

Article

A Multi-Start Algorithm for Solving the Capacitated Vehicle Routing Problem with Two-Dimensional Loading Constraints

Leandro Pinto Fava¹, João Carlos Furtado^{1*}, Gilson Augusto Helfer², Jorge Luis Victória Barbosa², Marko Beko^{3,4}, Sérgio Duarte Correia^{4,5} and Valderi Reis Quietinho Leithardt^{4,5}

- ¹ Industrial Systems and Processes Graduate Program, University of Santa Cruz do Sul; Av. Independência 2293, Santa Cruz do Sul, RS 96815-900, Brazil; jcarlosf@unisc.br
- ² Applied Computing Graduate Program, University of Vale do Rio dos Sinos, Av. Unisinos 950, São Leopoldo, RS 93022-750, Brazil; jbarbosa@unisinos.br
- ³ Instituto de Telecomunicações, Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisbon, Portugal; marko.beko@tecnico.ulisboa.pt
- ⁴ COPELABS, University Lusófona - ULHT, 1749-024 Lisbon, Portugal.
- ⁵ VALORIZA – Research Centre for Endogenous Resource Valorization, Polytechnic Institute of Portalegre, Portugal; scorreia, valderi@ippportalegre.pt
- * Correspondence: jcarlosf@unisc.br; Tel.: +55-51-98441-4343



Citation: Fava, L. P.; Furtado, J. C.; Helfer, G. A.; Barbosa, J. L. V.; Beko, M.; Correia, S. D.; Leithardt, V. R. Q. A multi-start algorithm for solving the capacitated vehicle routing problem with two-dimensional loading constraints. *Preprints* 2021, 1, 0. <https://doi.org/>

Received:
Accepted:
Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Abstract: This work presents a multi-start algorithm for solving the capacitated vehicle routing problem with two-dimensional loading constraints (2L-CVRP) allowing for the rotation of goods. Researches dedicated to graph theory and symmetry considered the vehicle routing problem as a classical application. This problem has complex aspects that stimulate the use of advanced algorithms and symmetry in graphs. The use of graph modeling of the 2L-CVRP problem by undirected graph allowed the high performance of the algorithm. The developed algorithm is based on metaheuristics such as the Constructive Genetic Algorithm (CGA), to construct promising initial solutions; a Tabu Search (TS), to improve the initial solutions on the routing problem; and a Large Neighborhood Search (LNS), for the loading subproblem. Although each one of these algorithms allowed to solve parts of the 2L-CVRP, the combination of these three algorithms to solve this problem was unprecedented in the scientific literature. In our approach, a parallel mechanism for checking the loading feasibility of routes was implemented using multi-threading programming to improve the performance. Additionally, memory structures, like hash-tables, were implemented to save time by storing and querying previously evaluated results for the loading feasibility of routes. For benchmarks, tests were done on well-known instances available in the literature. The results proved that the framework matched or outperformed most of the previous approaches. As the main contribution, this work brings higher quality solutions for large-size instances of the pure CVRP. This paper involves themes related to the symmetry journal, mainly complex algorithms, graphs, search strategies, complexity, graph modeling, and genetic algorithms. In addition, the paper especially focuses on topic-related aspects of special interest of the community involved in symmetry studies, such as, graph algorithms and graph theory.

Keywords: Routing; 2L-CVRP; Loading; Multithreading; Rotation; Tabu Search; Graph Theory.

1. Introduction

The vehicle routing problem (VRP) is a class of computational optimization problem that involves designing delivery routes and lower logistic cost collection to satisfy the demands of a set of customers or a group of geographically dispersed cities [1,2]. The VRP generalises the travelling salesman problem (TSP) [3]. Although the TSP is an old problem whose origin is not well known, many studies on it are still carried out, as is the case of [4]. The VRP is addressed in studies dedicated to graph theory and algorithms [5] and, especially, several research works related to different problems involving vehicles and routes were published considering symmetry aspects [6–10]. As global trade has expanded and led to a significant growth in the demand for transport, the resolution of VRPs, introduced by Dantzig and Ramser [11], has become increasingly essential for

enterprises looking to efficiently manage their transport and coordinate their supply chain. The purpose of addressing this problem is to minimize transport costs, typically by reducing the total distance driven and the number of vehicles used.

The VRP has been studied for years and, due to computational advances, several constraints have been added to the problem. For example, when the weight of the cargo is considered because of the capacity of each vehicle in a fleet, we have a variant of the VRP called the capacitated vehicle routing problem (CVRP). In another example, Vidal et al. in 2013 presented an hybrid genetic search with advanced diversity control for a large class of time-constrained VRP [12].

Another combinatorial problem that is addressed for a long time is the two-dimensional bin packing problem (2BPP). In 2BPP we need to allocate without overlapping several rectangular items into a minimum number of rectangular bins, and the edges of the items must be parallel to the bins [13].

The 2BPP and CVRP are considered when the customer demands a set of items that are represented by two-dimensional, rectangular forms. In this case, we have the 2L-CVRP. Also, solving the BPPs has been crucial to improving the logistics of moving, storing, and transporting products. For this reason, the purpose of addressing CVRP and BPP is to minimize logistics costs by maximizing the area or the volume of items packed within a transport vehicle.

The problem introduced in this study belongs to this class of integrated vehicle routing and loading problems. This study examines logistics activities with the following key characteristics: vehicles with a limited capacity based on a depot point that serve geographically dispersed customers that demand a heterogeneous product. The products transported are two-dimensional items that are considered non-stackable. Thus, effective two-dimensional arrangements for loading the transported items into vehicles must be identified.

This paper presents a metaheuristic approach for solving the Capacitated Vehicle Routing Problem (CVRP) with two-dimensional loading constraints (2L-CVRP). The use of graph modeling of the 2L-CVRP problem by undirected graph allowed the high performance of the algorithm. The proposal considers a combination of three techniques. For the CVRP we used a Constructive Genetic Algorithm (CGA) approach, which is a successful technique for solving related problems. We applied a Tabu Search (TS) approach to improve effectiveness. And for the 2BPP, we used a Large Neighborhood Search (LNS) based algorithm to determine the two-dimensional loading feasibility when the rotation of goods is permitted.

Neither the CVRP nor the 2L-CVRP has been previously studied with a CGA approach, although they have been reviewed with a TS or with another Genetic Algorithm approach. The main premise of this work is that it is both possible and advantageous to employ new optimization methods to model and solve the VRP and the 2BPP.

The contributions of this paper may be summarized as follows: (a) the construction of a new metaheuristic CGA improved by the TS for the routing problem in conjunction with the LNS for the two-dimensional packing problem which allows for rotation; (b) as far as we know, there are only four previous works which have considered allowing for the rotation of items in the 2L-CVRP. This paper can add results to these works through; (c) significant results by exploring the parallelism when processing the vehicle loading phase by using multithreading in a multi-core system; (d) a collection of the best results in the literature.

This work approaches several themes of interest of symmetry journal, such as, complex algorithms, graphs, search strategies, complexity, graph modeling, and genetic algorithms, especially, this work explores graph algorithms and graph theory applying these themes in the vehicle routing problem.

The paper is organized as follows. Section 2 introduces a literature review about the VRP, focusing mainly on the CVRP and the 2L-CVRP. Section 3 describes the problems and introduces the necessary notation. Section 4 presents our proposed framework: the

CGA-TS approach for the CVRP and the LNS for the 2L-CVRP. Section 5 presents the results of the testing and a comparative to other approaches found in the literature as well as a statistical validation for the comparison. Finally, section 6 shows conclusions based on the study and suggestions for future works.

2. Literature review

In the field of the VRP, we can find a combination of the capacitated vehicle routing problem (CVRP) with two-dimensional loading constraints, which is denoted as the 2L-CVRP. The first work on the 2L-CVRP was presented by Iori et al. [14]. They proposed an exact method based on the branch-and-cut algorithm to minimize the routing costs and a nested branch-and-bound algorithm to solve the two-dimensional sequential oriented loading subproblem (2|SO|L). Later, Gendreau et al. presented an approach using a metaheuristic to solve the 2L-CVRP [15]. They proposed a solution based on the tabu search algorithm. They applied heuristics, lower bounds and a branch-and-bound algorithm for the loading feasibility checks, considering both sequential and unrestricted oriented loading, 2|SO|L and 2|UO|L, respectively.

Fuellerer et al. [16] developed a metaheuristic approach based on ant colony optimization to solve the routing problem combined with heuristics for the loading sub-problem. For the first time, the results allowed for the rotation of goods by 90° (non-oriented) when considering the 2L-CVRP. Thus, two new variants of loading constraints were studied considering the non-oriented version with sequential and unrestricted loading, 2|SR|L and 2|UR|L respectively. Zachariadis et al. [17] proposed a metaheuristic algorithm based on the combination of the tabu search with a guided local search and a collection of packing heuristics for the loading subproblem. Strodl et al. [18] developed a solution for the routing problem using a variable neighborhood search and an exact branch-and-bound algorithm for the loading. A simulated annealing algorithm with a collection of packing heuristics to solve the 2L-CVRP was presented by Leung et al. in 2010 [19]. Later, Leung et al. [20] in 2011 also proposed an approach based on the tabu search combined with an extended guided local search and a collection of packing heuristics. Duhamel et al. [21] presented a study with greedy randomized adaptive search and evolutionary local search algorithms. Zachariadis et al. [22] presented a work with an algorithm referred to as Promise Routing-Memory Packing (PRMP) to solve 2|UO|L and 2|SO|L versions of the 2L-CVRP. Dominguez et al. [23] published a paper with a multi-start biased randomized algorithm (MS-BR) to solve the 2L-CVRP with two unrestricted loading configurations (2|UO|L and 2|UR|L). This was the second paper proposed in the literature to solve the two-dimensional loading constraint allowing for the rotation of items. Wei et al. proposed a variable neighborhood search algorithm combined with a skyline heuristic to solve 2|UO|L and 2|SO|L versions of the 2L-CVRP [24]. Recently, Wei et al. [25] proposed a simulated annealing algorithm with an open space based heuristic to check the loading feasibility, which outperformed all the previous approaches on 2|SO|L, 2|UO|L, 2|SR|L and 2|UR|L. Related to the previous 2L-CVRP work cited, the open space based heuristic for the two-dimensional strip packing problem was presented also by Wei et al. [26].

Many other constraints related to the real-world problems of the logistics industry have been studied with the 2L-CVRP. Zachariadis et al. [27] proposed an algorithm based on an extended version of their previous approach [22] to solve the 2L-CVRP with simultaneous pick-ups and deliveries. They also considered the results for the 2|UR|L version and obtained good results. In this paper, their algorithm is referred to as the xPRMP. Pinto et al. presented a study using variable neighborhood search algorithms to solve the 2L-CVRP with mixed linehauls and backhauls [28].

The CVRP with three-dimensional loading constraints (3L-CVRP) is a related problem that has been studied extensively in recent years. The 3L-CVRP was studied by many authors who proposed metaheuristics like a tabu search [29], a guided tabu search [30], the ant colony algorithm [31], and a GRASPxELS [32]. Bortfeldt [33] presented a hybrid algorithm for the 3L-CVRP. Koch, Bortfeldt and Wäscher [34] presented a hybrid algorithm

to solve the 3L-CVRP with backhauls and time windows. Recently, Bortfeldt and Yi [35] proposed a hybrid algorithm to solve the split delivery VRP (SDVRP) with the 3L-CVRP. We can cite some surveys on this subject area from Iori and Martello [36] and from Polaris et al. [37].

Finally, we can cite some other TSP and VRP related studies like the orienteering problem presented for the first time in [38] and [39] proposed a memetic approach to solve it. It should be noted that the green vehicle routing problem (GVRP) is an emerging research field, and a recent systematic literature review on this field can be seen in [40].

3. Problem description

Zachariadis et al. [17] describe the 2L-CVRP as follows. Let $G = (V, E)$ be an undirected graph, where $V = \{0, \dots, n\}$ is the vertices set, in which vertex 0 represents the single depot and the other n vertices correspond to the customers, and $E = \{(i_0, j_0), \dots, (i_n, j_n) : i, j \in V, i \neq j\}$ is the set of edges. Each edge $(i, j) \in E$ has an associated cost c_{ij} that corresponds to the cost for the transport from i to j .

There are v identical vehicles, each with a weight capacity equal to Q and a rectangular loading surface of width W and length L (related (x, y) coordinates). Let $A = W \times L$ denote the loading area. The demand of customer i ($i = 1, \dots, n$) consists a set of m_i items, denoted as IT_i , of total weight q_i : item $I_{ik} \in IT_i$ ($k = 1, \dots, m_i$) has width w_{ik} and length l_{ik} . Let $a_i = \sum_{k=1}^{m_i} w_{ik} l_{ik}$ denotes the total area of the items demanded by customer i .

The objective of the 2L-CVRP is to determine the minimum cost set of routes that satisfy the following constraints: (a) the quantity of generated routes does not exceed the number of available vehicles; (b) all routes must start and end at the central depot; (c) each customer i can be visited only once; (d) each customer i must be served by only one vehicle; (e) the total weight of items demanded for a route does not exceed Q ; and (f) all loading must be non-stackable by the set of customers covered by a route into the WL loading surface of the vehicles.

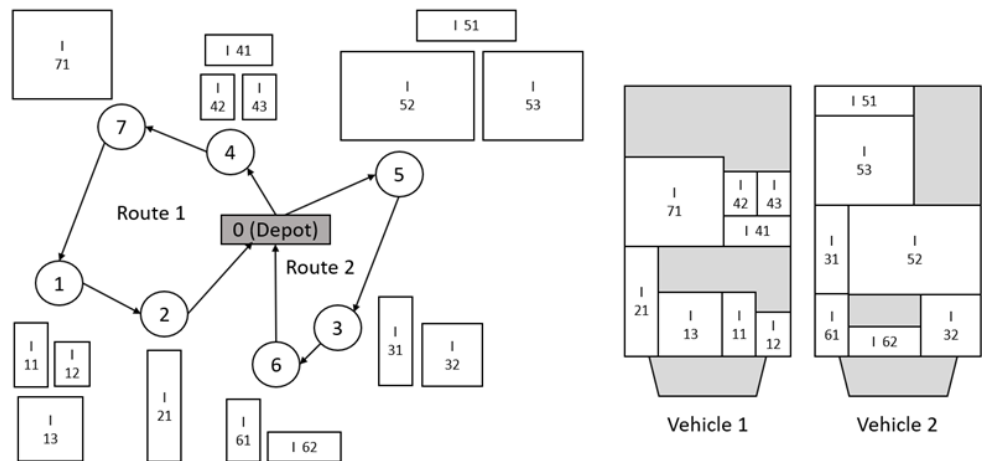


Figure 1. Example of the 2|SO|L 2L-CVRP.

