reported in the literature. These works are relevant as they highlight the importance of considering real-world conditions of the goods transport operations. For this purpose, the authors conducted a study on the different scenarios evaluating features such as: item rotation, sequential loading/unloading and the use of a heterogeneous fleet of vehicles.

Considering the fuel consumption within the objective function enables one to generate a set of routes in which less loaded vehicles travel longer distances (refer to Fig. 1). This variant is known in the literature as vehicle routing problems considering environmental impact, was formalized by (Suzuki, 2011) and (Bektaş & Laporte, 2011). (Xiao et al., 2011) are the ones who first relate the weight load to the distance traveled, with fuel consumption within the cost function. Following this, (Zhang et al., 2015) simplified this relation, thus establishing an objective function that seeks to minimize the transportation costs as well as the fuel consumption ones.

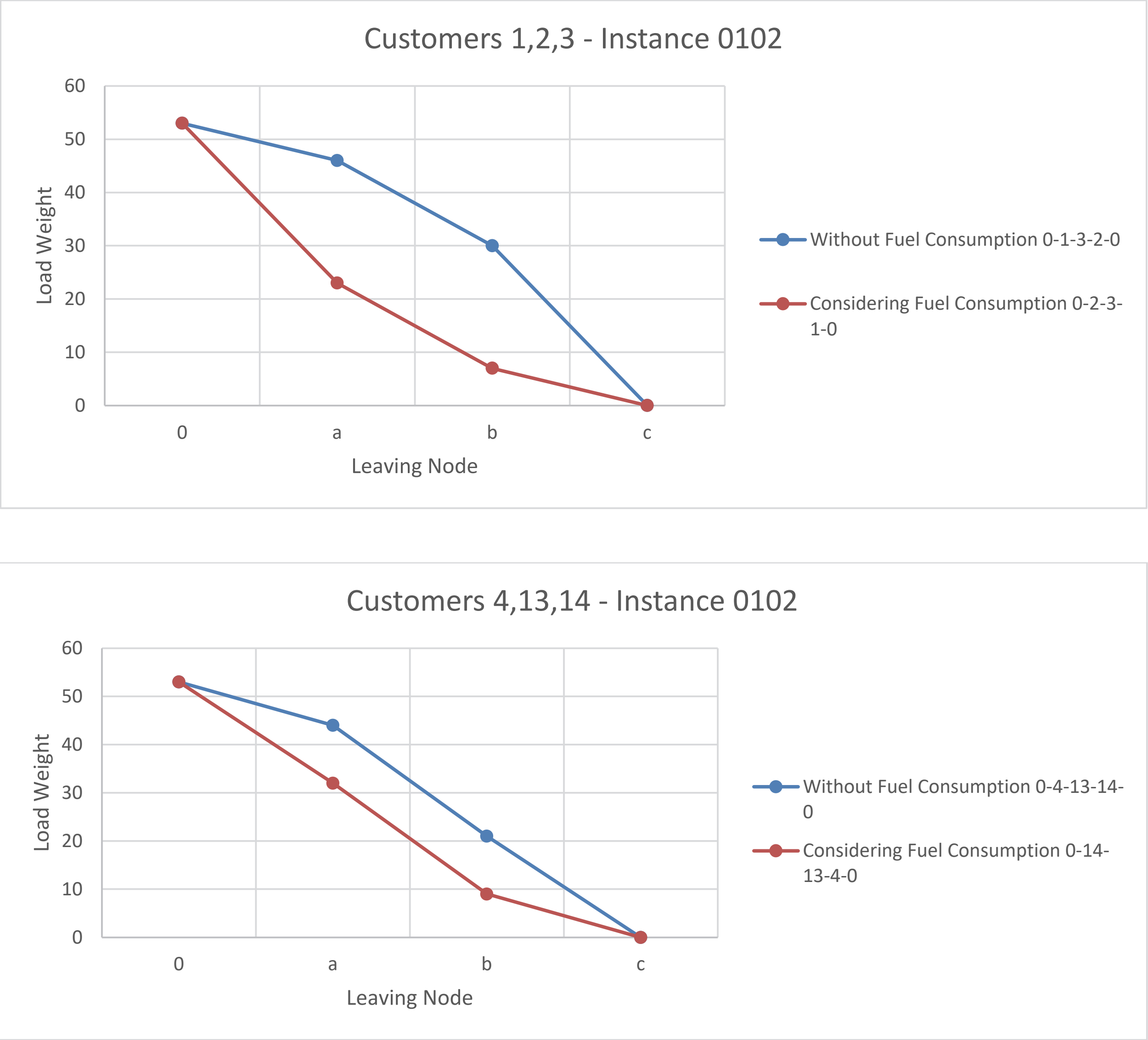

**Fig. 1.** Incidence of environmental impact, from 2L-HFVRP to 2L-FHFVRP.

By reviewing the works that integrate vehicle routing and packing of goods under practical conditions, one can find that assuming a demand or deterministic travel times are strong assumptions that affect the performance of the solutions in real-life applications. It is worth mentioning that Guimarans et al. (2016) and Guimarans et al. (2018) included uncertainty characteristics in the analysis, bringing them even closer to real-world vehicle routing problems. Recent studies considered such characteristics in travel times and load demand in order to combine them with distinct fleet capacities and establish an objective function closer to the current concerns of carriers, as found in Micheli and Mantella (2018).

The environmental variant of this study is known as Vehicle Routing Problem with Fuel Consumption, which seeks to reduce fixed and variable costs associated with the routes, added to those generated by fuel consumption. This version aligns with current concerns of the companies. On the other hand, in the literature one can find studies of the problem considering three-dimensional packing constraints. The most relevant work in this direction is the evolutionary local search presented by Zhang et al. (2015), which is capable of solving in a satisfactory fashion the combined problem of routing and packing with fuel consumption minimization. This study aims at extending the work proposed by Zhang et al. (2015) by considering the existence of a mixed fleet of vehicles in the context of non-stackable load.

## 4. Solution Methodology

In this study, we propose a metaheuristic algorithm by trying to exploit the ease of implementing this type of methodology. We adapt the structure of the genetic algorithm (GA) presented by Chu and Beasley (1996), and we incorporate a GRASP algorithm based on different works published in the field of cutting and packing. It is worth mentioning that the base of both algorithms is the encoding employed: for the GA, we adapted the *split* procedure, whereas for the GRASP, we adapted the concept of maximum residual space.

### *4.1 Encoding*

The encoding devised by Prins (2004) consists of a vector that represents the order of customer service, which uses an auxiliary directed graph, to create all the possible trips following the established order, assuring the capacity constraint of the available vehicle. Considering that such graph comprises all the feasible routes given an order, we must then select the best set of routes to visit all customers at a lower cost, which is achieved by solving a Shortest Path Problem. This encoding was extended by Liu et al. (2009), making it possible to consider a heterogeneous fleet of vehicles. In this work, there are adaptations to also represent the packing constraints, seeking to perform, in the creation and updating of the digraph, the filtering of trips that are infeasible by sequential packing in the different vehicles present in the fleet.

