



Metaheuristic Approaches for the Fleet Size and Mix Vehicle Routing Problem with Time Windows and Step Cost Functions

João L. V. Manguino^(✉)  and Débora P. Ronconi 

Department of Production Engineering, University of São Paulo,
Av. Luciano Gualberto, 1380, Cidade Universitária, São Paulo, SP 05508-010, Brazil
jmanguino@gmail.com

Abstract. The vehicle routing problem is a traditional combinatorial problem with practical relevance for a wide range of industries. In the literature, several attributes have been tackled by dedicated methods in order to better reflect real-life situations. This article addresses the fleet size and mix vehicle routing problem with time windows in which companies hire a third-party logistics company. The shipping charges considered in this work are calculated using step cost functions, in which values are determined according to the type of vehicle and the total distance traveled, with fixed values for predefined distance ranges. The problem is solved with three different metaheuristic methods: Variable Neighborhood Search (VNS), Greedy Randomized Adaptive Search Procedure (GRASP) and a hybrid proposition that combines both. The methods are examined through a computational comparative analysis in 168 benchmark instances from the literature, small-sized instances with known optimal solution, and 3 instances based on a real problem from the civil construction industry. The numerical experiments show that the proposed methods are efficient and show strong performance in different scenarios.

Keywords: Vehicle Routing Problem · Step cost functions · Metaheuristics · Local search · Third party logistics

1 Introduction

The vehicle routing problem (VRP) has been extensively studied in the literature due to its relevance to industry and broad applications. In the last decades, several variants of the problem were proposed to explore the diversity of operating rules and constraints encountered in real-life applications. To address the actual

Supported by FAPESP (grants 2016/01860-1 and 2013/07375-0) and CNPq (grant 306083/2016-7).

© Springer Nature Switzerland AG 2020
E. Lalla-Ruiz et al. (Eds.): ICCL 2020, LNCS 12433, pp. 231–245, 2020.
https://doi.org/10.1007/978-3-030-59747-4_15

needs of carriers and industry, the problem has been enriched with additional restrictions, as well as clients' and fleet's characteristics [12, 25].

This article addresses the situation in which companies hire a third-party logistics company (3PL), whose freight charges are calculated using discontinuous step cost functions, with fixed costs for each distance range according to the type of vehicle being used. The problem also takes in account a diverse fleet to choose from for the routing and time windows in every client and in the depot. This problem is the Fleet Size and Mix Vehicle Routing Problem with Time Windows and Step Cost functions and it is referred with the acronym FSMVRPTWSC, according to the usual nomenclature.

In this problem, the company prepares its own routing, evaluating all the restrictions and costs, but it does not own the fleet of vehicles, it belongs to a 3PL. In general, such providers have a vast fleet with multiple vehicles of each kind to offer to the clients. The use of 3PL allows the company to focus on its core business and to avoid costs related to the acquisition and maintenance of a fleet, equipment depreciation, drivers and employees' payroll, among other costs. The shipping charges, considered in this problem, are calculated using step cost functions in which values are determined according to the type of vehicle and the distance traveled, with fixed values for predefined distance ranges. This form of freight table is common in certain segments of the industries, as clients and providers can calculate and verify the charges quickly and conveniently, avoiding costs of personnel or software in the calculation of each invoice exchanged.

The first proposition of the VRP was made by [4] and ever since has been widely studied and more characteristics were considered. A fleet with multiple vehicle types was introduced to the problem by [8] to generate the FSMVRP (Fleet Size and Mix Vehicle Routing Problem). Time windows were introduced by [22], the first approach to the VRPTW, the same paper generated 56 instances of 100 clients that became an important benchmark for evaluating methods. The FSMVRPTW (Fleet Size and Mix Vehicle Routing Problem) was introduced by [14], who also added different vehicle costs and capacities to the problems generated by [22], combining to 168 reference problems.