Usually, a practical constraint can be imposed, considering the convenience of unloading: each time a customer i is visited, all items I_{ik} must be unloaded so that no items of other customers are moved, when k represents the items of a particular customer. This version is called sequential loading (also referred to as LIFO) and it can be denoted as 2|SO|L when the rotation of items is not allowed. Figure 1 depicts an example of the 2|SO|L 2L-CVRP. The version of loading without the LIFO constraint is called unrestricted. According to Fuellerer et al. [16], four combinations can be listed as follows: 2|UO|L: unrestricted, oriented loading; 2|UR|L: unrestricted, rotation allowed loading; 2|SO|L: sequential, oriented loading; and 2|SR|L: sequential, rotation allowed loading.

The following describes the employed solution approach to solve the 2|UR|L 2L-CVRP. Firstly, the routing component is determined in a multi-start approach. It consists of

156 a constructive procedure to generate an initial solution. It then continues to search to find a
157 trajectory in the search space that is the core of the optimization method to produce a final
158 solution. This master routing algorithm utilizes a loading feasibility method that generates
159 feasible loading patterns.

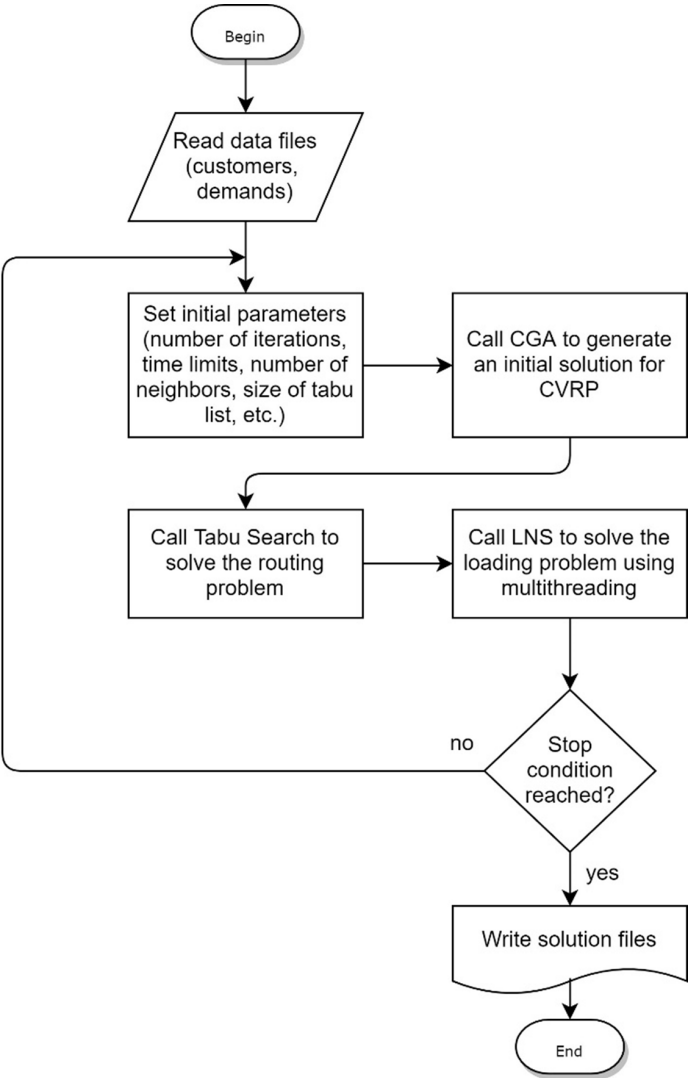


Figure 2. Macro flow of the CGA-TS-LNS algorithm.

160 4. The multi-start CGA-TS-LNS algorithm

161 The proposed algorithm to solve the 2|UR|L version of the 2L-CVRP in this work
162 makes use of the combination of heuristics derived from the genetic algorithm (GA) [41],
163 the tabu search (TS) [42] and the Large Neighborhood Search (LNS) [43].

164 The CGA-TS-LNS algorithm was developed with a multi-start approach, wherein
165 each global iteration, a new initial solution is generated from a small sequence of iterations
166 of the sub-algorithm CGA based. Immediately after the initial solution, the sub-algorithm
167 TS based is used intensively to improve the solutions for the CVRP. The last phase of each
168 global iteration, the constraints for the loading subproblem are evaluated by applying the
169 LNS heuristic based sub-algorithm for packing the items in the routes of the best solution
170 found by the TS. At the end of each loop, the algorithm records the feasible and unfeasible
171 routes in terms of loading in two hashtables respectively. Thus, with these records it
172 avoids generating solutions with unfeasible routes found in previous loops. It also avoids
173 reprocessing routes already evaluated as feasible that are found again in the current best
174 solution. The three-step process is repeated until a time limit is reached or if no feasible

global solution is found. Figure 2 shows a macro flow of the proposed algorithm. The details of each sub-algorithm are explained in the subsections 4.2.2, 4.3 and 4.4.2.

4.1. The algorithm initialization

It is necessary to define parameters when beginning the execution of the algorithm. Some of them are fixed with previously established values and others are calculated from the data of the tested instances, as detailed below. The parameters are linked to the strategies and to the techniques used in the developed method. The improvement strategies used in the routing phase and the initial parameters of the algorithm are detailed in the following.

4.1.1. Improvement strategies and their parameters

To intensify and diversify the local search in the tabu search algorithm, two strategies were employed to obtain better results to solve the routing problem. The first is a list of better solutions, called elite solutions, which is built during the execution of the algorithm [44,45].

The list of elite solutions serves to make random choices, with a certain degree of probability. At certain times during the execution of the TS, one of the best solutions found until that point becomes the current solution. In this way, the generation of new neighbors is promoted through an elite solution. This intensifies the search in regions that can be considered promising. Elite solutions can also be evaluated for loading feasibility in the final stage of the algorithm, but only if the best solution found in the current global iteration is not feasible. The size of the list of elite solutions is identified by the *sizeEL* parameter.

The second strategy for improving the routing problem deals with a dynamic variation in the number of neighbors generated from the current solution. The *numNeighbors* variable is calculated at the beginning of the algorithm's execution, right after obtaining customer data. Stochastically, during the iterations of the TS, this variable can have its value changed within a range of values close to the value initially calculated. This strategy is pseudo coded at line 15 of **Algorithm 2** in subsection 4.3.

A simple strategy to facilitate loading is applied during the routing phase. The first global iteration of the algorithm considers 100% of the vehicle's loading surface area. After each global iteration, with a 50% probability, the vehicle area considered for loading can be reduced by 0.05%. Note the minimum limit is 93%. The variable used to control this restriction is called *percAreaVehicle*. This strategy, applied during the steps of generating the initial solution and during the routing phase, tends to facilitate the feasibility of loading in the last stage of each global iteration of the algorithm.

All percentages used in the strategies described above were chosen after an amount of tests and observations, that is, the best results were found using these values.

4.1.2. Other control parameters

The *maxGen* parameter indicates the maximum number of generations, that is, the maximum iterations to be performed by the genetic algorithm to build the initial solution (Input data - Algorithm 1). The size of the tabu list is fixed and is identified as *sizeTL*. The *maxNoImproveCnt* parameter signals the TS algorithm when it is time to try one of the strategies (intensification/diversification) previously mentioned in Subsection 4.1.1. The *maxNoImproveCnt2End* parameter establishes one of the exit conditions for the TS during the routing phase, indicating the maximum number of consecutive iterations without improvement to end the TS and start the next phase. *MaxLNSTime*, on the other hand, indicates the maximum time, in seconds, for executing the LNS algorithm in the feasibility assessment phase of loading the routes of the best solutions. Finally, *maxGTime* defines the maximum time, in seconds, for the execution of the main algorithm in the search for the best solution.

224 4.2. Solution structures and the initial solution

225 The initial solution is generated from a variant of the genetic algorithm, called the
226 constructive genetic algorithm (CGA) proposed by [46]. Some researchers have already
227 used genetic algorithms to generate an initial population of solutions, such as [47] who used
228 the GA and Push-Forward Insertion Heuristic (PFIH) [48] for this purpose. We decided to
229 use the CGA due to its constructive nature and because we have not found its use in the
230 VRP in the literature. Before detailing the construction of the initial solution, Subsection
231 4.2.1 shows, in detail, the structures used to represent and manipulate the solutions.

232 4.2.1. Solution structures

233 A matrix structure allowed to represent and manipulate the solutions in the memory
234 during the execution of the algorithm. This matrix structure focused on the improvement
235 of performance, mainly in stage two of the algorithm where the tabu search is performed.
236 Figure 3 shows a sample of the solution matrix (S) for the instance 0902 from [15]. In
237 this instance, 8 vehicles are available, and their trips are represented in rows 1 to 8 of the
238 matrix S. The first column, identified by 0, represents the trip id in rows 1 to 8, and the
239 first row (row id=0) is the row of the summarized data of the solution. In the first row of
240 S, there are 25 customers to be visited, as showed in column 1. Column 2 represents the
241 total cost of the solution. Column 3 shows that each vehicle has 48 unit of measurement
242 capacity, and column 4 shows that the surface area for loading is 800. As row 0 represents
243 the summarized data of the solution, columns 5 to c_{max} ($c_{max} = 80$) are marked as empty,
244 i.e. with value -1.

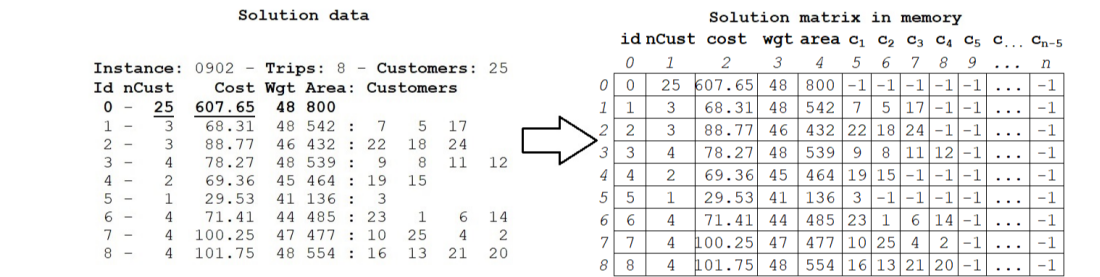


Figure 3. A sample of solution data matrix.

245 Considering this example, rows 1 to 8 have the trip data. These rows contain columns
246 1 to 4, which have the summarized data of each trip, representing the number of customers,
247 the trip cost, the total weight and the total area respectively. For the trips, the customers'
248 ids ate stored in columns 5 to c_{max} of the S matrix, and the order in which they are stored
249 represents the sequence of visits to the customers. Note, the depot is excluded, which is by
250 default, the start and end points in this study.

251 4.2.2. Generating the initial solution using constructive genetic algorithm

252 The CGA is efficient in finding optimal, or near optimal solutions when employed in a
253 variety of problems. Some researchers have already used CGA in their works and obtained
254 excellent results. Among them we can mention [49,50]. One of the main differences of the
255 CGA, when compared to a classical genetic algorithm, is its fitness process [46].

256 To construct the initial solution according to the 2L-CVRP definition, the CGA's
257 principles must be employed. This clustering problem in graphs is stated as the search for
258 partitions on the vertex set V in a predefined number of clusters normally indicated by the
259 quantity of available vehicles. A general overview of the CGA framework is provided by
260 the pseudo code in the Algorithm 1.

Algorithm 1 Constructive genetic algorithm - CGA**Input:** maxGen // maximum number of generations (max iterations)**Output:** initial solution

```

1: let currGen = 0; // initialize the current generation number
2: define initPSize;
3: Generate an initial population  $P_0$  with size initPSize;
4: Evaluate the population  $P_0$ ;
5: while currGen < maxGen do
6:   Select  $P$ ; //  $P$  is the population of routes
7:   Recombination  $P$ ;
8:   Evaluate  $P$ ;
9:   Evaluate the loading feasibility of  $P$  in terms of weight and area;
10:  let currGen = currGen + 1;
11: end while
12: return the best feasible solution found;

```

261 The size of the initial population P_0 is defined as parameter *initPSize* and currGen
 262 counts the number of generations (line 1 and 2 of **Algorithm 1**). To define the P_0 of
 263 individuals, all the arcs (i, j) comprised by the set of edges have their associated costs $c_{i,j}$
 264 and are sorted by the increasing value of cost $c_{i,j}$ to be electable as the new routes (one for
 265 each available vehicle) (i.e., the initial edge of clusters that in some way attract the other
 266 edges that participate in the representation). Figure 4 shows an exemple of the ranking
 267 matrix of Euclidean distances between the customers.

	1	2	3	4	5	6	7
1	-	2.00	5.39	4.24	6.71	4.12	3.16
2		-	3.48	3.16	5.00	2.24	3.16
3			-	3.61	3.16	1.41	5.00
4				-	3.00	4.12	2.00
5					-	4.47	5.00
6						-	5.00
7							-

Figure 4. Exemple of ranking of Euclidian distance between the customers.

268 Using the ranking of distances, the closest customers are combined to form the initial
 269 population P_0 (line 3 of **Algorithm 1**) (Figure 5).

#	#	#	3	6	#	#
1	2	3	4	5	6	7
#	1	2	#	#	#	#
1	2	3	4	5	6	7
#	#	#	#	4	7	#
1	2	3	4	5	6	7
#	#	#	2	6	#	#
1	2	3	4	5	6	7

Figure 5. Initial population of the CGA framework.