Hence, the construction of the digraph will consist of the creation of the arcs corresponding to the trips that can be made considering only the capacity constraint (supported weight) of the distinct vehicles. Given that the fixed and variable transportation costs directly relate to the vehicle capacity, it is not necessary to create the same trips (arcs) to large-sized vehicles if a smaller sized vehicle can perform this trip. Therefore, not considering the packing constraints in the creation requires verifying if the solution to the shortest path problem does not contain infeasible trips. To this end, we will use a GRASP algorithm, which will be described later. Thus, each one of the solution arcs must be validated, in case it is not packable, large-sized vehicles are tested until the available fleet is depleted, but if it is not possible to find a vehicle to transport the goods, this arc must be eliminated from the digraph and a shortest path problem must be solved again.

It is important to mention that the resulting auxiliary graph is a directed acyclic graph (DAG), which allows for using the version of the algorithm of Bellman-Ford for DAG Ahuja et al. (1988), in order to calculate the shortest path in an efficient way. Due to the possibility of obtaining infeasible arcs in terms of packing, the Bellman-Ford algorithm will be called iteratively. In this manner, all the infeasible arcs by packing are deleted from the auxiliary digraph, and the Bellman-Ford algorithm is called with the updated graph until a feasible solution is reached. This coding scheme offers great flexibility to tackle multiple variants of the problem. The selection of the arcs that will compose the digraph and the information that will be associated with each one of these arcs provides the possibility of representing variants that allow for stacking (three-dimensional packing), multiple depots, homogeneous or heterogeneous fleet, time windows, among others.

The main objective of this work is to improve the vehicle routing problems and packing, including the minimization of fuel consumption. In terms of encoding, the environmental impact will be reflected in the cost of the arcs of the auxiliary graph. Figure 2.a illustrates an instance with five dispersed customers. The proposed encoding then requires to store the order of customer service in an array. Figure 2.b shows an example of the order of service of these five customers. In order to find the set of feasible routes, it is necessary to create an auxiliary graph, using the available distinct vehicles. Figure 2.c depicts an example of graph for two types of vehicles (V1 and V2): a dotted arc represents a trip made by a vehicle type V1, whereas the continuous blue arcs represent the trips made in vehicles type V2. On the other hand, in Figure 2.c, the green dotted lines correspond to the shortest path, as a result of executing the Bellman-Ford using as the cost of the arcs the cost of each trip given by Equation (1). This green dotted solution is the one that will enter the iterative scheme of verifying the packing constraints for each trip. When all resulting arcs are valid, a feasible solution will be finally obtained, given the order of customer service, as shown in Figure 2.d, where two trips are performed: the first one in vehicle V1, serving customers B and C, while the second one in vehicle V2 visiting customers A, E and D.

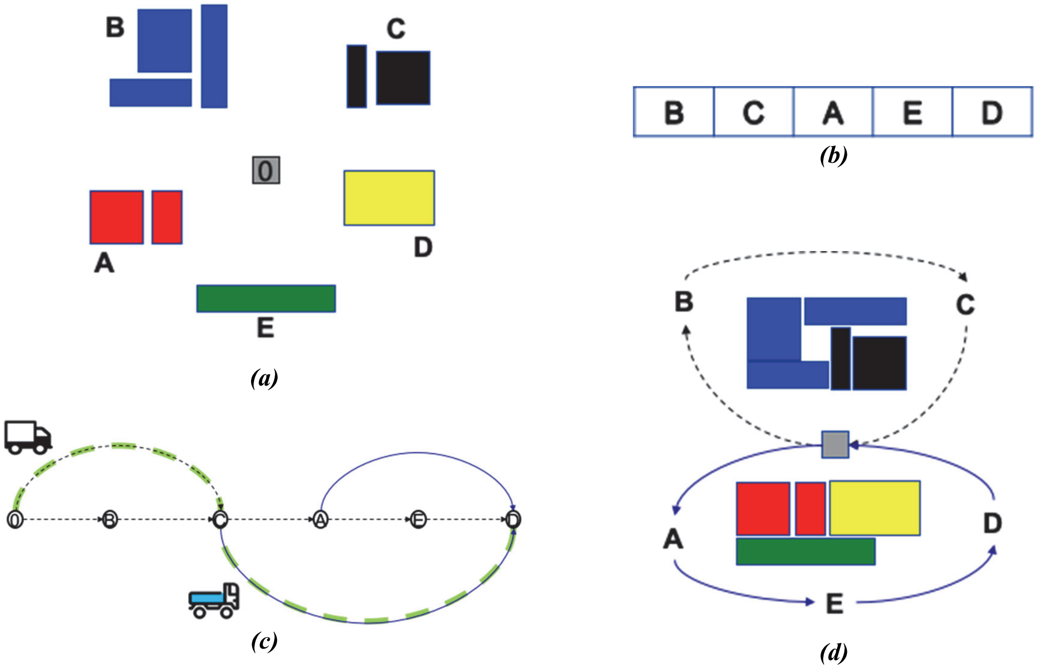

**Fig. 2.** Encoding and decoding of the solution. **a.** Set of customer locations and items demanded by the customers. **b.** Example of order of customer service. **c.** Auxiliary Graph and the shortest path. **d.** Set of routes and packing within them.

*5.2 GRASP Algorithm for Packing*

In this work, we adapted a GRASP algorithm presented in Álvarez-Martínez et al. (2015) that allows for obtaining, in its construction phase, feasible solutions through the control of the positioning of items inside the vehicle, thus satisfying the sequential loading constraint. The GRASP algorithm was developed by Feo and Resende (1989) to solve difficult combinatorial optimization problems. GRASP is an iterative procedure that combines a construction phase with an improvement phase. In the construction phase, a solution is built gradually, by adding elements to it. This phase must be iterative, greedy, random and adaptive. The search strategy proposed in this algorithm enables, in the construction phase, the random choice of items to be located in each empty space, while in the local search phase, emptying and filling movements are performed.

The Algorithm 1 proposed begins by selecting one of the available empty spaces ($s'$) from the list $S$, then the list of layers of items ($L$) that fit in the space $s'$ is generated. Next, this list $L$ is reduced to the Restricted Candidate List ($RCL$) and from this reduced list one of its elements ($C$) is randomly selected. This layer $C$ is located, thus generating the pattern $P$ and imposing to update the empty space list as well as the list of the remaining items ($S$, $B_j$, and $B$, respectively). If when finishing to pack the items of a customer there are still empty spaces and customers to pack, the empty space list must be updated by customer change, in which the spaces that violate the sequential loading constraints must be eliminated. When we have finished trying to assign all the items demanded by the clients of the route, we analyze if the objective function of the obtained pattern is of good quality, that is, if this solution built is promising. Through the *ImproveBuiltPattern(P)* function, items will be randomly removed and an attempt will be made to refill the vehicle by eliminating random choices from the search scheme. Finally, we verify if all the demanded items have been packed, returning true if it is feasible. In case the iterations run out and in no attempt it was possible to pack all the items, the algorithm will return infeasible.