The use of third party logistics providers (3PL) has already been studied in the literature in many occasions with different approaches. The first approach was made by [2], in which the decision is to choose, for each route, if the company uses its own fleet, or a 3PL's fleet. In a market research, [13] point out that, by 2013, the 3PL industry had evolved into an important outsourcing option for logistics managers around the globe, generating nearly \$700 billions in annual operating revenues. An extensive review on the relationship between companies and 3PL partners is presented in [17]. Finally, the combination of FSMVRPTW with the step function freight costs, generating the FSMVRPTWSC, was proposed by [16].

Solution quality is lost when solving the FSMVRPTWSC by artificially creating a linear freight cost curve based on the freight table of the problem and applying a linear cost optimization method. As exemplified in [16], the decision to add more clients to a route may cause a zero cost addition as long as the

distance after adding such client still fits in the same distance range and the updated demand can be carried by the same kind of vehicle as before. On the other hand, adding a client that increases the route distance just a small amount can cause a spike in transportation cost, as the small distance change might be just enough to change the distance range. That is why a method that addresses the step cost functions can provide better results.

The use of metaheuristic has been frequent in the multi attribute VRP problems. [12] points out multiple methods of this kind applied to the VRP problem during the 50 years since it started being studied. [24] provide a very extensive review of the evolution of metaheuristics and local search for MAVRP (Multi Attribute VRP). In a survey with the intent to classify and review the taxonomy of the multiple variations of the vehicle routing problems, [3] classify 277 articles of journals with strong impact factor from 2009 and 2015. Among these papers, over 70% of them apply metaheuristic methods.

In Manguino and Ronconi [16], a MILP formulation and two constructive heuristics with local search methods are proposed. This work will further explore the problem through the proposition of metaheuristics approaches that can generate solutions taking advantage of the problem’s characteristics. The metaheuristics are GRASP, VNS and a hybrid method combining both. A computational study is carried out with 168 reference instances, three real case inspired instances and small sized instances for comparison with optimum results.

The next section (Sect. 2) describes details of the problem. Sect. 3 explains the proposed metaheuristics. The result of computational experiments are presented and analysed at Sect. 4, followed by the conclusion in Sect. 5.

2 Problem Description

In the problem, there are $n + 1$ points geographically scattered, $N = \{0, 1, 2, \dots, n\}$. Each route begins and ends at the central depot ($i = 0$), respecting its working hours limited by $[e_0, \ell_0]$. Each client i ($i = 1, 2, \dots, n$) has a predetermined demand q_i , a service time s_i , and the start time of the service should be within a specific time window, i.e. between time instants e_i and ℓ_i . The distance d_{ij} and travel time t_{ij} between every pair of points are known before the routing plan is defined. There are K different types of vehicles available. Each type of vehicle k ($k = 1, 2, \dots, K$) has a load capacity a_k ($a_1 < a_2 < \dots < a_K$).

The cost of each vehicle has a fixed value for each predefined distance range, i.e. each vehicle k has a cost $C_{k,f}$ whether its total traveled distance varies from W_f to W_{f+1} for $f = 0, 1, 2, \dots, F - 1$, where $F - 1$ is the penultimate distance range. The last range, F , is an exception, since it has no upper bound and the cost grows linearly, starting from $C_{k,F-1}$, plus C_{kF} for each unit of distance added. Thus, given a traveled distance $d > 0$, the step cost function C_k for vehicle k can be defined as

$$C_k(d) = \begin{cases} C_{k,f}, & \text{if } W_f < d \leq W_{f+1} \text{ for } f \in \{0, 1, \dots, F - 1\}, \\ C_{k,F-1} + C_{kF}(d - W_F), & \text{if } W_F < d. \end{cases}$$

To illustrate the cost calculation, Fig. 1 presents an illustrative example.