270 This initial population creates new generations through an iterative process of evaluation
 271 until convergence criteria are met to reach an optimal solution. For each $p \in P_0$, the
 272 fitness is calculated based on the distance between the customers. Equation (1) calculates
 273 the fitness (line 4 of **Algorithm 1**).

$$\text{Fitness} = \frac{\sum \text{distance in the route}}{\text{number of customers}} \quad (1)$$

274 From that point, until the stop condition is reached, and a valid solution is found, the
 275 steps of selection, recombination (crossover operation), and evaluation are performed. This
 276 ensures that other customers are selected successively to be inserted into the routes, re-
 277 specting the genetic operators necessary to maintain the adaptation characteristics acquired
 278 by previous generations.

279 The selections (line 6 of **Algorithm 1**) are performed according to the number of
 280 vehicles, which establishes the number of routes. Figure 6 shows the selection process of
 281 the CGA framework. Recombination is performed for each route, inserting the closest
 282 customers outside of the route. Figure 7 shows the recombination process of the CGA
 283 algorithm (line 7 of **Algorithm 1**). Then, the population is evaluated once again using the
 284 fitness Equation (1) (line 8 of **Algorithm 1**).

#	#	#	3	6	#	#	Route 1
#	1	2	#	#	#	#	Route 2
#	#	#	3	6	#	#	Route 1
#	#	#	#	4	7	#	Route 2
#	1	2	#	#	#	#	Route 1
#	#	#	#	4	7	#	Route 2
⋮							

Figure 6. Select population of the CGA framework.

#	#	#	3	6	5	#	Route 1
7	1	2	#	#	#	#	Route 2
#	#	#	3	6	5	#	Route 1
#	#	#	2	4	7	#	Route 2
#	1	2	6	#	#	#	Route 1
#	#	#	5	4	7	#	Route 2
⋮							

Figure 7. Recombine population of the CGA algorithm.

285 Since the feasibility of a solution is determined mainly by the load constraints of the
 286 problem, the proposed constructive algorithm checks the items of all the customers in each
 287 route that can be loaded into the vehicle when considering loading constraints, like the
 288 maximum load surface and the maximum vehicle weight (line 9 of **Algorithm 1**). This
 289 strategy managed to successfully construct feasible initial solutions for all of the CVRP
 290 instances (Figure 8).

-	-	4	7	1	2	-	Route 1
-	3	5	6	-	-	-	Route 2

Figure 8. Initial solution of the CGA algorithm.

291 The stop condition is triggered at a predefined number of generations. The population
292 increases after the initial generations, continues growing until reaching an upper limit and
293 decreases for higher values of the evolution parameter. The structure corresponding to the
294 best problem solution must be kept in the process.

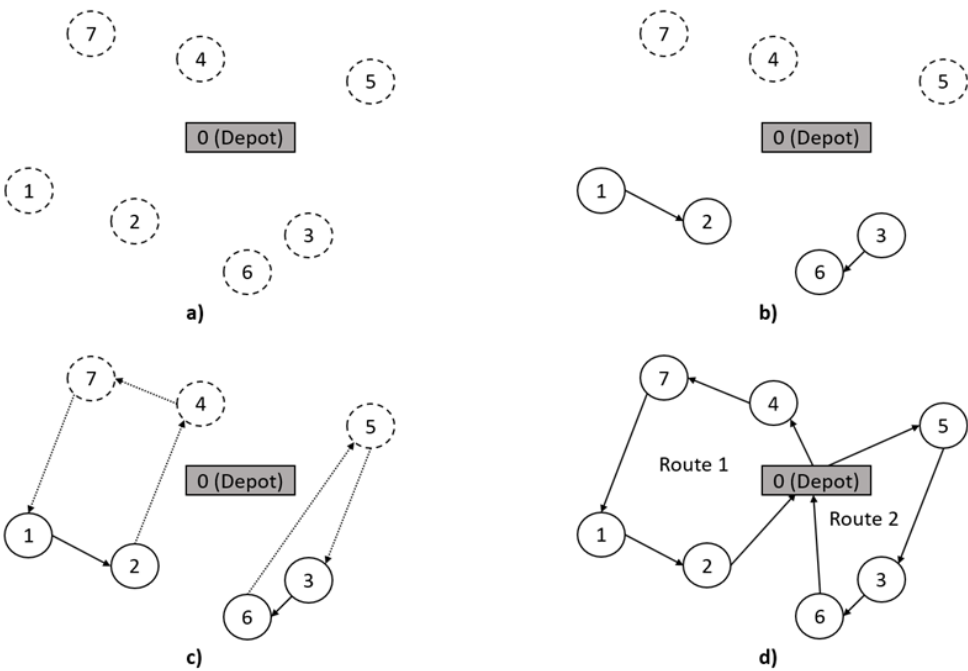


Figure 9. Initial solution construction through the CGA framework edge/vertex insertion process.

295 Figure 9 depicts an example of the CGA framework steps where the filled rectangle
296 represents the central depot (vertex 0), the empty circles correspond to the customers,
297 the arrows represent the routes (i, j) , the dashed circles represent the vertices selected
298 for insertion, and the dotted arrows show possible insertion edges. The vehicle loading
299 constraint check is integrated into the insertion process.

300 4.3. Tabu search algorithm for routing problem

301 The TS is a well-known metaheuristic that is widely used to solve combinatorial
302 analysis problems. The TS method stands out for being reasonably easy implement and for
303 producing very satisfactory results. Several researchers [15,17,33,51–54] have successfully
304 used this technique, alone or combined with another method to solve some variant of the
305 VRP.

306 According to Glover [42], the TS consists of a local search in the neighborhood of the
307 current solution that was generated from movements carried out from that solution. To
308 escape from local optima, some movements are considered prohibited. These movements
309 are recorded in the memory. The structure that stores these movements is a finite-sized list
310 called the tabu list. These movements are prohibited from being carried out until they cease
311 to exist in the list. This takes place as iterations occur and the list is updated. The process
312 is done iteratively. With each iteration, the best neighbor must be selected to become the
313 current solution and generate a new neighborhood. This is true even if that neighbor is
314 worse than the best solution found until that point. The process is repeated until a stop
315 condition is reached.

316 In this work, the CGA and TS heuristics are applied to find better solutions to the
 317 capacitated vehicle routing problem (CVRP). **Algorithm 2** shows TabuSearch method that
 318 starts right after the CGA builds an initial solution for the CVRP. The initial solution
 319 becomes the first current solution of the tabu search and the best local solution as well.

Algorithm 2 Tabu search algorithm for the CVRP

Input: initialSolution

Output: bestSolution

```

1: //Global variables maxNoImproveCnt, maxNoImproveCnt2End, sizeTL, sizeEL
2: //numNeighbors, maxLNSTime and maxGTime are set at the beginning of the algorithm
3: currSolution = initialSolution;
4: bestSolution = initialSolution;
5: noImproveCnt = 0;
6: noImproveCnt2End = 0;
7: numCandidates = numNeighbors;
8: maxTime = maxGTime - maxLNSTime;
9: Define arrays tabuList[sizeTL] and eliteList[sizeEL];
10: while noImproveCnt2End < maxNoImproveCnt2End and currTime < maxTime do
11:   Put currSolution in tabuList[], considering FIFO method when tabuList[] is full;
12:   if noImproveCnt >= maxNoImproveCnt then
13:     noImproveCnt = 0;
14:     With a probability of 40% set currSolution = bestSolution, or with a probability
       of 30% set currSolution = one of the eliteList[] randomly or with a probability of 30%
       stay the same;
15:     With a probability of 25% set r = random number between -n and +n, where
       n=(numNeighbors/2) and set numCandidates = numNeighbors + r or with a probab-
       ility of 75% stay the same;
16:   end if
17:   Select a neighborhood structure NS randomly;
18:   Generate candidateList[numCandidates] from currSolution using NS;
19:   //Each candidate in candidateList[] must be feasible in terms of area and weight
20:   //For loading feasibility, hash tables irHT and frHT are checked
21:   //Each candidate in candidateList[] must not be in the tabuList[]
22:   currSolution = best candidate of candidateList[];
23:   if cost(currSolution) < cost(bestSolution) then      ▷ function cost() gets the total
       distance of all routes in a solution structure
24:     bestSolution = currSolution;
25:     noImproveCnt = 0;
26:     noImproveCnt2End = 0;
27:   else
28:     Increase noImproveCnt and noImproveCnt2End by 1;
29:   end if
30:   if eliteList[] has empty entry or cost(currSolution) < worst cost(eliteList[]) then
31:     Put currSolution into the eliteList[] at empty entry or replace the worst entry;
32:   end if
33: end while

```

320 A tabu list (*tabuList*) and a list of elite solutions (*eliteList*) are initialized to their
 321 predefined sizes at the beginning of the algorithm, as described in Subsection 4.1.1. The
 322 *tabuList* keeps the last solutions found to temporarily prohibit them from becoming the
 323 current solution. The *eliteList* stores the latest *sizeEL* best solutions found.

324 The *eliteList* has two purposes. The first is to provide, at certain times, the return of
 325 one of the best solutions previously registered as the next current solution. This purpose
 326 serves to intensify the search in promising regions of the search space [55]. The second

purpose is for the best solutions found in the routing problem (CVRP) to be evaluated for loading feasibility (2L-CVRP) in the next stage of the framework. This is only carried out if the local best solution is not feasible.

A local search cycle is initiated until the stop condition is true. The main condition for stopping the tabu search is when a maximum consecutive number of iterations without improvement is achieved ($noImproveCnt2End \geq maxNoImproveCnt2End$). Another stop condition is if the maximum global time minus the time for load evaluation is reached ($currTime \geq maxGTime - maxLNSTime$).

At the beginning of each iteration, the current solution is placed on the tabu list. As the tabu list has a predefined size ($sizeTL$), when it is filled, the new entries follow a first in, first out method (FIFO). From the current solution, a neighborhood is generated by following a pattern of neighborhood structures as detailed in Subsection 4.3.1. The size of the neighborhood is initially defined by the $numNeighbors$ variable. After the generation of the neighborhood, a local search is carried out to find the best candidate solution to become the next current solution. As already mentioned, in this choice, the neighborhood solutions cannot be on the $tabuList$ and must have feasible routes in terms of weight and area. The best neighborhood solution is then selected and becomes the new current solution for the next iteration, even if it is worse than the most recent current solution.

To speed up the loading evaluation process, two hash table structures (HT) are used to store routes that have already been evaluated in stage 3 of the algorithm by the LNS. Toffolo et al. [56], Zachariadis et al. [27], Leung et al. [19] and Wei et al. [25] have used similar strategies. One structure registers the feasible routes ($frHT$) already evaluated and the other registers the infeasible routes ($irHT$). During the selection of the best neighbor in the tabu search, the HT structures are consulted to previously check the feasibility or infeasibility of the solutions found.

4.3.1. Neighborhood structures

Like Wei et al. [25], we used four types of neighborhood structures in our study as shown in Figure 10. When selecting one of the four types of structures, changes in the current solution can be applied to a single route or a pair of routes. The selection of the route or the pair of routes to apply the movements from a structure is random. Subsection 4.3.2 provides more details on how structures are used to generate neighbors for the current solution.

The first type of structure is customer relocation. In this type of movement, a customer is relocated to another position within the same route or is relocated to another route (Figure 10-a). These movements are known as *intra-shift* and *inter-shift*, respectively in Wei et al. [26]. The second type is a customer exchange (Figure 10-b), where there is an exchange of positions between two customers on the same route or the exchange of two customers between two different routes within the solution (*intra-swap* and *inter-swap* according to Wei et al. [26]). The third structure is a *routeinterchange* (Figure 10-c). If applied to only one route, two positions are randomly selected and the customers between these two positions are repositioned in reverse order (*intra-2opt* in Wei et al. [26]). If applied to a pair of routes, where a customer on each route is randomly chosen, the initial portion of the first route to the customer's position is connected to the final portion of the second route from the position of the other customer, and vice versa. The last structure is called *blockexchange* (Figure 10-d), where there is an exchange of two route segments that can randomly be 1 to 3 in size.

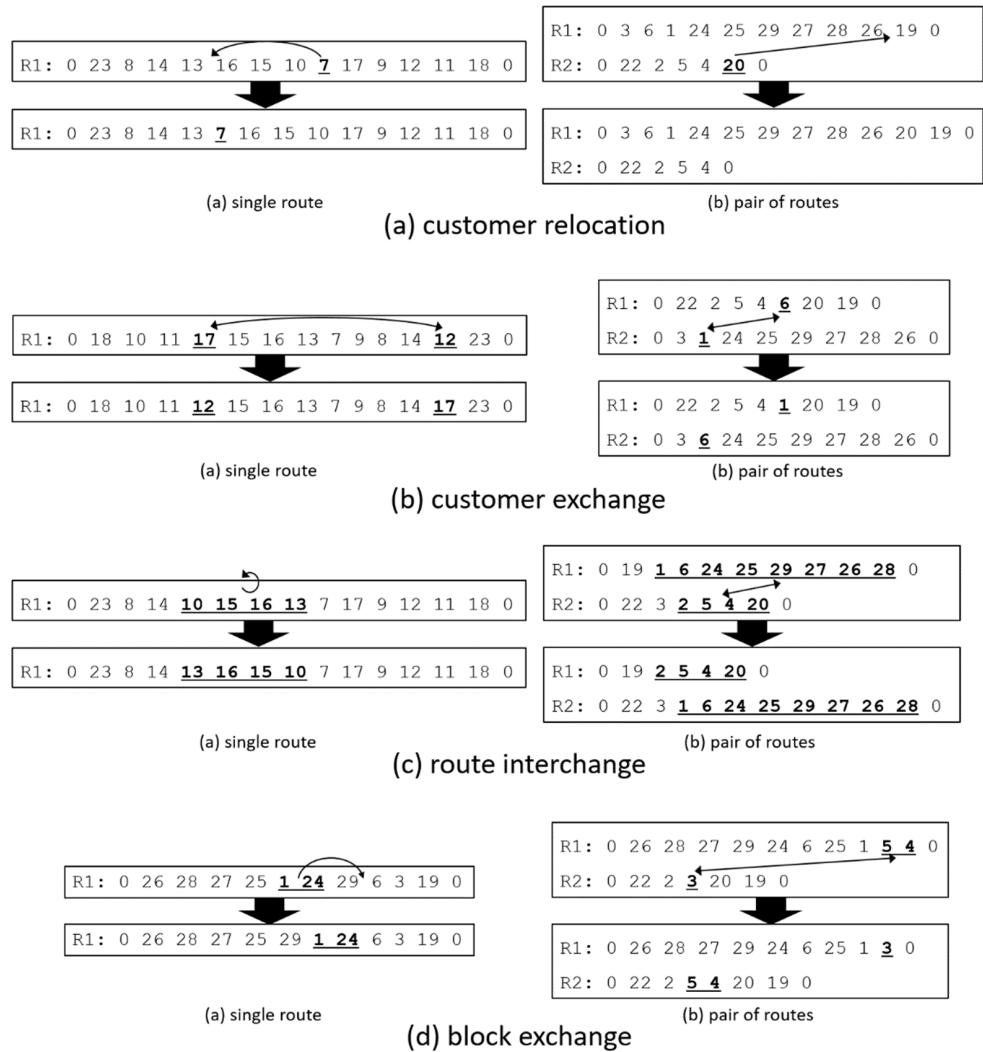


Figure 10. The neighborhood structures.