For more details on the construction phase and randomization that are key to understanding the packing algorithm, we refer to Álvarez-Martínez et al. (2015). The constructive algorithm of this work is based on the construction phase proposed by Escobar et al. (2015). Both studies share the characteristic of having cargo with multiple destinations, which requires special treatment for update structures of the remaining (empty) spaces. This part is laborious given that empty spaces must be verified in order to guarantee the sequential loading, which implies that some of them must be deleted or reduced. The constructive procedure uses the maximum empty spaces to represent the available area of the vehicle. In this case, each selected item is packed in an empty space and this generates at most two new maximum spaces (see Fig. 3). The constructive algorithm uses a list *S* of the maximum spaces and a list $B_j$ of the boxes of customer *j* that still have to be packed. The steps 0-4 summarize the constructive algorithm.

```
Algorithm 1 GRASP - Packing Validation
Data: Route, Vehicle, Customers, Max_Iterations
Result: feasible or unfeasible
for $i \leftarrow 1...Max\_Iterations$ do
    List $S = Vehicle.Dimensions$
    List $B = Customers.Pieces$
    for $i \leftarrow 1...Max\_Iterations$ do
        List $B_j = Customers(j).Pieces$
        while $B_j \neq \emptyset$ and $S \neq \emptyset$ do
            Space $s' = SelectSpace(S)$
            List $L = GenerateLayerList(s', B_j)$
            List $RCL = BuildRCL(L, \delta)$
            Layer $C = RandomLayerSelection(RCL)$
            Pattern $P = LocateLayer(C, s', P)$
            List $S = UpdateMaximalSpacesList(C, S)$
            Lists $B_j, B = UpdateListRemainingItems(C, B_j, B)$
        end
        List $S = UpdateListCustomersSpacesSwap(S)$
        if $IsPromising(P)$ then
            Pattern $P = ImproveBuiltPattern(P)$
        end
        if $B = \emptyset$ then
            return $true$
        end
    end
end
return $false$
```

*Step 0:* Initialization. A list of the maximum empty spaces is created and initiated with the vehicle space (area) to travel the route. Be the lists $B_1, B_2, ..., B_n$ the sets of items of each customer to be served in the route and the list *B* the set of all demanded items.

*Step 1:* Choose a maximum space of *S*. Given that the list *S* contains the largest empty rectangles available and that there are many candidates (items to be packed), one must determine a selection mechanism based on some quality criterion or packing strategy. In this work, two selection criteria are proposed: choose the maximum space with minimum distance to the bottom of the vehicle and choose the maximum space with greater area. In addition, the rear corner closest to one of the rear corners of the vehicle is selected from the chosen space, as an anchor point (or reference) when locating the items in the empty space.

*Step 2:* Choose the items to be packed. Once the maximum space s' has been selected, each box *k* that fits inside $s'$ must be taken from the ordered list $B_i$ (where *i* is the current customer to be packed). If there are many copies of box *k*, each one of the possible layers must be generated. This consists of packing items in arrays of columns or rows combining the different axes. As in Escobar-Falcón et al. (2016), two criteria are considered to select one of the configurations of the items (boxes). Choose the layer of boxes that produces the largest increase in the objective function (maximum area) or choose the layer of boxes that best fit in the maximum space (best-fit). The latter is a criterion in which the distances between the sides of the layer of boxes and the maximum space are computed, so that the distances of each configuration are sorted in non-decreasing manner and select the configuration with the shortest distance (the one that best fits the space).

*Step 3:* Update the list S. Unless the box (or the layer) fits exactly in the space $s'$, the packing process will produce new empty maximum spaces that will replace $s'$ on the list *S*. On the other hand, as the maximum spaces are not disjoint, the packed boxes may be intercepted with other maximum spaces which can be reduced or eliminated. The list *B* is also updated and the maximum space that cannot locate any of the remaining boxes to be packed must be eliminated from *S*. If there are still boxes of a customer to be packed, it is necessary to return to *Step 1*.

*Step 4:* Update the list *S* to the next customer in the route. When the current customer has been packed, the maximum spaces must be updated to ensure the sequential loading constraints, that is, all the maximum spaces that are completely invisible from the vehicle door will be removed from the list. The maximum spaces that have a visible and an invisible part must also be updated (modified). If *S* or *B* are empty, this phase is terminated.

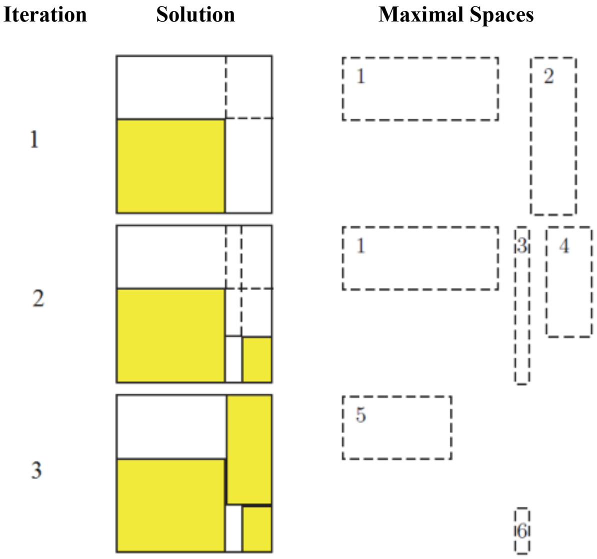

**Fig. 3.** Evolution of the maximum spaces in three iterations of the construction phase. Adapted from Parreño et al. (2010)

Because a greedy construction would not allow a good exploration of the search space, a high degree of randomness is introduced in *Step 2*. Hence, a layer is constructed for each type of item, according to the selection criteria (maximum area or best fit). Each one of these is either called configuration or candidate. A restricted candidate list is built with the whole set of possible layers and one of the these is selected at random. The parameter $\alpha \in \{0,1\}$ controls the size of the candidate list. The performance of the constructive algorithm depends mainly on this parameter. Therefore, in this work, we tuned it in a reactive way as proposed by Camacho et al. (2018), thus allowing for different list sizes to be adjusted according to their historical performance.

The improvement phase consists of applying the aforementioned constructive procedure, eliminating all the random characteristics, after having eliminated *K%* of the packed boxes. The percentage of boxes and the boxes to be removed from the solution are performed as recommended in Escobar et al. (2015).

### *5.3 GA-GRASP Algorithm*

The solution methodology of this study adapted a Chu-Beasley Genetic Algorithm, where recombination operators are used on vectors of the order of customer service, over which the split procedure is applied, which helps to determine the objective function. The split adaptation to the problem of the present work was described in Section 5.1.
The Algorithm 2 illustrates the proposed optimization scheme. In this, the initial population is partially generated in a random way (Lines 1-3), but most of it is built by solutions found by a *Simulated Annealing* based algorithm, in which the TSP obtained by the heuristic of Christofides (1976) is used as the initial solution, applied solely to the customers (Lines 4-7). In total, $\left\lceil \frac{n}{2} \right\rceil$ individuals are generated, where $n$ is the number of customers that must be served in each case.