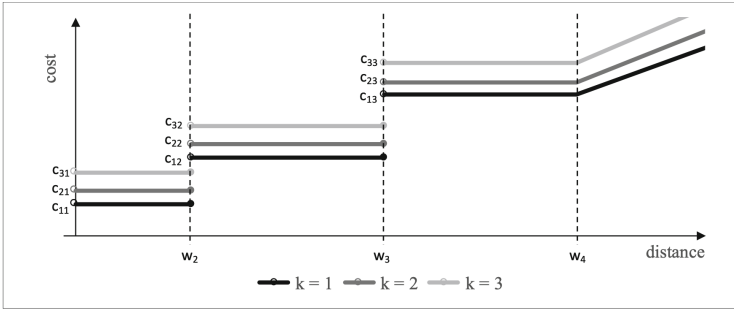


Fig. 1. A illustrative figure from [16] with the step cost function chart representation for case with 3 vehicle types ($k = 1, 2$ and 3) and 4 distance ranges ($f = 1, 2, 3$ and 4). Every distance range starts at w_f and each cost is fixed at $c_{k,f}$. For the last distance range ($F = 4$), cost starts in $c_{k,F-1}$ and adds $c_{k,F}$ at each extra distance unit.

3 Proposed Metaheuristics

In this work, the FSMVRPTWSC is approached using metaheuristics. In [16] two constructive sequential insertion heuristics are proposed, as well as two local search procedures. This work utilizes the knowledge of the problem and the most successful constructive method (SCIH2), as well as the local search movements, and proposes additional local search movements and the development of metaheuristics.

The choice of the methods to approach was based on [1]. In this paper, it is discussed that there is not one definitive method that achieves best results, but rather methods that can explore and extract the best of each problem’s characteristics. With such inspiration, GRASP and VNS were chosen. While the latter uses local searches alternated with increasing amounts of shakes to achieve best results, the first adds controlled variances to a constructive method in a multi-start procedure. Also, taking advantage of possible synergy among methods, a hybrid combination of both is explored.

The main components of the proposed methods were calibrated in 54 instances extracted from the complete set of instances described in Sect. 4.1. This set combines all considered characteristics, with three instances of each combination. More details can be found in [15].

3.1 Local Searches

As the local search is a crucial element for the proposed methods, this section describes their structures and applied movements. Two local searches are proposed by [16], Cross and Relocate. In this work the two methods are reviewed and the 2-opt* is explored.

Relocate consists of removing a client from a route and re-inserting it back to the solution in a different position, either in the same route or a different one. The procedure follows along the proposed by [11], but with one criteria for removing and inserting clients to routes. Figure 2 is an illustration to aid the understanding of the movement.

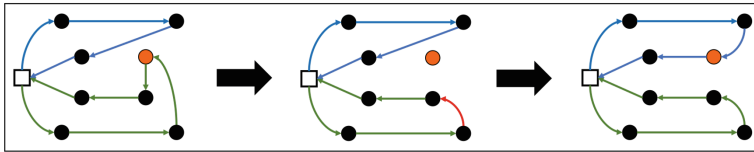


Fig. 2. Illustration of the Relocate local search movement.

The client whose removal generates the biggest cost reduction is chosen. To be re-inserted, the position that causes the minimal cost increase, as long as it is not the same position it was removed from, is chosen. In each iteration up to 60% of the clients with the greatest remove criteria values will be evaluated for the relocation to find the largest total cost reduction.

Cross is an exchange of sections between two different routes of the problem, as it is illustrated in Fig. 3. It was first proposed by [23]. The movement is performed by selecting two distinct routes and for each one a starting position and number of clients of the section is chosen. In this work, the neighborhood is composed by the exchanges of every pair of routes, with sections starting in every position of each route and with section sizes from 1 to 5 clients. The cross exchange that generates the biggest cost reduction is selected.

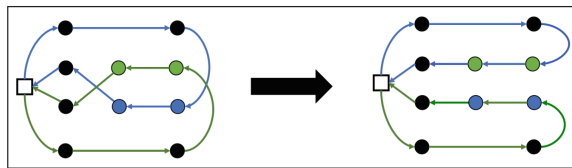


Fig. 3. Illustration of the Cross local search movement.

2-opt* exchanges final sections of routes, as it is illustrated in Figure 4. It was first proposed by [20]. The movement is performed by selecting two distinct routes and for each of them the position from which the section will be exchanged. In this work, the neighborhood is composed by the exchanges of every pair of routes, with sections starting in every position of each route. The 2-opt* exchange that generates the biggest cost reduction is selected.