4.3.2. Generating neighbors as candidate solutions for the routing problem

Each iteration of the tabu search calls for a procedure that generates new candidate solutions from the current solution. One of the four structures mentioned in Subsection 4.3.1 is selected randomly with an equal probability to be applied to the current solution and generate a candidate solution. The feasibility of each candidate solution is tested in terms of the total weight and total area of the load, observing the strategy of the area of the vehicle considered for loading. This is controlled by the *percAreaVehicle* variable. Also, for not to waste time, the routes of the solutions already analyzed regarding the feasibility of loading by the LNS are consulted in the hash tables *irHT* and *frHT*. If an infeasible route is generated and it is in the *irHT*, then it is discarded. On the other hand, if a feasible route is generated and it is in the *frHT*, it is then kept, even if the area of its load exceeds the maximum considered area for loading, this is controlled by the *percAreaVehicle*. The latter conditional is applied when the parameter *percAreaVehicle* < 100%. Thus, the best candidate is selected to be the next current solution and the procedure is carried out successively to generate new neighbors in search of the best solution.

4.4. The loading procedure

For the loading sub-problem, we developed the main procedure given by **Algorithm 3** (*EvalSolutionLoading*). This procedure is responsible for carrying out assessments regarding the feasibility of loading the best solution found by the tabu search in the routing stage, as well as elite solutions if necessary.

Algorithm 3 EvalSolutionLoading**Input:** eliteList[], bestRoutingSolution**Output:** Solution evaluation procedure for loading

```

1: Define array evalList[sizeEL + 1];
2: Put bestRoutingSolution and eliteList[] entries into evalList[];
3: Sort evalList[] by cost ascending;
4: for each solution entry in the evalList[] do
5:   for each route in the solution do
6:     if route is in frHT then
7:       Sign the route as feasible;
8:     else
9:       Start new thread with procedure to evaluate route loading in parallel:
10:      Set items[] = all items demanded by customers in route;
11:      isFeasible = LSNpack(items[]);
12:      if isFeasible then
13:        Sign route as feasible and insert it into frHT;
14:      else
15:        Sign route as infeasible and insert it into irHT;
16:      end if
17:      End thread;
18:    end if
19:  end for
20:  Wait until all threads end;
21:  if all routes are feasible then
22:    if evaluated feasible solution is better than globalBestSolution or the globalBest-
      Solution is null then
23:      Update globalBestSolution;
24:    end if
25:    Break the loop;
26:  end if
27: end for

```

393 To verify the feasibility of loading each route of the evaluated solutions, the Eval-
 394 SolutionLoading algorithm calls a procedure based on the Large Neighborhood Search,
 395 denoted as **Algorithm 4** (*LNSpack*). The code of LNSpack was partially based on the work
 396 published by Erdoğan [57]. The two procedures are described in the next subsections.

Algorithm 4 LNSpack**Input:** items[]**Output:** Large Neighborhood Search for 2BPP

```

1: for sortType in {area, circumference} do
2:   Sort items[] by sortType descending;
3:   Use the first-fit-decreasing heuristic to repack the removed items;
4:   while elapsedTime < maxLNSTime do
5:     Perturb solution: randomly remove items from vehicle;
6:     Use the first-fit-decreasing heuristic to repack the removed items;
7:     if all items were packed then
8:       return success;
9:     if new solution is better than the best-known solution then
10:      Update best-known solution;
11:    else
12:      Revert the best-known solution;
13:    end if
14:  end if
15:  Set elapsedTime;
16: end while
17: end for
18: return failure

```

397 4.4.1. A procedure to evaluate the loading feasibility of the routing solutions

398 The *EvalSolutionLoading* algorithm receives two parameters: the best solution (*bestRoutingSolution*) and the elite solutions (*eliteList[]*) found in the routing phase. Then these
399 solutions are put on a list to be evaluated (*evalList[]*). The list is ordered from lowest to
400 highest cost so that the best solutions are evaluated first. As soon as a solution with feasible
401 loading is found, the evaluation of the other solutions does not proceed. In the evaluation
402 of each solution registered in *evalList[]*, the verification of the feasibility of loading for
403 each route is performed by the LNSpack algorithm (**Algorithm 4**), which is discussed in
404 Subsection 4.4.2. To save time and processing, the first step in evaluating the route is to
405 check if it already exists in the *frHT*, that is, to check if it has already been evaluated and
406 identified as feasible. In that case, there is no need to evaluate it again. The routes that are
407 already identified as infeasible are no longer generated during the routing process.

409 For the evaluation of the routes, a multithreading mechanism was implemented to en-
410 hance the performance of the algorithm. Thus, several routes are evaluated simultaneously
411 within a time limit which was defined in the algorithm's initial parameters. Therefore, the
412 power of parallel processing is exploited using multiple CPU cores. Within each thread,
413 the LNSpack heuristic is called to evaluate the route loading. The LNSpack determines
414 whether the route is feasible or not. Then the hash tables *frHT* and *irHT* are updated,
415 respectively. After all the route evaluation threads are finished, the solution is flagged as
416 feasible if all routes are feasible. If this is the case, and, if the solution is better than the
417 best global solution found or if a better global solution still does not exist, the best global
418 solution is updated, ending the procedure and returning to the global algorithm flow.

419 4.4.2. Large Neighborhood Search for two-dimensional bin packing

420 In this work, a version of a large neighborhood search (LNS) algorithm, which is
421 denoted as **Algorithm 4** (LNSpack), was developed to solve the 2BPP. The LNSpack
422 requires a list of rectangular items demanded by all customers in the route as parameters.
423 When the LNSpack is called by the procedure for evaluating the feasibility of a route, firstly,
424 the items are sorted and then the First-Fit Decreasing heuristic [58] is used to try to load
425 the vehicle. Two forms of ordering are applied: the first is by the area and the second is by
426 the circumference of the item. Since in this work, only the 2|UR|L was considered, the

ordering of the items by area or by circumference is quite effective. Therefore, orders by the height or width of the items were not used. Then an iteration loop is initiated where the solution is disturbed by the random removal of some loaded items. Next, the First-Fit Decreasing heuristic is performed again to try to reload them. If all items have been loaded, the procedure has been completed with success. Otherwise, it checks if the loaded area of the current solution is larger than the area of the best solution found and updates the best solution. If the current solution is not better, the best solution is once again the current solution for the next iteration. The condition for stopping the iterations is finding the feasible packaging route items or reaching the maximum time ($maxLNSTime$). If it is the case, the procedure returns as a packaging failure.

5. Results

The algorithm proposed in this study, called the CGA-TS-LNS, was coded in C and the tests were performed on a virtual machine configured as an Intel 2 x Deca Core Xeon E5-2640 CPU with a clock speed of 2.40 giga-hertz and 16 gigabyte of RAM running on a Windows Server 2019 Standard edition operating system. For the benchmarks, the tests were carried out with the well-known instances of Gendreau et al. [15]. The database is composed of 180 instances that are divided into five classes according to the characteristics of the items. Class 1 is characterized as the pure CVRP, where each customer is associated with only one item with a unit of width and a unit of length, and there are no loading restrictions. In Classes 2-5, the quantity of items demanded for each customer i is generated by an uniform distribution within a given interval, according to Column 2 of Table 1. There are 3 categories in which the items can be classified according to their forms: vertical, homogeneous and horizontal. The dimensions of the items ($w \times l$) are uniformly distributed according to the ranges established for the categories of the items (Columns 3-8).

Table 1. Characteristics of items in Classes 2-5 instances.

Class	m_i	Vertical Length	Width	Homogeneous Length	Width	Horizontal Length	Width
2	[1,2]	[0.4L, 0.9L]	[0.1W, 0.2W]	[0.2L, 0.5L]	[0.2W, 0.5W]	[0.1L, 0.2L]	[0.4W, 0.9W]
3	[1,3]	[0.3L, 0.8L]	[0.1W, 0.2W]	[0.2L, 0.4L]	[0.2W, 0.4W]	[0.1L, 0.2L]	[0.3W, 0.8W]
4	[1,4]	[0.2L, 0.7L]	[0.1W, 0.2W]	[0.1L, 0.4L]	[0.1W, 0.4W]	[0.1L, 0.2L]	[0.2W, 0.7W]
5	[1,5]	[0.1L, 0.6L]	[0.1W, 0.2W]	[0.1L, 0.3L]	[0.1W, 0.3W]	[0.1L, 0.2L]	[0.1W, 0.6W]

5.1. Initial parameters

After carrying out experiments with the algorithm in 15 instances of the pure CVRP, with low, medium and high complexities, we observed the results from the use of values from 5 to 50 for the size of the $tabuList[]$. We chose to set $sizeTL=8$, because with this value we obtained the best results in the experiments. As it is not a very high value, the performance of the TabuSearch algorithm improved. This is because in each iteration the new current solution must not be in the $tabuList[]$, and for that, the $tabuList[]$ must be checked line by line. So with fewer positions in the list, less computational time is required. The $numNeighbors$ variable, which dynamically defines the number of neighbors during the tabu search, is defined initially from a polynomial function of degree 3. This function was found from a data analysis as described in the following.

Firstly, tests of the algorithm were performed with instances from the pure CVRP (Class 1). Several values between 5 and 200 were randomly assigned to the $textit{numNeighbors}$ for each execution of the framework on the instances chosen for the test. The values of the variable where the best results were obtained for each instance were noted. Then, using Pearson's correlation coefficient, a correlation analysis was performed on the data of the instances in relation to the values of $numNeighbors$ noted. The data showed that the number of customers presented a correlation coefficient of more than 80% in relation to

the values of *numNeighbors* noted. Therefore, the number of customers together with the values of *numNeighbors* noted in the tests was chosen to obtain the polynomial regression function. The function $y = \lceil 0.00002x^3 - 0.009x^2 + 1.573x - 18.394 \rceil$, where $x \in \mathbb{Z} \mid x \geq 15$ and $y \in \mathbb{Z}$, $y = \text{numNeighbors}$ and x is the number of customers, was obtained through polynomial regression of degree 3, as shown in the graph of Figure 11. Also, it was evident that the function found for values of *numNeighbors* must be used for problems with more than 15 customers, otherwise negative numbers would be obtained. For problems with fewer than 15 customers, *numNeighbors* is limited to a minimum of 5.

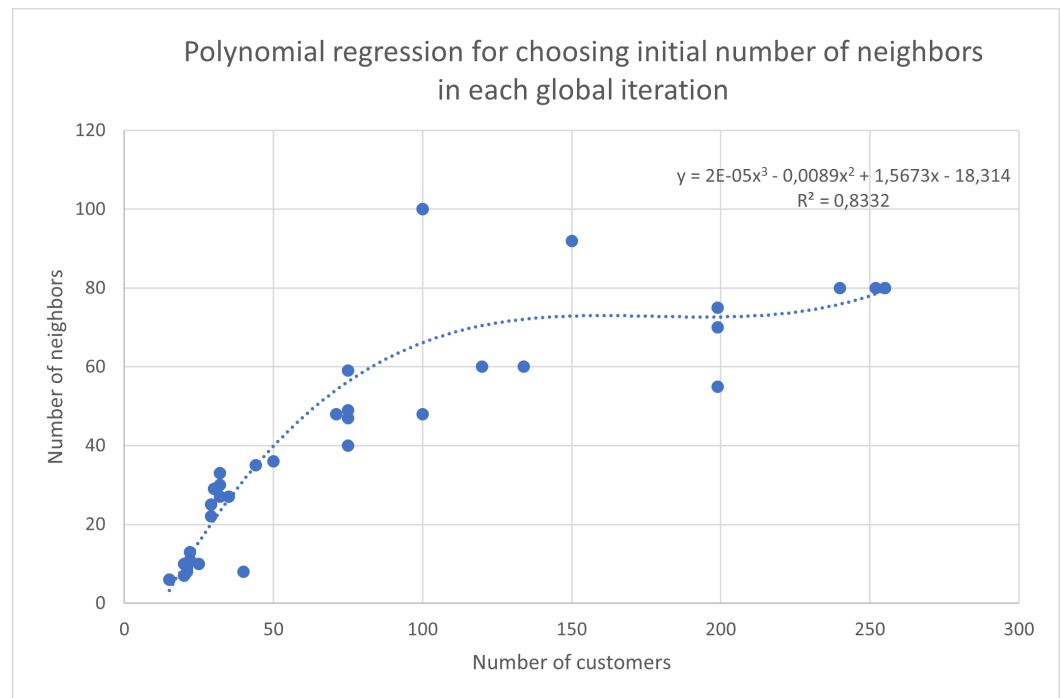


Figure 11. Polynomial regression to get the function to set *numNeighbors* in the TS algorithm.