On the other hand, the generational scheme of the proposed genetic algorithm consists of selection, recombination and mutation. The selection is carried out by tournament (Line 9), in which a couple of parents is selected.  In the recombination, five classical operators of the literature are employed (PMX, SJX, OX, CX y OBX), generating a son for each operator (Line 10). After this, the critical phase of the algorithm thus begins, which requires the calculation of the adaptation function (*fitness*) for each offspring created. Hence, the *split* is adapted to find the routes in an efficient way for the order of service given in the chromosome. In this manner, the first step consists of creating the directed graph (Line 12), considering the different vehicles available in the fleet and the weight of the items and having the objective function (Equation 1) as the cost of the arcs from the routes. After creating the graph, one must solve the shortest path problem over it, as to identify the set of routes that visit all nodes in a more inexpensive fashion. Given that the resulting graph is a directed acyclic graph (DAG), it is possible to use the version of Bellman-Ford algorithm for this type of graph (Line 14). The selected routes (arcs that form the shortest path) have a chance of being mutated (Line 16). The mutation of the routes consists of finding the first improvement by means of the classical 2-opt operator for the TSP (Line 17). The mutated routes directly affect the chromosome, hence, the order of service is corrected so that the individual can pass this information on to the next generations. The routes being mutated or not may be considered infeasible from the packing perspective, making it necessary to validate them with GRASP (Line 19). Each time there is an infeasibility, it is corrected in the graph, eliminating the corresponding arc (Line 21). While there are infeasibilities, one must continue to execute the *split*.

Finally, each feasible individual is presented to the current population, if it meets the degree of diversity and its objective function is better than at least the worst individual in the population, this one will replace it (Lines 25 and 26). In addition, the aspiration criterion was implemented, meaning that, although the individual is not diverse, if it has a better objective function than the worst individual of the "elite group" of the population, this must enter the elite group, thus displacing the worst individual in the population Chu and Beasley (1996). In this way, the basic structure of the elitist algorithm is preserved.

Because validating the packing constraint is an NP-hard problem, it is important to define a fast and efficient route feasibility scheme. To this end, a storage of the results provided by GRASP was implemented as acceleration strategies. In this way, the hash table which stores the validations already made is consulted first, in order to eliminate the number of times the GRASP is called.

```
Algorithm 2 Hybrid GA_GRASP
Data: depot, vehicles, customers, items, max_iterations, mutation_rate, diversity and seed
Result: Solution Routes
1  for $i = 1 \ldots \lfloor \frac{n}{6} \rfloor$ do
2      population[i] = CreateRandomSolution(seed)
3  end
4  constructive_solution = ChristofidesTSP(customers)
5  for $i = \lceil \frac{n}{6} \rceil \ldots \lfloor \frac{n}{2} \rfloor$ do
6      population[i] = SimulatedAnnealing(constructive_solution)
7  end
8  for $k = 1 \ldots Max\_Iterations$ do
9      parents = Tournament(population)
10     offspring = CrossoverPMX-SJX-OX-CX-OBX(parents)
11     for $i = 1 \ldots$ offspring.Size() do
12         graph = CreateDigraph(offspring[i], depot, vehicles, customers, items)
13         while $flag\_SP == false$ do
14             shortest_path = Bellman-FordADG(graph)
15             for each route in shortest_path do
16                 if mutation_rate > Rnd() then
17                     route = Mutate2OPT(route)
18                 end
19                 flag = GRASP(route, items)
20                 if flag == false then
21                     graph = DeleteUnfeasibleArc(route)
22                     flag_SP = true
23                 end
24             end
25         end
26         if AcceptanceCriteria(population, diversity, Decode(shortest_path)) then
27             UpdatePopulation(Decode(shortest_path))
28         end
29     end
30 end
31 return BestSolution(population)
```

## 6. Computational experiments and results analysis

In this section, the results obtained by the proposed methodology are presented. For this purpose, the set of classic instances published in the literature were used. The battery of cases is divided into five classes, each one composed of 36 instances. Because there are no previous works that address the fuel minimization problem, we analyzed the worsening of the distance of the obtained routes, aiming at minimizing the fuel consumption. For this, we executed the proposed methodology on the cases and, similarly, we evaluated the behavior by eliminating fuel consumption. Following this idea, we used the results obtained without considering fuel in order to compare them with previous works in a fair manner. In this way, the performance of the methodology is compared, and it is demonstrated that it achieves better results for some instances, which validates the correct development of the algorithm. All the experiments carried out in this study were executed in a machine with the following specifications: an Intel® Core ™ i5-4570 processor 3.20 GHz, with 8 GB of RAM memory running Ubuntu Linux 16.04.03 64 bits. All algorithms were coded in C++®.

Table 1 shows that the parameter values of the proposed algorithm are mostly calibrated according to the characteristics of the problem. For that purpose, an experimental adjustment of the proportions for each parameter was made, using ranges of values recommended in the literature. The genetic algorithm requires five parameters and each one of these was calibrated with respect to the number of customers to be routed. On the other hand, the GRASP algorithm only requires two parameters due to its reactive nature, which enables it to calibrate its main parameter, based on the historical performance for a reduced number of values.

**Table 1**
Selected values for the parameters

| *Algorithm* | *Parameter* | *Value** |
|---|---|---|
| *GA* | *Number of generations* | $\frac{n}{2log\,n}$ |
| | *Number of individuals* | $\frac{n}{2}$ |
| | *Number of individuals generated at random* | $\frac{n}{6}$ |
| | *Mutation rate (%)* | $\frac{n}{log\,n}$ |
| | *Diversity (genes)* | $\frac{n}{2log\,n}$ |
| *GRASP* | *Number of packing iterations* | $\frac{m}{2log\,m}$ |
| | *Percentage of removed boxes (%)* | *30* |

* *n* is the number of customers to be routed and *m* is the number of boxes to be packed in the route.