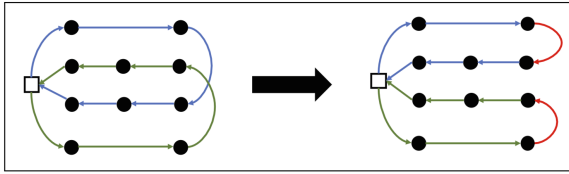


Fig. 4. Illustration of the 2-opt* local search movement.

3.2 GRASP

The Greedy Randomized Adaptive Search Procedure [5] is a multi start method with the execution of a randomized constructive method followed by a local search. The randomized decisions are made with the use of Restricted Candidate Lists (RCLs), that are very well discussed by [21], which are lists with the best candidates for the next decision. It is not a random move, it has criteria for selecting which candidates will be in the list and selecting a candidate from it.

Some further detail of the GRASP was provided by [6]. It lays out some concepts from which the Algorithm 1 was drawn.

Algorithm 1. GRASP

```

1: Input: InstanceData, MaxTime
2: Output: BestSol
3: for CPUTime < MaxTime do
4:   SolTemp ← RandomizedSCIH2()
5:   SolTemp ← LocalSearch(SolTemp)
6:   UpdateSol(SolTemp, BestSol)
7: end for
8: return BestSol

```

At line 4, the *RandomizedSCIH2()* refers to an adaptation of the SCIH2 insertion constructive method proposed by [16]. To illustrate the adaptation, Fig. 5 presents a side-by-side comparison in a flowchart format of the original SCIH2 and the *RandomizedSCIH2*. Note that it is the same procedure, with the main differences highlighted in the key decision moments of the sequential insertion, when starting a new route, and when deciding which client to add to the route under construction.

SCIH2 is a sequential insertion constructive method, based on the procedure originally proposed for the VRPTW (Vehicle Routing Problem with Time Windows) by [22] and adapted for other multi attributes routing problems throughout the literature. It interactively creates new routes using the furthest unrouted client available and starts adding clients to this route. Clients that can be added to the route, if their demand is not greater than what would be the available capacity if the route was served by the biggest vehicle available are evaluated for insertion through criteria C1 and C2. C1 is the impact caused by the insertion

of a client to the route by a cost increase, reducing the vehicle’s capacity, using distance of the distance range and time available for the route. $C1$ is calculated for each position of the route and the minimum value is considered, always with care to never violate the time windows in every client of the route. $C2$ is the benefit of avoiding an exclusive route to the candidate client after considering $C1$. A positive $C2$ means that it is advantageous to insert the client to the route, a negative value means this insertion should be disregarded. The client with the greatest $C2$ should be inserted and a new list of candidates formed for another consideration of $C1$ and $C2$. If there is no client with positive $C2$, that route is done and a new one should be generated. This process is repeated until no clients are left unrouted.

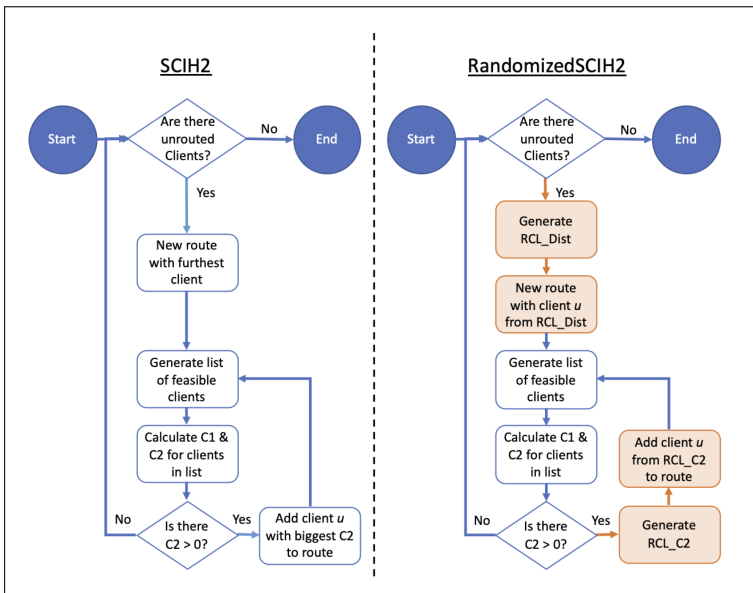


Fig. 5. Side-by-side comparison of the flowcharts for SCIH2 and RandomizedSCIH2 with highlights for the changes among each procedure.