The *sizeEL* that defines the size of the elite solution list was set to 20. The *percAreaVehicle* variable was initially set to 100%, as described in Subsection 4.3.1 and it was applied to resolve instances of Classes 2-5. This strategy proved to be more effective for routes in which the items to be loaded had very irregular dimensions. Although it can be considered a simple strategy that can go against the idea of maximizing the loading space of the vehicle used, in general it has shown good results. It is important to note that due to the tests carried out in our experiments, this strategy is not very efficient in the largest and most complex instances. This is because a lot of time is needed to achieve promising results. The parameters for stopping conditions of the stages of the algorithm as well as the global stopping condition parameter were defined as follows: The parameter *maxNoImproveCnt2End*, which is one of the stopping conditions of the TabuSearch in the routing phase, was set to 400,000 iterations. Namely, the tabu search algorithm implemented in this framework can perform thousands of iterations per second. The parameter *maxNoImproveCnt* was set to 40,000. In our tests, generally few iterations of the CGA were necessary to generate promising initial solutions, so the *maxGen* parameter was set to 20. The *maxLNSTime* parameter was set to 60 seconds. The *maxGTime* was set to 7,200 seconds. The vehicle fleet was considered homogeneous and the dimensions of the loading area were $H = 40$ and $W = 20$.

5.2. Results comparison

This work solved the pure CVRP (Class 1) and the 2|UR|L version of the 2L-CVRP (Classes 2-5), where the rotation of the items is allowed, and the loading is unrestricted. We compared our results for Class 1 with 4 of the best previous approaches: PRMP [22], VNS

[24], SA [25] and BR-LNS [59]. It is important to remark that after an extensive research, we found only these four published studies that handle with 2|UR|L-CVRP which is the focus of our work, and all of them did not implement parallelism in their approaches. To solve the 2|UR|L we found only four approaches with which to compare the results: ACO by Fuellerer et al. [16], MS-BR by Dominguez et al. [23], xPRMP by Zachariadis et al. [27] and SA by Wei et al. [25]. The framework was run 10 times for each instance with 1-10 random seeds, as was done in several previous works, and the best solutions were compared to the previous approaches.

The running time to find best cost values for all instances can be considered compatible with the previous works. The running time for each one of the 180 instances remains less than the maximum global time limit to run the algorithm, i.e., less than 7,200s. For the pure CVRP our method proved to be more effective and the running time to find best solutions in this class was less than the time informed by the other previous approaches. For the 2L-CVRP our algorithm takes a little more time running, but it was able to find some unpublished results.

5.3. Results for Class 1 instances

Table A1 shows the results for the Class 1 instances, where the best costs are compared with the other four previous frameworks. The CGA-TS-LNS found better and higher quality solutions for the 6 most complex instances (30-36) and matched the BKS for 28 instances. In addition, Figure 12 shows the average cost of the solutions found by the CGA-TS-LNS for the 36 instances of Class 1 is lower than all previous approaches, and lower than the average cost of all the BKS. This demonstrates the effectiveness of our algorithm to solve the CVRP.

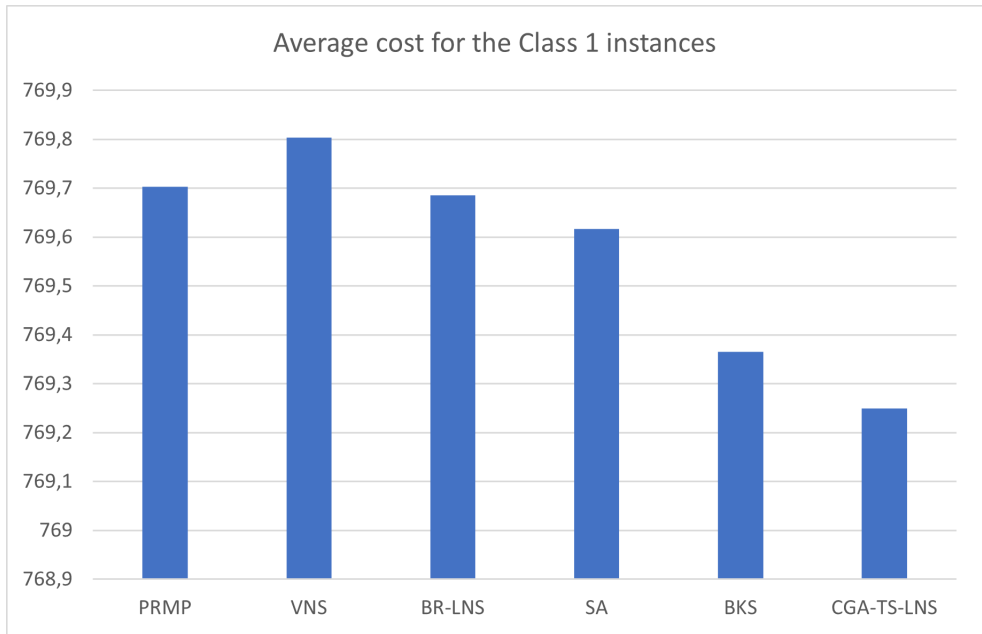


Figure 12. The average cost for the Class 1 instances.

5.4. Results for 2|UR|L instances

When comparing our results for the 2|UR|L in Classes 2-5, we find that our average cost in each class is better than two of the only four algorithms proposed to solve this version found in the literature. In total, the CGA-TS-LNS surpassed the BKS in 7 of the 144 instances and matched in another 66. The CGA-TS-LNS only loses to the ACO and MS-BR in 5 instances of 144, with 4 instances of Class 2 and 1 instance of Class 4. It was evident that for the 2L-CVRP, the SA approach is still the best, achieving the lowest average cost.

Table A2 compares the best 2|UR|L results for the Class 2 instances. In comparison with the other four approaches and the BKS of the 36 instances, the CGA-TS-LNS found

better solutions for 2 instances and matched the other 15 better solutions. Considering the average cost, xPRMP is the best framework to solve the Class 2 instances.

Table A3 compares the best results from the 2|UR|L over the 36 instances of Class 3, where our CGA-TS-LNS found better higher quality solutions for 3 instances and matched 18 other better solutions. Considering the average cost for 36 instances, SA is the best approach for this Class.

Table A4 compares the best results of the 2|UR|L for the 36 instances of Class 4. The CGA-TS-LNS found better higher quality solutions for 1 instance and matched the other 14 best solutions. Again, the SA algorithm is the best to solve Class 4 instances.

Table A5 compares the best results of the 2|UR|L for the 36 instances of Class 5. Our algorithm found better higher quality solutions for 1 instance and matched 19 other better solutions. One more time, the SA algorithm is the best approach to solve the Class 5 instances.

5.5. Statistical validation for algorithm comparison

According to [60] inside the field of inferential statistics, hypothesis testing can be employed to draw inferences about one or more populations of given samples (results). To perform that, two hypotheses, the null hypothesis H_0 and the alternative hypothesis H_1 , are defined. The null hypothesis is a statement of no effect or no difference, whereas the alternative hypothesis represents the presence of an effect or a difference (in our case, significant differences between algorithms). When applying a statistical procedure to reject a hypothesis, a level of significance is used to determine at which level the hypothesis may be rejected. The Sign test for multiple comparisons described in [60] allows us to highlight those algorithms whose performances are statistically different when compared to the CGA-TS-LNS algorithm.

As defined in [60], when using a level of significance $\alpha = 0.10$ and setting our hypotheses to be $H_0: M_j \geq M_1$ and $H_1: M_j < M_1$ and $m = 4$ ($m = k - 1$) and $n = 36$, that is, our CGA-TS-LNS algorithm performs significantly better than the remaining algorithms. Also, according to Table A.21 of [60], for $m = 4$ ($m = k - 1$) and $n = 36$ reveals that the critical value of R_j is 10.

Tables A6 to A10 present the number of times the CGA-TS-LNS algorithm was superior to its competitors, as well as the number of occasions on which the CGA-TS-LNS algorithm obtained the optimal solution. In tables A1 to A5, the best known solution from the literature (BKS) is the optimal solution to the problem and, therefore, it is impossible to obtain better solutions. Thus, we add the number of times the GGA-TS-LNS algorithm was better to the competitors to the number of times the BKS was obtained.

Table A6 shows the performance comparison of GGA-TS-LNS with PRMP, VNS, BR-LNS and SA algorithms for the instances of Class 1 (pure CVRP). Since the number of minuses in the pairwise comparison between the GGA-TS-LNS algorithm and all others is equal to 36, we can conclude that GGA-TS-LNS has a significantly better performance than all of other competitors.

Table A7 shows the performance comparison of GGA-TS-LNS with ACO, MS-BR, SA and xPRMP algorithms for the 2|UR|L instances of Class 2. Since the number of minuses in the pairwise comparison between the GGA-TS-LNS algorithm and ACO and MS-BR algorithms is equal to 34 and 25, respectively, we can conclude that GGA-TS-LNS has a significantly better performance than these two competitors.

Table A8 shows the performance comparison of GGA-TS-LNS with ACO, MS-BR, SA and xPRMP algorithms for the 2|UR|L instances of Class 3. Since the number of minuses in the pairwise comparison between the GGA-TS-LNS algorithm and ACO and MS-BR algorithms is equal to 35 and 27, respectively, we can conclude that GGA-TS-LNS has a significantly better performance than these two competitors.

Table A9 shows the performance comparison of GGA-TS-LNS with ACO, MS-BR, SA and xPRMP algorithms for the 2|UR|L instances of Class 4. Since the number of minuses in the pairwise comparison between the GGA-TS-LNS algorithm and ACO and MS-BR

584 algorithms is equal to 36 and 30, respectively, we can conclude that GGA-TS-LNS has a
585 significantly better performance than them. Also, in the pairwise comparison between the
586 GGA-TS-LNS algorithm and XPRMP algorithm, with a level of significance $\alpha = 0.05$, we
587 can conclude that GGA-TS-LNS has a significantly better performance than it as well.

588 Table A10 shows the performance comparison of GGA-TS-LNS with ACO, MS-BR, SA
589 and xPRMP algorithms for the 2|UR|L instances of Class 5. Since the number of minuses
590 in the pairwise comparison between the GGA-TS-LNS algorithm and ACO and MS-BR
591 algorithms is equal to 36 and 34, respectively, we can conclude that GGA-TS-LNS has a
592 significantly better performance than these two competitors.

593 **6. Conclusion**

594 This paper presents a new hybrid algorithm to solve the 2L-CVRP. The combination of
595 algorithms based on the CGA, the TS, and the LNS proved to be competitive when applied
596 to the 2L-CVRP. Our results outperformed two of the only four algorithms proposed, so
597 far, to solve version 2|UR|L of this problem. As stated by Wei et al. [25], it is not efficient
598 to simply combine conventional algorithms for the CVRP and the 2BPP, so some loading
599 strategies need to be employed to facilitate the solution of these integrated problems. In our
600 case, a simple strategy was to consider the loading area dynamically during the iterations
601 of the algorithm. This was combined with the diversification and intensification strategies,
602 plus the use of multithreading to evaluate the loading of the routes. This facilitated the
603 process to find appropriate solutions for the 2L-CVRP, but it was not enough to outperform
604 the algorithms proposed by Wei et al. [25] and by Zachariadis et al. [27].

605 When analyzing the results for the pure CVRP, the framework proved to be extremely
606 efficient, surpassing all other approaches that were applied to the instances proposed by
607 Gendreau et al. [15]. This study brings, as a reference for future works, new best-known
608 solutions for the most complex instances of the pure CVRP and a better average of solutions
609 for the 36 instances of this Class.

610 The algorithm proposed in this work has great potential, but it can still be improved.
611 For future works, we can think about the use of other strategies to improve solutions
612 regarding the loading sub-problem. The ability to solve types 2|OU|L, 2|SO|L and
613 2|SR|L can be added to the algorithm. The algorithm can also be adapted to solve other
614 problems, such as the 3L-CVRP. Additionally, it can be tested in other databases. Also,
615 to provide a more practical information measure it is possible to use average values to
616 compare the results with other researches.

617 The main application would be to solve the CVRP and to compare the results and the
618 effectiveness. In addition, we believe that the algorithm can serve as a basis for other studies
619 to obtain improvements in logistic and transport processes. Its application can contribute
620 to reducing costs in the routing problems of capacitated vehicles in the real world. In this
621 sense, the integration of this algorithm with intelligent models for logistics management
622 [61] will improve the routes processing and consequently the safety of transportation. The
623 algorithm also will allow the implementation of ubiquitous intelligent services for vehicular
624 users [62]. Finally, future work will integrate the algorithm with strategies used to treat
625 Context Histories [63–66] such as pattern analysis [67], context prediction [68], similarity
626 analysis [69], cryptography [70] and IoT challenges in smart environments [71,72]. This
627 integration will support better solutions, specifically, to routing problems and in general
628 for problems involving context histories.

629 **Author Contributions:** Conceptualization, L.P.F and J.C.F; Investigation, L.P.F and J.C.F; Methodol-
630 ogy, L.P.F and J.C.F; Software, L.P.F and J.C.F; Project Administration, L.P.F and J.C.F; Supervision,
631 L.P.F and J.C.F; Validation, L.P.F and J.C.F; Writing—original draft, L.P.F and J.C.F; Writing—review
632 and editing, J.C.F, G.A.H., J.L.V.B., S.D.C., M.B. and V.R.Q.L.; Financial, S.D.C., M.B. and V.R.Q.L. All
633 authors have read and agreed to the published version of the manuscript.

634 **Funding:** We would like to thank to Seed Funding ILIND—Instituto Lusófono de Investigação e De-
635 senvolvimento, project COFAC/ILIND/COPELABS/1/2020 and COFAC/ILIND/COPELABS/3/2020.

636 This work was supported in part by the Fundação para a Ciência e a Tecnologia under Project
637 UIDB/04111/2020.

638 **Informed Consent Statement:** This research did not require ethical approval in accordance with the
639 regulations of the University of Santa Cruz do Sul (UNISC).

640 **Acknowledgments:** This study was financed in part by the Coordenação de Aperfeiçoamento de
641 Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. The work described in this paper was
642 supported by Master's in Systems and Industrial Processes of the University of Santa Cruz do Sul. This
643 work was supported by national funds through the Fundação para a Ciência e a Tecnologia, I.P.
644 (Portuguese Foundation for Science and Technology) by the project UIDB/05064/2020 (VALORIZA –
645 Research Centre for Endogenous Resource Valorization).

646 **Conflicts of Interest:** The authors declare no conflict of interest.