Table 2 shows the computational results obtained by the proposed methodology, where column Cost is the objective function achieved by the best solution found and the column Computation Time is the computational time in seconds, required to achieve this solution. Distance is the sum of the distances of the arcs that form the solution. Meanwhile, the column GAP is the percentage difference between the distances of the solution minimizing fuel costs and the solution minimizing only the distances. From Table 2, it can be observed that the proposed methodology manages to solve all the benchmark instances, requiring less than one second for the instances of Class 1 (where the items are homogeneous) and less than a minute for the instances of Class 5 (where the items are highly heterogenous). The difference between the obtained distances is quite small, which means that the cost structure of the objective function incorporated into this work guides the algorithm to balance the weight of the load with which it crosses the arcs. As illustrated in Figure 1, a large part of the edges of a solution is preserved in the other, modifying only its direction. On the other hand, Table 3 presents the results obtained by modifying the methodology so that it seeks to minimize the traveled total distance of the routes. These results are fairly compared with the best solutions reported by previous works in the literature Leung et al. (2013), Domínguez et al. (2016a) and Domínguez et al. (2016b). In particular, the column BKS is the best known solution for each instance and the column GAP is the percentage difference between the distances of BKS and the one obtained in the present work. It should be noted that the same parameters established for the 2L-FHFVRP were used. The computational times achieved are similar to those obtained in the fuel version, however they are not reported because a one-to-one comparison would not be fair due to the differences between the computer architectures and programming languages. In the Table 3, it is shown that the proposed methodology manages to solve all the benchmark instances, achieving many of the best reported solutions and improving others. The difference between the distances obtained is quite small, 8,5% on average. It is worth mentioning that the algorithms against which are compared aim at improving its objective function (minimizing the total distances) through specialized search operators. The proposed algorithm was easily adapted to this version of the loading and packing problem, preserving its original structure.

## 7. Conclusions and future work

We solved the integrated vehicle routing problem and the loading of goods with a view to reducing fuel consumption, through an elitist genetic algorithm of easy implementation, which is a basic characteristic of metaheuristic approaches. We obtained satisfactory results in reasonable computational times (less than 30 seconds on average). Its behavior was validated by comparing its performance with the best previous works published for a variant of the problem that does not consider fuel consumption. We achieved several of the best known solutions and improved some instances, having a small gap against the best reported results. We presented an efficient and fast adaptation of the *split* encoding, which enables to easily incorporate the packing constraints, avoiding to dramatically increase the computational times. The use of this representation allows for introducing real constraints of the problem without modifying the main structure of the algorithm, given that the practical considerations are validated during the creation of the auxiliary graph. Constraints such as time windows, duration of the routes, among others, are satisfied by simply limiting the set of arcs to be created.

We adapted the reactive GRASP algorithm to the validation of feasible routes by packing. We used the maximal spaces encoding to consider the sequential loading constraint in such a way that, between customers, only the feasible spaces are preserved at the time of loading. In addition, its light structure makes it independent of a large number of parameters and it adjusts to the complexity of the variety of items. As for future work, we propose comparing the present study with methodologies that achieve good results to other variants of the vehicle routing problem and packing, especially we suggest implementing the ILS metaheuristic, which appears with better performance in the literature. In addition, we recommend using and adapting the proposed methodology to problems that consider real-world constraints, such as: time windows, maximum travel times, electric vehicles, stackable goods, among others.