For the RandomizedSCIH2 procedure, instead of starting new routes using the furthest client, a RCL of most distance is formed and the new route is started with a client chosen from this list. Also, instead of choosing the client with the greatest $C2$, a RCL of clients with positive $C2$ is generated, from which a client is chosen to be inserted.

The RCLs of the RandomizedSCIH2 are formed and clients are chosen with the following rules:

- *RCL_Dist*:
 - Fixed size list with up to 5 candidates;
 - The 5 furthest clients from the deposit are inserted;
 - Client is chosen randomly from the RCL.
- *RLC_C2*:
 - Clients are accepted based on their C2 value, there is no maximum number of clients to be accepted;
 - From the biggest positive C2 value, every client with C2 greater than 10% less is included;
 - Clients are chosen randomly, with bias to their C2 values.

After the generation of each new solution, it is enhanced using a local search with the Cross movement.

3.3 VNS

VNS (Variable Neighborhood Search) was proposed by [18] and has been widely utilized in the literature. In a survey, [10] explain that the method is based on facts statements: (i) A local minimum with respect to one neighborhood structure is not necessary so for another; (ii) A global minimum is a local minimum with respect to all possible neighborhood structures; and (iii) For many problems local minima with respect to one or several neighborhoods are relatively close to each other. Fact (iii) is empirical and implies that a local optimum can provide some information of the global optimum and, therefore, a careful search of the neighborhood is important.

The method starts with an initial solution obtained from a constructive method that can provide strong results. From that solution local searches that can best generate and explore neighborhoods using the problem's characteristics are performed. After every local optimum is found, the solution is moved to a different neighborhood using a shake procedure. As iterations happen and no new local optimums are found, the shake procedure has to be stronger, to explore further neighborhoods. The Algorithm 2 presents the general outline of the VNS proposed.

As explained in Sect. 3.1, three different local search procedures are used in this work. According to [9], the search in the different neighborhoods should start from the smallest to the largest. That is a form of intensifying the search near to the local minimum since, as the three stated facts suggest, local minimums tend to have similar characteristics.

In a neighborhood analysis, 2-opt* has the smallest neighborhood, as it combines two routes at a time and all positions in each of them. Cross is next, as it adds to the neighborhoods of 2-opt* up to 5 different section sizes. Relocate is the biggest as it combines up to 60% of the clients with every position available in the current solution. Therefore, the sequence of neighborhoods is: 2-opt*, Cross, and Relocate.

During the shake step, each local search performs a random move. For 2-opt*, two routes are selected randomly as well as the positions for exchange in each

Algorithm 2. VNS

```

1: Input:  $InitSol$ ,  $MaxNeighborhoods$ ,  $MaxTime$ 
2: Output:  $BestSol$ 
3:  $BestSol \leftarrow InitSol$ 
4: repeat
5:    $TempSol \leftarrow BestSol$ 
6:    $Neighborhood \leftarrow 1$ 
7:   while  $Neighborhood \leq MaxNeighborhoods$  do
8:      $SolTemp \leftarrow Shake(SolTemp)$ 
9:      $SolTemp \leftarrow LocalSearch(SolTemp, Neighborhood)$ 
10:    if  $Cost(SolTemp) < Cost(BestSol)$  then
11:       $BestSol \leftarrow SolTemp$ 
12:    else
13:       $Neighborhood \leftarrow Neighborhood + 1$ 
14:    end if
15:  end while
16: until  $CPUTime > MaxTime$ 
17: return  $BestSol$ 

```

of them. For Cross, after the same selections as 2-opt*, also a random section size is selected. Finally for Relocate, a client is randomly chosen for removal and inserted in a route and position chosen randomly. Infeasible solutions are never accepted, so if the shake generates an one, it is discarded and the shake is repeated until a feasible solution is achieved.