647 **Abbreviations**

648 The following abbreviations are used in this manuscript:

2BPP	Two-Dimensional Bin Packing Problem
2 SO L	Two-Dimensional Sequential Oriented Loading Problem
2 SR L	Two-Dimensional Sequential Non-oriented Loading Problem
2 UO L	Two-Dimensional Unrestricted Oriented Loading Problem
2 UR L	Two-Dimensional Unrestricted Non-oriented Loading Problem
2L-CVRP	Two-Dimensional Loading Constraints
ACO	Ant Colony Optimisation
BKS	Best-known Solution
BR-LNS	Biased-Randomised Large Neighbourhood Search
CGA	Constructive Genetic Algorithm
CVRP	Capacitated Vehicle Routing Problem
649 FIFO	First In, First Out
frHT	Feasible Routes Hash Table
GVRP	Green Vehicle Routing Problem
irHT	Infeasible Routes Hash Table
LNS	Large Neighborhood Search
MS-BR	Multi-Start Biased Randomized Algorithm
PRMP	Promise Routing-Memory Packing Problem
SA	Simulated Annealing
TS	Tabu Search
TSP	Travelling Salesman Problem
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem

Appendix A

Table A1. Comparative results on the pure CVRP instances of Class 1.

Inst	BKS	PRMP Cost	Imp	VNS Cost	Imp	BR-LNS Cost	Imp	SA Cost	Imp	CGA-TS-LNS Cost	Imp
1	278.73	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00
2	334.96	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00
3	358.40	358.40	0.00	358.40	0.00	358.40	0.00	358.40	0.00	358.40	0.00
4	430.88	430.88	0.00	430.88	0.00	430.88	0.00	430.88	0.00	430.88	0.00
5	375.28	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00
6	495.85	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00
7	568.56	568.56	0.00	568.56	0.00	568.56	0.00	568.56	0.00	568.56	0.00
8	568.56	568.56	0.00	568.56	0.00	568.56	0.00	568.56	0.00	568.56	0.00
9	607.65	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00
10	535.74	535.80	-0.01	535.80	-0.01	535.80	-0.01	535.80	-0.01	535.80	-0.01
11	505.01	505.01	0.00	505.01	0.00	505.01	0.00	505.01	0.00	505.01	0.00
12	610.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00
13	2006.34	2006.34	0.00	2006.34	0.00	2006.34	0.00	2006.34	0.00	2006.34	0.00
14	837.67	837.67	0.00	837.67	0.00	837.67	0.00	837.67	0.00	837.67	0.00
15	837.67	837.67	0.00	837.67	0.00	837.67	0.00	837.67	0.00	837.67	0.00
16	698.61	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00
17	861.79	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00
18	723.54	723.54	0.00	723.54	0.00	723.54	0.00	723.54	0.00	723.54	0.00
19	524.61	524.61	0.00	524.61	0.00	524.61	0.00	524.61	0.00	524.61	0.00
20	241.97	241.97	0.00	241.97	0.00	241.97	0.00	241.97	0.00	241.97	0.00
21	687.60	687.60	0.00	687.60	0.00	687.60	0.00	687.60	0.00	687.60	0.00
22	740.66	740.66	0.00	740.66	0.00	740.66	0.00	740.66	0.00	740.66	0.00
23	835.26	835.26	0.00	835.26	0.00	835.26	0.00	835.26	0.00	835.26	0.00
24	1024.69	1024.69	0.00	1024.69	0.00	1024.69	0.00	1024.69	0.00	1024.69	0.00
25	826.14	826.14	0.00	826.14	0.00	826.14	0.00	826.14	0.00	826.14	0.00
26	819.56	819.56	0.00	819.56	0.00	819.56	0.00	819.56	0.00	819.56	0.00
27	1082.65	1082.65	0.00	1082.65	0.00	1082.65	0.00	1082.65	0.00	1082.65	0.00
28	1040.70	1042.12	-0.14	1042.12	-0.14	1042.12	-0.14	1042.12	-0.14	1042.12	-0.14
29	1162.96	1162.96	0.00	1162.96	0.00	1162.96	0.00	1162.96	0.00	1162.96	0.00
30	1028.42	1028.42	0.00	1028.42	0.00	1028.42	0.00	1029.79	-0.13	1028.42	0.00
31	1299.21	1299.56	-0.03	1302.48	-0.25	1299.21	0.00	1301.03	-0.14	1297.25	0.15
32	1296.18	1296.91	-0.06	1300.22	-0.31	1296.18	0.00	1300.30	-0.32	1295.87	0.02
33	1296.13	1299.55	-0.26	1298.02	-0.15	1297.50	-0.11	1296.13	0.00	1294.29	0.14
34	708.39	709.82	-0.20	708.39	0.00	709.08	-0.10	708.66	-0.04	708.26	0.02
35	862.79	866.06	-0.38	865.39	-0.30	864.63	-0.21	862.79	0.00	862.61	0.02
36	583.98	585.46	-0.25	586.49	-0.43	590.16	-1.06	583.98	0.00	582.92	0.18
Avg	769.37	769.70	-0.04	769.80	-0.06	769.69	-0.04	769.62	-0.03	769.25	0.01

BKS: Best-known solution from the literature. Bold entries correspond to higher quality solutions.
Imp: Percentage improvement between the cost and BKS ($Imp = 100 \cdot (BKS - cost) / BKS$).

Table A2. Comparative results for the 2|UR|L instances of Class 2.

Inst	BKS	ACO Cost	Imp	MS-BR Cost	Imp	SA Cost	Imp	xPRMP Cost	Imp	CGA-TS-LNS Cost	Imp
1	278.73	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00
2	334.96	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00
3	380.35	380.35	0.00	380.35	0.00	380.35	0.00	385.29	-1.30	380.35	0.00
4	430.88	430.88	0.00	430.88	0.00	430.89	0.00	430.89	0.00	430.88	0.00
5	375.28	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00
6	495.85	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00
7	715.02	715.02	0.00	715.02	0.00	715.02	0.00	715.02	0.00	715.02	0.00
8	665.17	674.19	-1.36	665.17	0.00	665.17	0.00	665.17	0.00	665.17	0.00
9	607.65	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00
10	667.42	684.42	-2.55	667.42	0.00	667.86	-0.07	667.42	0.00	673.95	-0.98
11	664.48	678.93	-2.17	664.48	0.00	666.16	-0.25	666.16	-0.25	668.48	-0.60
12	610.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00
13	2502.65	2504.53	-0.08	2502.65	0.00	2504.53	-0.08	2502.65	0.00	2502.65	0.00
14	1029.34	1032.01	-0.26	1029.34	0.00	1029.34	0.00	1029.34	0.00	1029.34	0.00
15	1001.51	1008.56	-0.70	1001.51	0.00	1001.51	0.00	1001.51	0.00	1001.51	0.00
16	698.61	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00
17	861.79	863.27	-0.17	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00
18	982.44	988.91	-0.66	988.61	-0.63	987.10	-0.47	982.44	0.00	986.78	-0.44
19	711.97	730.16	-2.55	726.51	-2.04	723.93	-1.68	711.97	0.00	725.35	-1.88
20	488.69	492.91	-0.86	489.23	-0.11	488.69	0.00	488.69	0.00	488.17	0.11
21	962.10	978.07	-1.66	964.49	-0.25	968.42	-0.66	962.10	0.00	965.25	-0.33
22	976.70	988.15	-1.17	976.70	0.00	985.16	-0.87	993.50	-1.72	984.73	-0.82
23	984.00	1005.94	-2.23	985.18	-0.12	984.00	0.00	991.99	-0.81	987.88	-0.39
24	1140.13	1160.48	-1.78	1152.35	-1.07	1140.13	0.00	1142.02	-0.17	1139.84	0.03
25	1345.89	1360.72	-1.10	1356.24	-0.77	1345.89	0.00	1348.66	-0.21	1354.70	-0.65
26	1255.16	1267.04	-0.95	1262.43	-0.58	1257.00	-0.15	1255.16	0.00	1256.80	-0.13
27	1266.89	1283.66	-1.32	1285.24	-1.45	1271.10	-0.33	1266.89	0.00	1270.38	-0.28
28	2482.86	2528.64	-1.84	2517.27	-1.39	2491.86	-0.36	2482.86	0.00	2502.18	-0.78
29	2128.53	2184.59	-2.63	2151.68	-1.09	2129.10	-0.03	2128.53	0.00	2130.64	-0.10
30	1740.87	1780.54	-2.28	1755.89	-0.86	1740.87	0.00	1744.11	-0.19	1747.45	-0.38
31	2154.33	2232.71	-3.64	2171.60	-0.80	2162.88	-0.40	2154.33	0.00	2242.80	-4.11
32	2165.96	2221.66	-2.57	2191.58	-1.18	2165.96	0.00	2169.66	-0.17	2188.29	-1.03
33	2157.23	2205.34	-2.23	2175.85	-0.86	2157.23	0.00	2161.03	-0.18	2167.13	-0.46
34	1120.44	1150.81	-2.71	1140.83	-1.82	1121.67	-0.11	1120.44	0.00	1134.40	-1.25
35	1310.33	1350.91	-3.10	1340.41	-2.30	1310.33	0.00	1312.88	-0.19	1319.46	-0.70
36	1623.54	1702.33	-4.85	1679.27	-3.43	1625.42	-0.12	1623.54	0.00	1678.95	-3.41
Avg	1092.16	1110.74	-1.70	1100.86	-0.80	1093.90	-0.16	1093.53	-0.13	1100.04	-0.72

BKS: Best-known solution from the literature. Bold entries correspond to higher quality solutions.

Imp: Percentage improvement between the cost and BKS ($Imp = 100 \cdot (BKS - cost) / BKS$).

Table A3. Comparative results for the 2|UR|L instances of Class 3.

Inst	BKS	ACO Cost	Imp	MS-BR Cost	Imp	SA Cost	Imp	xPRMP Cost	Imp	CGA-TS-LNS Cost	Imp
1	284.10	284.23	-0.05	284.10*	0.00	284.11	0.00	284.10	0.00	284.10	0.00
2	352.16	352.16	0.00	352.16	0.00	352.16	0.00	352.16	0.00	352.16	0.00
3	385.32	390.55	-1.36	385.32	0.00	385.32	0.00	385.32	0.00	385.32	0.00
4	430.88	430.88	0.00	430.88	0.00	430.89	0.00	430.89	0.00	430.88	0.00
5	379.94	379.94	0.00	379.94	0.00	379.94	0.00	379.94	0.00	379.94	0.00
6	498.16	498.16	0.00	498.16	0.00	498.16	0.00	498.16	0.00	498.16	0.00
7	664.96	678.75	-2.07	664.96	0.00	664.96	0.00	664.96	0.00	664.96	0.00
8	738.43	738.43	0.00	738.43	0.00	738.43	0.00	738.43	0.00	738.43	0.00
9	607.65	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00
10	591.16	615.68	-4.15	615.36	-4.09	591.61	-0.08	591.16	0.00	615.36	-4.09
11	699.35	706.94	-1.09	699.35	0.00	699.35	0.00	699.35	0.00	699.35	0.00
12	610.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00
13	2377.39	2450.19	-3.06	2377.39	0.00	2377.39	0.00	2377.39	0.00	2377.39	0.00
14	988.79	996.11	-0.74	988.79	0.00	988.80	0.00	988.80	0.00	988.79	0.00
15	1116.07	1145.04	-2.60	1120.75	-0.42	1116.07	0.00	1116.07	0.00	1116.07	0.00
16	698.61	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00
17	861.79	862.62	-0.10	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00
18	986.30	1025.35	-3.96	986.30	0.00	986.30	0.00	1009.62	-2.36	986.30	0.00
19	749.43	753.66	-0.56	752.06	-0.35	749.43	0.00	751.56	-0.28	751.18	-0.23
20	511.46	517.61	-1.20	511.46	0.00	511.46	0.00	511.46	0.00	511.46	0.00
21	1086.72	1114.16	-2.53	1089.75	-0.28	1086.72	0.00	1087.79	-0.10	1085.69	0.09
22	1024.11	1046.71	-2.21	1031.79	-0.75	1024.11	0.00	1028.33	-0.41	1023.50	0.06
23	1041.60	1068.63	-2.60	1056.56	-1.44	1041.60	0.00	1044.06	-0.24	1045.13	-0.34
24	1064.38	1082.30	-1.68	1073.01	-0.81	1066.15	-0.17	1064.38	0.00	1064.43	0.00
25	1325.24	1355.61	-2.29	1353.90	-2.16	1333.64	-0.63	1325.24	0.00	1333.14	-0.60
26	1311.11	1344.32	-2.53	1335.80	-1.88	1311.11	0.00	1312.94	-0.14	1314.93	-0.29
27	1329.33	1376.34	-3.54	1354.76	-1.91	1329.33	0.00	1332.15	-0.21	1326.73	0.20
28	2541.02	2604.08	-2.48	2587.25	-1.82	2541.02	0.00	2544.39	-0.13	2545.98	-0.20
29	2040.83	2090.56	-2.44	2067.69	-1.32	2040.83	0.00	2051.52	-0.52	2048.04	-0.35
30	1767.72	1811.22	-2.46	1812.72	-2.55	1767.72	0.00	1772.71	-0.28	1775.00	-0.41
31	2196.26	2276.01	-3.63	2246.54	-2.29	2196.26	0.00	2196.40	-0.01	2196.32	0.00
32	2166.18	2247.06	-3.73	2219.26	-2.45	2166.18	0.00	2179.06	-0.59	2177.63	-0.53
33	2276.31	2355.08	-3.46	2325.36	-2.15	2276.31	0.00	2284.46	-0.36	2311.18	-1.53
34	1163.05	1204.31	-3.55	1176.71	-1.17	1165.57	-0.22	1163.05	0.00	1172.77	-0.84
35	1393.90	1439.13	-3.24	1437.30	-3.11	1393.90	0.00	1397.77	-0.28	1420.79	-1.93
36	1706.70	1791.54	-4.97	1739.36	-1.91	1708.05	-0.08	1706.70	0.00	1724.60	-1.05
Avg	1110.18	1137.49	-2.46	1124.20	-1.26	1110.58	-0.04	1112.45	-0.21	1114.55	-0.39

BKS: Best-known solution from the literature. Bold entries correspond to higher quality solutions.

Imp: Percentage improvement between the cost and BKS ($Imp = 100 \cdot (BKS - cost) / BKS$).

*Lower scores for this instance was mistakenly reported in Dominguez et al. [23] and Zachariadis et al. [27].

Table A4. Comparative results for the 2|UR|L instances of Class 4.