**Table 2**
Results obtained by the GA-GRASP

| | Class 1 | | | | Class 2 | | | | Class 3 | | | | Class 4 | | | | Class 5 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Id | Cost | Computation Time | Distance | GAP (%) | Cost | Computation Time | Distance | GAP (%) | Cost | Computation Time | Distance | GAP (%) | Cost | Computation Time | Distance | GAP (%) | Cost | Computation Time | Distance | GAP (%) |
| 1 | 780 | <1 | 604 | 0,0 | 804 | <1 | 647 | 3,7 | 824 | <1 | 668 | 1,2 | 829 | <1 | 666 | 0,0 | 780 | <1 | 635 | 0,0 |
| 2 | 887 | <1 | 698 | 0,5 | 975 | <1 | 771 | 2,3 | 1043 | <1 | 846 | 0,1 | 937 | <1 | 735 | 0,0 | 887 | <1 | 727 | 1,9 |
| 3 | 1034 | <1 | 810 | 0,4 | 1082 | <1 | 865 | 0,0 | 1070 | <1 | 871 | 0,0 | 1022 | <1 | 822 | 0,0 | 1034 | <1 | 855 | 1,0 |
| 4 | 1007 | <1 | 799 | 3,6 | 981 | <1 | 755 | 0,4 | 981 | <1 | 755 | 0,4 | 1000 | <1 | 776 | 1,6 | 1007 | <1 | 838 | 3,7 |
| 5 | 1086 | <1 | 843 | 0,7 | 1182 | <1 | 940 | 0,0 | 1165 | <1 | 919 | 1,4 | 1142 | <1 | 924 | 3,0 | 1086 | <1 | 894 | 5,2 |
| 6 | 1145 | <1 | 853 | 0,0 | 1215 | <1 | 931 | 0,0 | 1215 | <1 | 931 | 2,5 | 1298 | <1 | 1012 | 0,0 | 1145 | <1 | 886 | 0,0 |
| 7 | 4414 | <1 | 3509 | 0,0 | 8851 | <1 | 7587 | 2,3 | 7301 | 1 | 6052 | 0,0 | 7679 | 2,0 | 6641 | 1,5 | 4414 | 2 | 5945 | 0,9 |
| 8 | 4413 | <1 | 3441 | 0,0 | 7431 | <1 | 6096 | 0,0 | 8585 | 1 | 7426 | 0,0 | 8679 | <1 | 7451 | 0,0 | 4413 | 4 | 4336 | 0,8 |
| 9 | 1498 | <1 | 1116 | 1,7 | 1500 | <1 | 1182 | 0,4 | 1490 | <1 | 1174 | 0,4 | 1501 | <1 | 1182 | 0,4 | 1498 | <1 | 1192 | 1,7 |
| 10 | 7176 | <1 | 5591 | 4,0 | 11261 | 1 | 9350 | 0,0 | 10376 | 2 | 8646 | 0,0 | 11485 | 3,7 | 10042 | 1,4 | 7176 | 9 | 7091 | 4,1 |
| 11 | 7136 | <1 | 5351 | 0,9 | 11379 | 1 | 9355 | 0,0 | 11924 | 2 | 10092 | 0,0 | 12572 | 5,4 | 10703 | 1,1 | 7136 | 13 | 7038 | 0,0 |
| 12 | 2244 | <1 | 1764 | 0,8 | 2296 | <1 | 1823 | 0,3 | 2292 | <1 | 1841 | 0,9 | 2287 | <1 | 1816 | 0,5 | 2244 | <1 | 1802 | 0,3 |
| 13 | 22251 | <1 | 16692 | 0,9 | 37425 | 2 | 31794 | 2,2 | 36753 | 3 | 30907 | 4,7 | 38896 | 2,3 | 32619 | 2,0 | 22251 | 9 | 26745 | 2,4 |
| 14 | 13100 | <1 | 10124 | 0,2 | 15085 | <1 | 12032 | 0,0 | 16082 | 1 | 13336 | 0,0 | 15682 | <1 | 12894 | 0,0 | 13100 | 1 | 12102 | 0,0 |
| 15 | 12920 | <1 | 9988 | 0,9 | 15107 | <1 | 12381 | 0,0 | 14938 | <1 | 11989 | 0,0 | 17062 | 1,2 | 14248 | 0,0 | 12920 | 3 | 13351 | 0,0 |
| 16 | 1749 | <1 | 1343 | 0,4 | 1806 | <1 | 1400 | 0,0 | 1824 | <1 | 1424 | 0,7 | 1801 | <1 | 1382 | 0,0 | 1749 | <1 | 1343 | 0,2 |
| 17 | 2379 | <1 | 1827 | 0,1 | 2452 | <1 | 1925 | 0,0 | 2418 | <1 | 1837 | 0,0 | 2429 | <1 | 1878 | 0,0 | 2379 | <1 | 1894 | 0,0 |
| 18 | 4594 | <1 | 3521 | 2,6 | 7260 | 3 | 6437 | 0,5 | 6991 | 4 | 6083 | 0,0 | 8187 | 16,6 | 7354 | 0,3 | 4594 | 14 | 5117 | 0,0 |
| 19 | 2137 | <1 | 1637 | 1,3 | 6109 | 3 | 5312 | 1,0 | 6048 | 5 | 5412 | 1,3 | 6504 | 10,6 | 5727 | 0,6 | 2137 | 17 | 4077 | 2,0 |
| 20 | 2446 | <1 | 1945 | 0,0 | 7097 | 26 | 6538 | 0,2 | 6747 | 60 | 6217 | 0,7 | 7459 | 97,1 | 6904 | 0,0 | 2446 | 178 | 5195 | 0,5 |
| 21 | 3539 | <1 | 2713 | 0,7 | 11245 | 5 | 9914 | 0,5 | 12090 | 11 | 10687 | 0,4 | 11042 | 15,1 | 9613 | 0,8 | 3539 | 25 | 7736 | 1,2 |
| 22 | 3582 | <1 | 2765 | 1,3 | 11247 | 5 | 10002 | 0,2 | 12283 | 11 | 10887 | 1,1 | 11504 | 20,0 | 10102 | 0,4 | 3582 | 40 | 8137 | 0,3 |
| 23 | 3539 | <1 | 2713 | 0,7 | 11782 | 5 | 10342 | 0,8 | 10964 | 11 | 9719 | 0,2 | 11444 | 18,5 | 10157 | 0,6 | 3539 | 31 | 7589 | 0,8 |
| 24 | 3523 | <1 | 2705 | 0,8 | 6819 | 1 | 5432 | 0,1 | 6303 | 1 | 5120 | 2,4 | 6288 | 2,3 | 4997 | 1,8 | 3523 | 2 | 4307 | 2,3 |
| 25 | 3958 | <1 | 2972 | 0,4 | 15831 | 14 | 14196 | 0,2 | 14211 | 27 | 12659 | 0,2 | 15797 | 47,3 | 14271 | 0,8 | 3958 | 101 | 9097 | 1,2 |
| 26 | 5058 | <1 | 3741 | 0,3 | 15762 | 6 | 13667 | 1,2 | 15565 | 15 | 13354 | 0,7 | 16367 | 27,2 | 14164 | 0,7 | 5058 | 52 | 9304 | 1,0 |
| 27 | 4039 | <1 | 3034 | 0,2 | 7825 | 2 | 6210 | 0,3 | 8002 | 3 | 6482 | 0,8 | 7851 | 6,3 | 6396 | 1,4 | 4039 | 11 | 5691 | 1,2 |
| 28 | 6114 | <1 | 4446 | 0,9 | 27734 | 31 | 24970 | 0,0 | 26203 | 58 | 23560 | 0,0 | 28516 | 102,6 | 25668 | 0,4 | 6114 | 228 | 18350 | 0,2 |
| 29 | 14545 | <1 | 10957 | 1,4 | 28022 | 55 | 24182 | 0,1 | 25478 | 116 | 22077 | 0,0 | 27280 | 244,1 | 23534 | 0,6 | 14545 | 307 | 21425 | 0,1 |
| 30 | 5633 | <1 | 4311 | 0,6 | 21571 | 17 | 19004 | 0,4 | 21301 | 34 | 18902 | 0,6 | 22363 | 69,4 | 19933 | 0,1 | 5633 | 113 | 13314 | 0,8 |
| 31 | 7722 | <1 | 5924 | 0,5 | 26599 | 16 | 23530 | 0,5 | 27467 | 31 | 24099 | 0,3 | 28355 | 84,4 | 25150 | 0,2 | 7722 | 133 | 17463 | 0,3 |
| 32 | 7677 | <1 | 5846 | 0,2 | 27738 | 20 | 24606 | 0,4 | 24942 | 44 | 22031 | 0,8 | 28337 | 81,9 | 24816 | 0,5 | 7677 | 123 | 16280 | 0,3 |
| 33 | 7805 | <1 | 5952 | 0,1 | 26983 | 20 | 23770 | 0,2 | 26428 | 44 | 23364 | 0,5 | 28718 | 79,2 | 25457 | 0,7 | 7805 | 134 | 17357 | 1,3 |
| 34 | 5744 | <1 | 4589 | 0,3 | 18066 | 14 | 16046 | 0,2 | 17228 | 40 | 15294 | 0,5 | 19662 | 74,0 | 17476 | 0,2 | 5744 | 104 | 11708 | 0,5 |
| 35 | 5372 | <1 | 4164 | 0,1 | 11259 | 19 | 9851 | 0,0 | 11397 | 36 | 10025 | 0,1 | 12025 | 72,4 | 10637 | 0,1 | 5372 | 184 | 8574 | 0,2 |
| 36 | 3765 | <1 | 3011 | 0,0 | 5806 | 20 | 5124 | 0,0 | 5802 | 37 | 5120 | 0,0 | 5543 | 61,0 | 4861 | 0,0 | 3765 | 141 | 4288 | 0,0 |
| **Average** | | <1 | | 0,8 | | 7,9 | | 0,5 | | 16,6 | | 0,6 | | 31,9 | | 0,6 | | 55,0 | | 1,0 |