If after going through all the local searches no improvement in the current solution is achieved, the shake is intensified. That is done only after 10 iterations without improvement. The shake starts with one random move for each local search and is added one extra movement at each intensification and can scale up to 50% of the solution routes for 2-opt*, 1% of the solution routes for Cross and 40% of the number of clients in the problem for Relocate.

The proposed VNS version will have the following sequence:

- constructive heuristic SCIH2 from [16] as seed;
- Local Searches:
 - 2-opt*;
 - Cross;
 - Relocate.
- For each Local Search a specific shake is performed;
- For each full cycle of performing all the local searches without any improvement, the shake is intensified, starting from a minimal shake;
- Repeat process for a predetermined CPU Time.

In each and every step of the process, if a new best solution is found, it is always accepted. Infeasible solutions are never considered.

3.4 Hybrid

As mentioned in the previous sections, both methods have the same background idea that local optimal solutions have similar characteristics and search could be performed nearby. On the other hand, each one uses different strategies in different steps of the solution, so a combination of both does not present conflict, but it is rather complimentary.

Hybrid metaheuristic methods are vastly used in the literature to take advantage of strong search procedures of different methods. [7] explore multiple forms of applying GRASP as the chosen strategy, but also combined in hybrid methods, including the VNS. [7] claim that GRASP can generate a proper seed to start the search procedure of the VNS, as the present paper attempts. The combination of GRASP as a seed generator for a VNS procedure is also experimented by [19] for the orienteering problem.

For the FSMVRPTWSC, a hybrid procedure based on the GRASP and VNS metaheuristic is proposed as described in Algorithm 3.

Algorithm 3. GRASP+VNS Hybrid Metaheuristic

```

1: Input: InstanceData, MaxTime, GRASPTimeShare
2: Output: BestSol
3: BestSol  $\leftarrow$  SCIH2
4: while  $CPUTime \leq (GRASPTimeShare \times MaxTime)$  do
5:   RandomizedSCIH2(TempSol);
6:   if  $Cost(SolTemp) < Cost(BestSol)$  then
7:     BestSol  $\leftarrow$  SolTemp
8:   end if
9: end while
10: while  $CPUTime \leq MaxTime$  do
11:   Neighborhood  $\leftarrow$  1
12:   while  $Neighborhood \leq MaxNeighborhoods$  do
13:     SolTemp  $\leftarrow$  Shake(SolTemp)
14:     SolTemp  $\leftarrow$  LocalSearch(SolTemp, Neighborhood)
15:     if  $Cost(SolTemp) < Cost(BestSol)$  then
16:       BestSol  $\leftarrow$  SolTemp
17:     else
18:       Neighborhood  $\leftarrow$   $Neighborhood + 1$ 
19:     end if
20:   end while
21: end while
22: return BestSol;

```

The procedure starts in line 3, where the solution from the greedy original constructive heuristic is used as an initial solution. From that point on, a share of the solution time (*GRASPTimeShare*) is dedicated for executing the *RandomizedSCIH2* procedure and capturing the best solution from this phase. After the time share, the remainder of the execution time is dedicated to the execution of the VNS search.

GRASPTimeShare was calibrated by running the calibration for 12 min with values ranging in: 1%, 3%, 5%, 10%, 15%, 20%, ... , 50%. After analysis and calibration, the time share for the GRASP procedure (*GRASPTimeShare*) was determined in 5%. Even though this apparently runs the VNS share for much longer than the GRASP part, the GRASP process is much faster to run, as it is a multi start of a constructive method that in average lasts less than 1×10^{-4} s, therefore a great number of solutions is generated in the same time as a local search can generate new solutions for a much longer time.

4 Numerical Experiments

Computational experiments were conducted to evaluate the proposed methods. The codes of the proposed procedures were written in C programming language and tests were conducted on a 2.1 GHz Intel Core i7-3612QM with 8 GB of RAM memory. Each instance was executed for 12 min.