Inst	BKS	ACO Cost	Imp	MS-BR Cost	Imp	SA Cost	Imp	xPRMP Cost	Imp	CGA-TS-LNS Cost	Imp
1	282.95	282.95	0.00	282.95	0.00	282.95	0.00	282.95	0.00	282.95	0.00
2	334.96	342.00	-2.10	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00
3	358.40	362.41	-1.12	358.40	0.00	362.41	-1.12	362.41	-1.12	362.41	-1.12
4	447.37	447.37	0.00	447.37	0.00	447.37	0.00	447.37	0.00	447.37	0.00
5	383.87	383.88	0.00	383.87	0.00	383.88	0.00	383.88	0.00	383.87	0.00
6	498.32	498.32	0.00	498.32	0.00	498.32	0.00	498.32	0.00	498.32	0.00
7	686.26	702.45	-2.36	686.26	0.00	686.26	0.00	686.26	0.00	686.26	0.00
8	688.32	692.47	-0.60	688.32	0.00	688.32	0.00	688.32	0.00	688.32	0.00
9	625.10	625.13	0.00	625.10	0.00	625.10	0.00	625.10	0.00	625.10	0.00
10	703.64	703.64	0.00	703.64	0.00	703.64	0.00	703.64	0.00	703.64	0.00
11	771.93	782.31	-1.34	773.58	-0.21	771.93	0.00	773.58	-0.21	773.58	-0.21
12	610.23	614.24	-0.66	614.23	-0.66	610.23	0.00	610.23	0.00	614.23	-0.66
13	2500.85	2583.27	-3.30	2533.79	-1.32	2533.79	-1.32	2500.85	0.00	2533.79	-1.32
14	955.09	981.90	-2.81	981.00	-2.71	955.09	0.00	968.21	-1.37	968.39	-1.39
15	1164.63	1216.14	-4.42	1164.77	-0.01	1164.63	0.00	1164.63	0.00	1164.63	0.00
16	703.35	703.35	0.00	703.35	0.00	703.35	0.00	703.35	0.00	703.35	0.00
17	861.79	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00
18	1100.52	1110.48	-0.91	1100.66	-0.01	1100.52	0.00	1100.52	0.00	1100.52	0.00
19	747.03	772.05	-3.35	765.51	-2.47	747.03	0.00	755.04	-1.07	754.98	-1.06
20	533.77	546.91	-2.46	534.14	-0.07	533.77	0.00	535.03	-0.24	535.07	-0.24
21	958.58	976.48	-1.87	967.85	-0.97	959.82	-0.13	958.58	0.00	961.78	-0.33
22	1041.80	1057.15	-1.47	1052.60	-1.04	1041.80	0.00	1042.01	-0.02	1040.65	0.11
23	1047.32	1079.63	-3.09	1064.76	-1.67	1047.32	0.00	1048.43	-0.11	1047.32	0.00
24	1086.09	1103.28	-1.58	1099.40	-1.23	1086.09	0.00	1086.09	0.00	1091.76	-0.52
25	1366.28	1408.64	-3.10	1402.08	-2.62	1366.28	0.00	1374.79	-0.62	1367.80	-0.11
26	1362.22	1414.28	-3.82	1391.02	-2.11	1362.22	0.00	1378.21	-1.17	1379.29	-1.25
27	1284.94	1318.93	-2.65	1318.45	-2.61	1284.94	0.00	1289.02	-0.32	1289.21	-0.33
28	2510.29	2638.07	-5.09	2647.15	-5.45	2510.29	0.00	2514.87	-0.18	2527.78	-0.70
29	2199.79	2267.37	-3.07	2274.09	-3.38	2199.79	0.00	2209.58	-0.45	2201.05	-0.06
30	1784.14	1834.68	-2.83	1851.15	-3.76	1784.14	0.00	1795.94	-0.66	1792.12	-0.45
31	2314.76	2385.63	-3.06	2387.72	-3.15	2314.76	0.00	2326.63	-0.51	2322.37	-0.33
32	2206.72	2268.67	-2.81	2267.57	-2.76	2206.72	0.00	2212.94	-0.28	2219.94	-0.60
33	2318.77	2393.01	-3.20	2387.22	-2.95	2318.77	0.00	2326.73	-0.34	2322.01	-0.14
34	1163.96	1208.19	-3.80	1210.66	-4.01	1163.96	0.00	1171.04	-0.61	1165.76	-0.15
35	1452.59	1503.42	-3.50	1519.28	-4.59	1452.59	0.00	1460.85	-0.57	1471.74	-1.32
36	1604.55	1683.25	-4.90	1670.84	-4.13	1605.00	-0.03	1604.55	0.00	1611.53	-0.44
Avg	1129.48	1159.83	-2.69	1154.27	-2.20	1130.55	-0.10	1132.96	-0.31	1134.	-0.43

BKS: Best-known solution from the literature. Bold entries correspond to higher quality solutions.
Imp: Percentage improvement between the cost and BKS ($Imp = 100 \cdot (BKS - cost) / BKS$).

Table A5. Comparative results for the 2|UR|L instances of Class 5.

Inst	BKS	ACO Cost	Imp	MS-BR Cost	Imp	SA Cost	Imp	xPRMP Cost	Imp	CGA-TS-LNS Cost	Imp
1	278.73	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00	278.73	0.00
2	334.96	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00	334.96	0.00
3	358.40	358.40	0.00	358.40	0.00	358.40	0.00	358.40	0.00	358.40	0.00
4	430.88	430.88	0.00	430.88	0.00	430.89	0.00	430.89	0.00	430.88	0.00
5	375.28	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00	375.28	0.00
6	495.85	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00	495.85	0.00
7	657.77	657.77	0.00	657.77	0.00	657.77	0.00	657.77	0.00	657.77	0.00
8	609.90	609.90	0.00	609.90	0.00	609.90	0.00	609.90	0.00	609.90	0.00
9	607.65	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00	607.65	0.00
10	678.62	680.26	-0.24	684.17	-0.82	678.62	0.00	678.66	-0.01	678.62	0.00
11	624.82	624.82	0.00	624.82	0.00	624.82	0.00	624.82	0.00	624.82	0.00
12	610.00	610.23	-0.04	610.00	0.00	610.00	0.00	610.00	0.00	610.00	0.00
13	2334.59	2334.78	-0.01	2334.78	-0.01	2334.59	0.00	2334.59	0.00	2334.59	0.00
14	871.22	889.20	-2.06	875.07	-0.44	871.22	0.00	871.22	0.00	871.22	0.00
15	1159.94	1160.20	-0.02	1160.96	-0.09	1159.94	0.00	1160.20	-0.02	1159.94	0.00
16	698.61	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00	698.61	0.00
17	861.79	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00	861.79	0.00
18	917.94	924.04	-0.66	921.29	-0.36	917.94	0.00	917.94	0.00	917.93	0.00
19	644.59	651.97	-1.14	644.59	0.00	644.59	0.00	644.59	0.00	644.59	0.00
20	466.79	477.32	-2.26	472.77	-1.28	466.79	0.00	468.60	-0.39	466.79	0.00
21	870.82	888.26	-2.00	886.04	-1.75	870.82	0.00	870.82	0.00	873.25	-0.28
22	928.02	942.06	-1.51	945.92	-1.93	928.02	0.00	930.83	-0.30	931.63	-0.39
23	922.34	942.80	-2.22	938.25	-1.72	922.34	0.00	926.68	-0.47	927.42	-0.55
24	1042.37	1048.33	-0.57	1046.84	-0.43	1042.37	0.00	1042.37	0.00	1042.41	0.00
25	1149.66	1170.38	-1.80	1168.87	-1.67	1149.66	0.00	1150.04	-0.03	1154.53	-0.42
26	1209.34	1231.72	-1.85	1220.83	-0.95	1209.34	0.00	1213.03	-0.31	1216.09	-0.56
27	1231.52	1260.11	-2.32	1258.12	-2.16	1231.52	0.00	1237.05	-0.45	1237.96	-0.52
28	2276.71	2336.45	-2.62	2322.37	-2.01	2276.71	0.00	2287.98	-0.50	2321.42	-1.96
29	2115.53	2158.78	-2.04	2152.26	-1.74	2115.53	0.00	2125.35	-0.46	2146.14	-1.45
30	1512.71	1542.14	-1.95	1548.29	-2.35	1512.71	0.00	1517.86	-0.34	1520.51	-0.52
31	1968.89	2016.59	-2.42	2011.88	-2.18	1968.89	0.00	1980.17	-0.57	1976.45	-0.38
32	1938.96	1983.34	-2.29	1992.03	-2.74	1938.96	0.00	1949.14	-0.53	1955.43	-0.85
33	1946.51	2002.72	-2.89	2001.26	-2.81	1946.51	0.00	1968.32	-1.12	1959.31	-0.66
34	1006.38	1036.16	-2.96	1040.78	-3.42	1006.38	0.00	1014.20	-0.78	1030.70	-2.42
35	1224.21	1256.34	-2.62	1271.21	-3.84	1224.21	0.00	1227.13	-0.24	1239.41	-1.24
36	1457.05	1505.54	-3.33	1522.73	-4.51	1457.05	0.00	1462.33	-0.36	1472.74	-1.08
Avg	1022.76	1038.45	-1.53	1037.94	-1.48	1022.76	0.00	1025.66	-0.28	1028.44	-0.56

BKS: Best-known solution from the literature. Bold entries correspond to higher quality solutions.

Imp: Percentage improvement between the cost and BKS ($Imp = 100 \cdot (BKS - cost) / BKS$).

Table A6. Sign test pairwise comparison for the instances of Class 1

CGA-TS-LNS	PRMP	VNS	BR-LNS	SA
a) Wins (+)	6	6	6	6
b) Loses (-)	0	0	0	0
c) Equal to BKS	30	30	30	30
a + c	36	36	36	36
Detected differences	$\alpha = 0.1$	$\alpha = 0.1$	$\alpha = 0.1$	$\alpha = 0.1$

Table A7. Sign test pairwise comparison for the 2|UR|L instances of Class 2

CGA-TS-LNS	ACO	MS-BR	SA	XPRMP
a) Wins (+)	24	15	9	6
b) Loses (−)	2	11	27	30
c) Equal to BKS	10	10	0	0
a + c	34	25	9	6
Detected differences	$\alpha = 0.1$	$\alpha = 0.1$	–	–

Table A8. Sign test pairwise comparison for the 2|UR|L instances of Class 3

CGA-TS-LNS	ACO	MS-BR	SA	XPRMP
a) Wins (+)	27	19	8	9
b) Loses (−)	1	9	28	27
c) Equal to BKS	8	8	0	0
a + c	35	27	8	9
Detected differences	$\alpha = 0.1$	$\alpha = 0.1$	–	–

Table A9. Sign test pairwise comparison for the 2|UR|L instances of Class 4

CGA-TS-LNS	ACO	MS-BR	SA	XPRMP
a) Wins (+)	29	20	2	10
b) Loses (−)	0	6	34	26
c) Equal to BKS	7	10	0	0
a + c	36	30	2	10
Detected differences	$\alpha = 0.1$	$\alpha = 0.1$	–	$\alpha = 0.05$

Table A10. Sign test pairwise comparison for the 2|UR|L instances of Class 5

CGA-TS-LNS	ACO	MS-BR	SA	XPRMP
a) Wins (+)	24	22	2	7
b) Loses (−)	0	2	34	29
c) Equal to BKS	12	12	0	0
a + c	36	34	2	7
Detected differences	$\alpha = 0.1$	$\alpha = 0.1$	–	–

References

1. Laporte, G. Fifty Years of Vehicle Routing. *Transportation Science* **2009**, *43*, 408–416. doi:10.1287/trsc.1090.0301.
2. Irnich, S.; Toth, P.; Vigo, D. Chapter 1: The Family of Vehicle Routing Problems. In *Vehicle Routing*; Society for Industrial and Applied Mathematics, 2014; pp. 1–33. doi:10.1137/1.9781611973594.ch1.
3. Dantzig, G.; Fulkerson, R.; Johnson, S. Solution of a Large-Scale Traveling-Salesman Problem. *Journal of the Operations Research Society of America* **1954**, *2*, 393–410. doi:10.1287/opre.2.4.393.
4. Baiocchi, M.; Milani, A.; Santucci, V.; Bartocchini, U. An experimental comparison of algebraic differential evolution using different generating sets. *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2019. doi:10.1145/3319619.3326854.
5. Barbu, T. Automatic Unsupervised Texture Recognition Framework Using Anisotropic Diffusion-Based Multi-Scale Analysis and Weight-Connected Graph Clustering. *Symmetry* **2021**, *13*, 925. doi:10.3390/sym13060925.
6. Kang, H.Y.; Lee, A. An Enhanced Approach for the Multiple Vehicle Routing Problem with Heterogeneous Vehicles and a Soft Time Window. *Symmetry* **2018**, *10*, 650. doi:10.3390/sym10110650.
7. Kucharska. Dynamic Vehicle Routing Problem—Predictive and Unexpected Customer Availability. *Symmetry* **2019**, *11*, 546. doi:10.3390/sym11040546.
8. Seo, M.; Lee, S.; Lee, S. Clustering-based Data Dissemination Protocol Using the Path Similarity for Autonomous Vehicles. *Symmetry* **2019**, *11*, 260. doi:10.3390/sym11020260.