**Table 3**
Results GA-GRASP and comparison with the best known solutions in the literature (BKS)

| | *Class 1* | | | *Class 2* | | | *Class 3* | | | *Class 4* | | | *Class 5* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Id | BKS | Solution | GAP (%) | BKS | Solution | GAP (%) | BKS | Solution | GAP (%) | BKS | Solution | GAP (%) | BKS | Solution | GAP (%) |
| *1* | *596,07* | *596,07* | *0,0* | *602,88* | *610,37* | *1,2* | *597,04* | *597,04* | *0,0* | *596,07* | *596,07* | *0,0* | *596,07* | *597,04* | *0,2* |
| *2* | *670,3* | *679,18* | *1,3* | *700,94* | *707,08* | *0,9* | *728,24* | *733,47* | *0,7* | *684,99* | *717,48* | *4,5* | *670,3* | *679,18* | *1,3* |
| *3* | *745,51* | *757,18* | *1,5* | *765,8* | *801,62* | *4,5* | *777,6* | *788,53* | *1,4* | *754,87* | *765,8* | *1,4* | *754,87* | *770,17* | *2,0* |
| *4* | *694,33* | *694,33* | *0,0* | *697,06* | *715,75* | *2,6* | *697,06* | *698,3* | *0,2* | *703,76* | *703,76* | *0,0* | *694,87* | *710,19* | *2,2* |
| *5* | *758,47* | *758,47* | *0,0* | *762,92* | *804,14* | *5,1* | *772,53* | *779,91* | *0,9* | *768,74* | *798,12* | *3,7* | *758,47* | *758,47* | *0,0* |
| *6* | *809,56* | *809,56* | *0,0* | *824,6* | *830,56* | *0,7* | *827,6* | *857,22* | *3,5* | *836,22* | *882,99* | *5,3* | *822,69* | *829,3* | *0,8* |
| *7* | *3211,53* | *3213,88* | *0,1* | *5952,09* | *6089,29* | *2,3* | *5135,66* | *5415,49* | *5,2* | *5466,9* | *5710,76* | *4,3* | *4723,64* | *4939,01* | *4,4* |
| *8* | *3184,45* | *3201,1* | *0,5* | *4804,14* | *4814* | *0,2* | *6400,11* | *6717,82* | *4,7* | *6290,15* | *6304,94* | *0,2* | *3864,97* | *3830,33* | *-0,9* |
| *9* | *1029,95* | *1031,55* | *0,2* | *1029,95* | *1089,67* | *5,5* | *1029,95* | *1037,79* | *0,8* | *1052,62* | *1084,6* | *2,9* | *1029,95* | *1029,95* | *0,0* |
| *10* | *4932,34* | *5082,18* | *2,9* | *7439,25* | *7888,78* | *5,7* | *6173,72* | *6360,87* | *2,9* | *7977,48* | *7879,66* | *-1,2* | *6065* | *6179,08* | *1,8* |
| *11* | *4932,34* | *5020,19* | *1,7* | *8007,51* | *8545,93* | *6,3* | *8289,87* | *8667,75* | *4,4* | *8820,13* | *8886,6* | *0,7* | *6053,52* | *6177,61* | *2,0* |
| *12* | *1655,73* | *1655,73* | *0,0* | *1674,56* | *1683,15* | *0,5* | *1681,51* | *1702,72* | *1,2* | *1680,01* | *1686,77* | *0,4* | *1677,52* | *1702,99* | *1,5* |
| *13* | *14579,87* | *14902,5* | *2,2* | *26318,79* | *27766* | *5,2* | *24598,85* | *25713,3* | *4,3* | *26103,64* | *27078,9* | *3,6* | *22808,22* | *23527,2* | *3,1* |
| *14* | *9097,52* | *9227,83* | *1,4* | *10389,98* | *10833,3* | *4,1* | *11345,99* | *11438,7* | *0,8* | *10233,7* | *10601,4* | *3,5* | *10005,07* | *10364,1* | *3,5* |
| *15* | *9097,52* | *9245,66* | *1,6* | *10780,29* | *10958,9* | *1,6* | *10753,2* | *11108,8* | *3,2* | *11449,4* | *12263,2* | *6,6* | *11527,11* | *12172,3* | *5,3* |
| *16* | *1276,22* | *1297,13* | *1,6* | *1280,9* | *1290,63* | *0,8* | *1288,07* | *1309,27* | *1,6* | *1290* | *1330,26* | *3,0* | *1280,9* | *1296,03* | *1,2* |
| *17* | *1764,32* | *1783,97* | *1,1* | *1756,5* | *1860,38* | *5,6* | *1766,5* | *1816,89* | *2,8* | *1766,5* | *1836,54* | *3,8* | *1768,57* | *1753,96* | *-0,8* |
| *18* | *3140,55* | *3203,28* | *2,0* | *5862,61* | *5882,47* | *0,3* | *5197,55* | *5479,62* | *5,1* | *6084,39* | *7334,08* | *17,0* | *4435,96* | *5416,18* | *18,1* |
| *19* | *1490,28* | *1566,72* | *4,9* | *3955,68* | *4797,48* | *17,5* | *3907,65* | *5209,65* | *25,0* | *4641,27* | *6027,38* | *23,0* | *3284,6* | *3775,79* | *13,0* |
| *20* | *1776,44* | *1870,38* | *5,0* | *5354,01* | *6547,4* | *18,2* | *5468,92* | *6194,45* | *11,7* | *5689,54* | *6894,8* | *17,5* | *4133,83* | *5103,29* | *19,0* |
| *21* | *2470,47* | *2576,9* | *4,1* | *6858,61* | *9597,62* | *28,5* | *8228,34* | *10261,5* | *19,8* | *7840,41* | *9592,66* | *18,3* | *5437,85* | *7768,4* | *30,0* |
| *22* | *2470,47* | *2566,33* | *3,7* | *7166,96* | *9463,43* | *24,3* | *8117,96* | *10259,2* | *20,9* | *8107,14* | *10217,6* | *20,7* | *5899,7* | *7229,97* | *18,4* |
| *23* | *2470,47* | *2600,85* | *5,0* | *7984,9* | *10190* | *21,6* | *7555,01* | *9747,97* | *22,5* | *8230,75* | *10190* | *19,2* | *5504,2* | *7503,28* | *26,6* |
| *24* | *2470,47* | *2580,33* | *4,3* | *4584,58* | *5561,92* | *17,6* | *4271,53* | *4800,68* | *11,0* | *4520,48* | *5050,71* | *10,5* | *3805,04* | *4243,31* | *10,3* |
| *25* | *2728,4* | *2961,77* | *7,9* | *10679,78* | *14814,3* | *27,9* | *10049,97* | *12714,1* | *21,0* | *11558,3* | *13917,1* | *16,9* | *7227,21* | *9408,58* | *23,2* |
| *26* | *3477,26* | *3587,97* | *3,1* | *10595,48* | *13400,8* | *20,9* | *10700,47* | *12833* | *16,6* | *11725,4* | *14424,9* | *18,7* | *8185,74* | *9234,58* | *11,4* |
| *27* | *2728,4* | *2958,89* | *7,8* | *5373,07* | *6536,98* | *17,8* | *5735,3* | *6326,48* | *9,3* | *5497,14* | *6202,87* | *11,4* | *5101,17* | *5640,89* | *9,6* |
| *28* | *4065,7* | *4281,62* | *5,0* | *19433,58* | *24899,1* | *22,0* | *20023,42* | *23315,4* | *14,1* | *21579,76* | *25619,3* | *15,8* | *16097,78* | *18793,9* | *14,3* |
| *29* | *9105,07* | *9897,04* | *8,0* | *20379,1* | *24508,5* | *16,8* | *20343,62* | *22334,8* | *8,9* | *21568,48* | *24508,5* | *12,0* | *20143,84* | *21399,6* | *5,9* |
| *30* | *3946,79* | *4289,22* | *8,0* | *14900,24* | *18742* | *20,5* | *15222,37* | *18742* | *18,8* | *16450,52* | *20148,9* | *18,4* | *10388,77* | *13369* | *22,3* |
| *31* | *5370,31* | *5836,01* | *8,0* | *18282,14* | *23119,8* | *20,9* | *20243,98* | *23516,9* | *13,9* | *21480,92* | *24618,5* | *12,7* | *14853,19* | *18341,8* | *19,0* |
| *32* | *5370,34* | *5860,83* | *8,4* | *18935,52* | *24364,8* | *22,3* | *17921,4* | *21408,4* | *16,3* | *20308,82* | *24038,8* | *15,5* | *13644,69* | *17064,7* | *20,0* |
| *33* | *5377,08* | *5870,44* | *8,4* | *18846,63* | *24566,5* | *23,3* | *20344,29* | *23102,4* | *11,9* | *21899,41* | *24670,6* | *11,2* | *13648,79* | *16183,3* | *15,7* |
| *34* | *4263,78* | *4493,55* | *5,1* | *13606,24* | *15922,4* | *14,5* | *14185,86* | *15938,9* | *11,0* | *15326,26* | *16858,9* | *9,1* | *10480,51* | *11227,4* | *6,7* |
| *35* | *3911,17* | *4139,41* | *5,5* | *8340,24* | *9859,26* | *15,4* | *8876,94* | *10169,6* | *12,7* | *9226,53* | *10635,9* | *13,3* | *7776,65* | *8608,66* | *9,7* |
| *36* | *2847,02* | *3010,48* | *5,4* | *4290,8* | *5093,51* | *15,8* | *4524,41* | *5093,51* | *11,2* | *4280,03* | *4931,87* | *13,2* | *3883,48* | *4279,52* | *9,3* |
| ***Average*** | | | *3,49* | | | *11,64* | | | *8,90* | | | *9,37* | | | *9,16* |