The proposed methods were applied in the instances presented in [16], which include a set of 168 instances with 100 clients adapted from the literature, three instances inspired in a real case and 72 small sized instances.

4.1 Literature Adapted Instances

The 168 instances adapted from the literature were adapted from the 56 instances generated by [22] for the VRPTW that were further adapted by [14] for the FSMVRPTW. The instances are grouped with the following characteristics, picking one option for each aspect:

- **Physical distribution of clientes:**
R: Randomly spread; *C*: Clustered; *RC*: Part randomly spread, part clustered
- **Time Windows Length:**
1: Tight; *2*: Large
- **Vehicle Costs:**
a: Most expensive; *b*: Medium; *c*: Least expensive.

Table 1 presents and compares the results obtained by the three proposed methods for the reference instances. For comparison, the original SCIH2 method followed by local search with the Cross movement was implemented (SCIH2+LS). Values are compared through the average objective function value for each group of instances as well as how many of the best known results are obtained.

It is noticeable that the three proposed methods outperform the SCIH2+LS in every group of instances. Among the proposed methods, VNS and GRASP have similar performances, with a smaller total average cost by VNS, but more best known results percentage by GRASP, with a very strong dominance in the RC2 instance group. There is a strong performance by the Hybrid method, with a smaller average total cost smaller then all the other methods. When the

share of best known solutions is analysed, the dominance of the Hybrid method becomes clear, as it provides 55.4% of the best known results.

Aiming to determine if there is a relevant difference between the results obtained by the proposed methods, statistical tests were applied. First, the Kolmogorov-Smirnov test was applied to accept or reject the normality of the results distribution, which was rejected. Based on that, the Wilcoxon signed-rank test was applied for every pair of method's results. With a significance level of

Table 1. Average Cost and percentage of best known solutions for each group of the instances. Values in bold are the best known average for each group and biggest percentage of known best results. The percentage of the methods may sum to over 100%, as more than one method may obtain the same best known value.

Data set	#Inst	SCIH2+LS		GRASP		VNS		Hybrid	
		Cost	%Best	Cost	%Best	Cost	%Best	Cost	%Best
C1a	9	6,168.8	0.0	6,151.4	11.1	6,145.4	55.6	6,140.9	66.7
C1b	9	1,597.5	0.0	1,571.3	100.0	1,584.0	33.3	1,579.1	33.3
C1c	9	1,025.7	0.0	995.6	77.8	1,011.8	11.1	1,002.1	44.4
C1	27	2,930.7	0.0	2,906.1	63.0	2,913.7	33.3	2,907.4	48.1
C2a	8	5,292.0	0.0	5,277.8	0.0	5,243.3	50.0	5,240.0	62.5
C2b	8	1,292.0	0.0	1,287.6	0.0	1,240.1	62.5	1,242.9	75.0
C2c	8	802.6	0.0	775.9	25.0	743.3	62.5	746.4	50.0
C2	24	2,462.2	0.0	2,447.1	8.3	2,408.9	58.3	2,409.8	62.5
R1a	12	2,805.7	0.0	2,756.0	8.3	2,749.0	25.0	2,738.5	66.7
R1b	12	799.8	0.0	763.6	25.0	765.5	0.0	755.2	75.0
R1c	12	539.8	0.0	503.5	8.3	500.8	33.3	495.8	58.3
R1	36	1,381.7	0.0	1,341.0	13.9	1,338.5	19.4	1,329.8	66.7
R2a	11	2,709.8	9.1	2,689.4	0.0	2,640.2	72.7	2,654.5	36.4
R2b	11	887.9	9.1	862.5	18.2	828.0	90.9	834.5	72.7
R2c	11	648.5	0.0	588.9	63.6	624.6	9.1	591.5	36.4
R2	33	1,415.4	6.1	1,380.3	27.3	1,364.3	57.6	1,360.2	48.5
RC1a	8	3,313.8	0.0	3,247.9	87.5	3,290.9	0.0	3,254.3	25.0
RC1b	8	1,036.7	0.0	1,006.0	12.5	997.4	25.0	994.7	62.5
RC1c	8	683.5	0.0	650.7	0.0	635.4	50.0	633.3	50.0
RC1	24	1,678.0	0.0	1,634.9	33.3	1,641.2	25.0	1,627.4	45.8
RC2a	8	3,119.9	0.0	3,057.9	62.5	3,086.5	25.0	3,055.3	50.0
RC2b	8	871.5	0.0	828.5	87.5	864.5	0.0	828.4	50.0
RC2c	8	534.6	12.5	514.0	100.0	534.6	12.5	517.5	62.5
RC2	24	1,508.7	4.2	1,466.8	83.3	1,495.2	12.5	1,467.0	54.2
Total	168	1,852.1	1.8	1,818.2	36.3	1,815.3	34.5	1,805.7	54.8