9. Kim, J. Vehicle Detection Using Deep Learning Technique in Tunnel Road Environments. *Symmetry* **2020**, *12*, 2012. doi:10.3390/sym12122012.
10. Xu, Y.; Tang, W.; Chen, B.; Qiu, L.; Yang, R. A Model Predictive Control with Preview-Follower Theory Algorithm for Trajectory Tracking Control in Autonomous Vehicles. *Symmetry* **2021**, *13*, 381. doi:10.3390/sym13030381.
11. Dantzig, G.B.; Ramser, J.H. The Truck Dispatching Problem. *Management Science* **1959**, *6*, 80–91. doi:10.1287/mnsc.6.1.80.
12. Vidal, T.; Crainic, T.G.; Gendreau, M.; Prins, C. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research* **2013**, *40*, 475–489. doi:10.1016/j.cor.2012.07.018.
13. Lodi, A.; Martello, S.; Vigo, D. Recent advances on two-dimensional bin packing problems. *Discrete Applied Mathematics* **2002**, *123*, 379–396. doi:10.1016/s0166-218x(01)00347-x.
14. Iori, M.; Salazar-González, J.J.; Vigo, D. An Exact Approach for the Vehicle Routing Problem with Two-Dimensional Loading Constraints. *Transportation Science* **2007**, *41*, 253–264. doi:10.1287/trsc.1060.0165.
15. Gendreau, M.; Iori, M.; Laporte, G.; Martello, S. A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks* **2007**, *51*, 4–18. doi:10.1002/net.20192.
16. Fuellerer, G.; Doerner, K.F.; Hartl, R.F.; Iori, M. Ant colony optimization for the two-dimensional loading vehicle routing problem. *Computers & Operations Research* **2009**, *36*, 655–673. doi:10.1016/j.cor.2007.10.021.
17. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. A Guided Tabu Search for the Vehicle Routing Problem with two-dimensional loading constraints. *European Journal of Operational Research* **2009**, *195*, 729–743. doi:10.1016/j.ejor.2007.05.058.
18. Strodl, J.; Doerner, K.F.; Tricoire, F.; Hartl, R.F. On Index Structures in Hybrid Metaheuristics for Routing Problems with Hard Feasibility Checks: An Application to the 2-Dimensional Loading Vehicle Routing Problem. In *Hybrid Metaheuristics*; Springer Berlin Heidelberg, 2010; pp. 160–173. doi:10.1007/978-3-642-16054-7_12.
19. Leung, S.C.H.; Zheng, J.; Zhang, D.; Zhou, X. Simulated annealing for the vehicle routing problem with two-dimensional loading constraints. *Flexible Services and Manufacturing Journal* **2010**, *22*, 61–82. doi:10.1007/s10696-010-9061-4.
20. Leung, S.C.; Zhou, X.; Zhang, D.; Zheng, J. Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem. *Computers & Operations Research* **2011**, *38*, 205–215. doi:10.1016/j.cor.2010.04.013.
21. Duhamel, C.; Lacomme, P.; Quilliot, A.; Toussaint, H. A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research* **2011**, *38*, 617–640. doi:10.1016/j.cor.2010.08.017.
22. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. Integrated distribution and loading planning via a compact metaheuristic algorithm. *European Journal of Operational Research* **2013**, *228*, 56–71. doi:10.1016/j.ejor.2013.01.040.
23. Dominguez, O.; Juan, A.A.; Faulin, J. A biased-randomized algorithm for the two-dimensional vehicle routing problem with and without item rotations. *International Transactions in Operational Research* **2014**, *21*, 375–398. doi:10.1111/itor.12070.
24. Wei, L.; Zhang, Z.; Zhang, D.; Lim, A. A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research* **2015**, *243*, 798–814. doi:10.1016/j.ejor.2014.12.048.
25. Wei, L.; Zhang, Z.; Zhang, D.; Leung, S.C. A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research* **2018**, *265*, 843–859. doi:10.1016/j.ejor.2017.08.035.
26. Wei, L.; Wang, Y.; Cheng, H.; Huang, J. An open space based heuristic for the 2D strip packing problem with unloading constraints. *Applied Mathematical Modelling* **2019**, *70*, 67–81. doi:10.1016/j.apm.2019.01.022.
27. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. The Vehicle Routing Problem with Simultaneous Pick-ups and Deliveries and Two-Dimensional Loading Constraints. *European Journal of Operational Research* **2016**, *251*, 369–386. doi:10.1016/j.ejor.2015.11.018.
28. Pinto, T.; Alves, C.; de Carvalho, J.V. Variable neighborhood search algorithms for the vehicle routing problem with two-dimensional loading constraints and mixed linehauls and backhauls. *International Transactions in Operational Research* **2018**, *27*, 549–572. doi:10.1111/itor.12509.
29. Gendreau, M.; Iori, M.; Laporte, G.; Martello, S. A Tabu Search Algorithm for a Routing and Container Loading Problem. *Transportation Science* **2006**, *40*, 342–350. doi:10.1287/trsc.1050.0145.
30. Tarantilis, C.; Zachariadis, E.; Kiranoudis, C. A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-Dimensional Container-Loading Problem. *IEEE Transactions on Intelligent Transportation Systems* **2009**, *10*, 255–271. doi:10.1109/tits.2009.2020187.
31. Fuellerer, G.; Doerner, K.F.; Hartl, R.F.; Iori, M. Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *European Journal of Operational Research* **2010**, *201*, 751–759. doi:10.1016/j.ejor.2009.03.046.
32. Lacomme, P.; Toussaint, H.; Duhamel, C. A GRASPES for the vehicle routing problem with basic three-dimensional loading constraints. *Engineering Applications of Artificial Intelligence* **2013**, *26*, 1795–1810. doi:10.1016/j.engappai.2013.03.012.
33. Bortfeldt, A. A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. *Computers & Operations Research* **2012**, *39*, 2248–2257. doi:10.1016/j.cor.2011.11.008.
34. Koch, H.; Bortfeldt, A.; Wäscher, G. A hybrid algorithm for the vehicle routing problem with backhauls, time windows and three-dimensional loading constraints. *OR Spectrum* **2018**, *40*, 1029–1075. doi:10.1007/s00291-018-0506-6.
35. Bortfeldt, A.; Yi, J. The Split Delivery Vehicle Routing Problem with three-dimensional loading constraints. *European Journal of Operational Research* **2020**, *282*, 545–558. doi:10.1016/j.ejor.2019.09.024.
36. Iori, M.; Martello, S. Routing problems with loading constraints. *TOP* **2010**, *18*, 4–27. doi:10.1007/s11750-010-0144-x.
37. Pollaris, H.; Braekers, K.; Caris, A.; Janssens, G.K.; Limbourg, S. Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum* **2014**, *37*, 297–330. doi:10.1007/s00291-014-0386-3.

38. Golden, B.L.; Levy, L.; Vohra, R. The orienteering problem. *Naval Research Logistics* **1987**, *34*, 307–318. doi:10.1002/1520-6750(198706)34:3<307::aid-nav3220340302>3.0.co;2-d.
39. Santucci, V.; Baioletti, M. A Memetic Approach for the Orienteering Problem. In *Communications in Computer and Information Science*; Springer International Publishing, 2020; pp. 38–48. doi:10.1007/978-3-030-45016-8_5.
40. Moghdani, R.; Salimifard, K.; Demir, E.; Benyettou, A. The green vehicle routing problem: A systematic literature review. *Journal of Cleaner Production* **2021**, *279*, 123691. doi:10.1016/j.jclepro.2020.123691.
41. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1992.
42. Glover, F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* **1986**, *13*, 533–549. doi:10.1016/0305-0548(86)90048-1.
43. Shaw, P. Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In *Principles and Practice of Constraint Programming — CP98*; Springer Berlin Heidelberg, 1998; pp. 417–431. doi:10.1007/3-540-49481-2_30.
44. Glover, F.; Taillard, E.; Taillard, E. A users guide to tabu search. *Annals of Operations Research* **1993**, *41*, 1–28. doi:10.1007/bf02078647.
45. Fiechter, C.N. A parallel tabu search algorithm for large traveling salesman problems. *Discrete Applied Mathematics* **1994**, *51*, 243–267. doi:10.1016/0166-218x(92)00033-i.
46. Lorena, L.A.N.; Furtado, J.C. Constructive Genetic Algorithm for Clustering Problems. *Evolutionary Computation* **2001**, *9*, 309–327. doi:10.1162/106365601750406019.
47. Ursani, Z.; Essam, D.; Cornforth, D.; Stocker, R. Localized genetic algorithm for vehicle routing problem with time windows. *Applied Soft Computing* **2011**, *11*, 5375–5390. doi:10.1016/j.asoc.2011.05.021.
48. Solomon, M.M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research* **1987**, *35*, 254–265. doi:10.1287/opre.35.2.254.
49. Nagano, M.S.; Ruiz, R.; Lorena, L.A.N. A Constructive Genetic Algorithm for permutation flowshop scheduling. *Computers & Industrial Engineering* **2008**, *55*, 195–207. doi:10.1016/j.cie.2007.11.018.
50. Gomes, S.P.; Lorena, L.A.N.; Ribeiro, G.M. A Constructive Genetic Algorithm for Discrete Dispersion on Point Feature Cartographic Label Placement Problems. *Geographical Analysis* **2015**, *48*, 43–58. doi:10.1111/gean.12082.
51. True. *Journal of the Operational Research Society* **2001**, *52*, 928–936. doi:10.1057/palgrave/jors/2601163.
52. Tao, Y.; Wang, F. An effective tabu search approach with improved loading algorithms for the 3L-CVRP. *Computers & Operations Research* **2015**, *55*, 127–140. doi:10.1016/j.cor.2013.10.017.
53. Reil, S.; Bortfeldt, A.; Mönch, L. Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints. *European Journal of Operational Research* **2018**, *266*, 877–894. doi:10.1016/j.ejor.2017.10.029.
54. Nara, E.O.B.; Sordi, D.C.; Schaefer, J.L.; Schreiber, J.N.C.; Baierle, I.C.; Sellitto, M.A.; Furtado, J.C. Prioritization of OHS key performance indicators that affecting business competitiveness – A demonstration based on MAUT and Neural Networks. *Safety Science* **2019**, *118*, 826–834. doi:10.1016/j.ssci.2019.06.017.
55. Glover, F.; Melián, B. Tabu Search. *Inteligencia Artificial* **2003**, *7*. doi:10.4114/ia.v7i19.714.
56. Toffolo, T.A.; Vidal, T.; Wauters, T. Heuristics for vehicle routing problems: Sequence or set optimization? *Computers & Operations Research* **2019**, *105*, 118–131. doi:10.1016/j.cor.2018.12.023.
57. Erdoğan, G. An open source Spreadsheet Solver for Vehicle Routing Problems. *Computers & Operations Research* **2017**, *84*, 62–72. doi:10.1016/j.cor.2017.02.022.
58. Johnson, D., Near-optimal bin packing algorithms; Massachusetts Institute of Technology. Dept. of Mathematics, 1973.
59. Dominguez, O.; Guimarans, D.; Juan, A.A.; de la Nuez, I. A Biased-Randomised Large Neighbourhood Search for the two-dimensional Vehicle Routing Problem with Backhauls. *European Journal of Operational Research* **2016**, *255*, 442–462. doi:10.1016/j.ejor.2016.05.002.
60. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* **2011**, *1*, 3–18. doi:10.1016/j.swevo.2011.02.002.
61. Oliveira, R.R.; Cardoso, I.M.; Barbosa, J.L.; da Costa, C.A.; Prado, M.P. An intelligent model for logistics management based on geofencing algorithms and RFID technology. *Expert Systems with Applications* **2015**, *42*, 6082–6097. doi:10.1016/j.eswa.2015.04.001.
62. Gomes, J.Z.; Barbosa, J.L.V.; Geyer, C.F.R.; dos Anjos, J.C.S.; Canto, J.V.; Pessin, G. Ubiquitous Intelligent Services for Vehicular Users: A Systematic Mapping. *Interacting with Computers* **2019**, *31*, 465–479. doi:10.1093/iwcomp/iwz030.
63. Barbosa, J.; Tavares, J.; Cardoso, I.; Alves, B.; Martini, B. TrailCare: An indoor and outdoor Context-aware system to assist wheelchair users. *International Journal of Human-Computer Studies* **2018**, *116*, 1–14. doi:10.1016/j.ijhcs.2018.04.001.
64. Aranda, J.A.S.; Bavaresco, R.S.; de Carvalho, J.V.; Yamin, A.C.; Tavares, M.C.; Barbosa, J.L.V. A computational model for adaptive recording of vital signs through context histories. *Journal of Ambient Intelligence and Humanized Computing* **2021**. doi:10.1007/s12652-021-03126-8.
65. Rosa, J.H.; Barbosa, J.L.V.; Kich, M.; Brito, L. A Multi-Temporal Context-aware System for Competences Management. *International Journal of Artificial Intelligence in Education* **2015**, *25*, 455–492. doi:10.1007/s40593-015-0047-y.
66. Machado, S.D.; da Rosa Tavares, J.E.; Martins, M.G.; Barbosa, J.L.V.; González, G.V.; Leithardt, V.R.Q. Ambient Intelligence Based on IoT for Assisting People with Alzheimer's Disease Through Context Histories. *Electronics* **2021**, *10*, 1260. doi:10.3390/electronics10111260.

-
67. Dupont, D.; Barbosa, J.L.V.; Alves, B.M. CHSPAM: a multi-domain model for sequential pattern discovery and monitoring in contexts histories. *Pattern Analysis and Applications* **2019**, *23*, 725–734. doi:10.1007/s10044-019-00829-9.
 68. da Rosa, J.H.; Barbosa, J.L.; Ribeiro, G.D. ORACON: An adaptive model for context prediction. *Expert Systems with Applications* **2016**, *45*, 56–70. doi:10.1016/j.eswa.2015.09.016.
 69. Filippetto, A.S.; Lima, R.; Barbosa, J.L.V. A risk prediction model for software project management based on similarity analysis of context histories. *Information and Software Technology* **2021**, *131*, 106497. doi:10.1016/j.infsof.2020.106497.
 70. Lucca, A.V.; Sborz, G.M.; Leithardt, V.; Beko, M.; Zeferino, C.A.; Parreira, W. A Review of Techniques for Implementing Elliptic Curve Point Multiplication on Hardware. *Journal of Sensor and Actuator Networks* **2020**, *10*, 3. doi:10.3390/jsan10010003.
 71. Martins, J.A.; Ochoa, I.S.; Silva, L.A.; Mendes, A.S.; González, G.V.; Santana, J.D.P.; Leithardt, V.R.Q. PRIPRO: A Comparison of Classification Algorithms for Managing Receiving Notifications in Smart Environments. *Applied Sciences* **2020**, *10*, 502. doi:10.3390/app10020502.
 72. Leithardt, V.; Santos, D.; Silva, L.; Viel, F.; Zeferino, C.; Silva, J. A Solution for Dynamic Management of User Profiles in IoT Environments. *IEEE Latin America Transactions* **2020**, *18*, 1193–1199. doi:10.1109/tla.2020.9099759.