## Acknowledgments

The authors thank the support of Integra S.A., MinCiencias, the Master Program in Electrical Engineering and the Doctoral Program in Engineering of the Universidad Tecnológica de Pereira, and the Universidad Libre Seccional Pereira.

## References

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1988). Network flows.

Alvarez-Martínez, D., Alvarez-Valdes, R., & Parreño, F. (2015). A GRASP algorithm for the container loading problem with multi-drop constraints. *Pesquisa Operacional*, *35*(1), 1-24. https://dx.doi.org/10.1590/0101-7438.2015.035.01.0001

Beasley, J. E., & Chu, P. C. (1996). A genetic algorithm for the set covering problem. *European Journal of Operational Research*, *94*(2), 392-404.

Bektaş, T., & Laporte, G. (2011). The pollution-routing problem. *Transportation Research Part B: Methodological*, *45*(8), 1232-1250.

Camacho, G. A., Alvarez, D., & Cuellar, D. (2018). Heuristic Approach For The Multiple Bin-Size Bin Packing Problem. *IEEE Latin America Transactions*, *16*(2), 620-626.

Christofides, N. (1976). *Worst-case analysis of a new heuristic for the travelling salesman problem.* Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group.

Côté, J. F., Guastaroba, G., & Speranza, M. G. (2017). The value of integrating loading and routing. *European Journal of Operational Research*, *257*(1), 89-105.

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

Demir, E., Bektaş, T., & Laporte, G. (2014). A review of recent research on green road freight transportation. *European Journal of Operational Research*, *237*(3), 775-793.

Dominguez, O., Juan, A. A., Barrios, B., Faulin, J., & Agustin, A. (2016). Using biased randomization for solving the two-dimensional loading vehicle routing problem with heterogeneous fleet. *Annals of Operations Research*, *236*(2), 383-404.

Domínguez Rivero, O. L., Juan Pérez, A. A., de la Nuez Pestana, I. A., & Ouelhadj, D. (2016). An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation. *Journal of the Operational Research Society*, *67*(1), 37-53.

Escobar, L. M., Martínez, D. Á., Escobar, J. W., Linfati, R., & Mauricio, G. E. (2015, October). A hybrid metaheuristic approach for the capacitated vehicle routing problem with container loading constraints. In *2015 International Conference on Industrial Engineering and Systems Management (IESM)* (pp. 1374-1382). IEEE.

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.

Feo, T. A., & Resende, M. G. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations research letters*, *8*(2), 67-71.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2006). A tabu search algorithm for a routing and container loading problem. *Transportation Science*, *40*(3), 342-350.

Guimarans, D., Domínguez, O., Juan, A. A., & Martínez, E. (2016). *A multi-start simheuristic for the stochastictwo-dimensional vehicle routing problem*. Proceedings of the 2016 Winter Simulation Conference (pp. 2326-2334). Piscataway: IEEEPress. https://doi.org/10.1109/WSC.2016.7822273

Guimarans, D., Dominguez, O., Panadero, J., & Juan, A. A. (2018). A simheuristic approach for the two-dimensional vehicle routing problem with stochastic travel times. *Simulation Modelling Practice and Theory*, *89*, 1-14.

Iori, M., Salazar-González, J. J., & Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation science*, *41*(2), 253-264.

Iori, M., & Martello, S. (2013). An annotated bibliography of combined routing and loading problems. *Yugoslav Journal of Operations Research*, *23*(3), 311-326(16).

Leung, S. C., Zhang, Z., Zhang, D., Hua, X., & Lim, M. K. (2013). A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research*, *225*(2), 199-210.

Liu, S., Huang, W., & Ma, H. (2009). An effective genetic algorithm for the fleet size and mix vehicle routing problems. *Transportation Research Part E: Logistics and Transportation Review*, *45*(3), 434-445.

Micheli, G. J., & Mantella, F. (2018). Modelling an environmentally-extended inventory routing problem with demand uncertainty and a heterogeneous fleet under carbon control policies. *International Journal of Production Economics*, *204*, 316-327.

Parreño, F., Alvarez-Valdés, R., Oliveira, J. F., & Tamarit, J. M. (2010). Neighborhood structures for the container loading problem: a VNS implementation. *Journal of Heuristics*, *16*(1), 1-22.

Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., & Limbourg, S. (2015). Vehicle routing problems with loading constraints: state-of-the-art and future directions.*OR Spectrum*, 37(2):297–330.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, *31*(12), 1985-2002.

Subramanian, A., Uchoa, E., & Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, *40*(10), 2519-2531.

Suzuki, Y. (2011). A new truck-routing approach for reducing fuel consumption and pollutants emission. *Transportation Research Part D: Transport and Environment*, *16*(1), 73-77.

Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: problems, methods, and applications*. Society for Industrial and Applied Mathematics.

Xiao, Y., Zhao, Q., Kaku, I., e Xu, Y. (2012). Development of a fuel consumption optimizationmodel for the capacitated vehicle routing problem.*Computers & Operations Research, 39*(7), 1419–1431.

Zhang, Z., Wei, L., & Lim, A. (2015). An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints. *Transportation Research Part B: Methodological*, *82*, 20-35.