1%: SCIH2+LS is different from all the other methods; VNS and GRASP cannot be differentiated and the Hybrid method is different.

4.2 Real Case Inspired Instances

The instances based on a real case presented in [16] reflect the operation of three days of a distribution center of a major construction material supplier in São Paulo, Brazil.

Table 2. Cost and quantity of routes generated by each metaheuristic versus the reported values. Best results are highlighted in bold.

Day	Reported		GRASP		VNS		Hybrid	
	Cost	#Routes	Cost	#Routes	Cost	#Routes	Cost	#Routes
1	27,676.6	58	26,425.2	64	24,433.5	41	24,683.9	44
2	35,151.3	65	28,650.0	61	27,500.0	50	27,800.0	53
3	32,379.9	59	27,781.2	64	25,604.5	43	25,419.8	40
Total	95,207.8	182	82,856.4	189	77,538.0	134	77,903.7	137

The results in Table 2 show that VNS presented the best results, with a very similar, but slightly lower cost when compared to the Hybrid method. GRASP, dominated by the other methods, has better results than the Reported values, despite generating more routes.

4.3 Small-Sized Instances

To enable the comparison of proposed methods to known optimal results, [16] generated 72 small-sizes instances based on the literature adapted instances. For each combination of characteristics there are four quantities of clients: 10, 15, 20 and 25, summing up to a total of 72 problems. These were solved in a commercial solver and 42 optimal solutions were achieved.

Table 3. Results obtained in the small-sized instances when comparing the results from the solver. “Optimal solutions” compare how many optimal solutions were found by each method; “Gap to optimal” evaluate the average percentage cost difference from the optimal and the method’s objective function cost; “Gap to LB” compare the results found with the lower bound provided by the solver.

	Solver	GRASP	VNS	Hybrid
Optimal solutions	42	28	42	36
Gap to Optimal (%)	–	3.3	0.0	1.2
Gap to LB (%)	17.3	19.7	16.7	17.7

Table 3 presents a comparison of the results obtained by the three proposed methods and the solver. VNS found all of the known optimal results and, among the problems without a known optimal, it obtained a result better than the best feasible solution found by the solver. The Hybrid method failed to find 6 of the optimal values, but the gap to optimal is only 1.2% and a similar gap to the lower bound as the solver. GRASP, on the other hand, is less successful when solving these problems when compared to the other methods. Even so, it finds more than half of the optimal values, with a gap of 3.3% to the optimal values.

5 Conclusion and Further Research

This work presented new methods for the FSMVRPTWSC, a rich vehicle routing problem that consider fixed costs per distance ranges and vehicle type. The investigation of this problem, that reflects a form of freight charge often present in the industry, shows opportunities for savings by using the length of the distance ranges to include clients to routes without additional costs, as well as avoiding that small distance increase cause a jump in the route cost.

Three metaheuristics, GRASP, VNS and Hybrid, were proposed with the purpose of generating methods that can best explore the particularities of the problem, that has a novelty aspect with a very particular characteristic of having the objective function determined by fixed values for distance ranges for each vehicle type. The methods combine the experience accumulated in the literature about VNS and GRASP, with the knowledge of the opportunities of minimizing the costs by exploiting the specific characteristics of the problem.

Numerical experiments confirm that the chosen methods have strong performances, outperforming other methodologies. The Hybrid method has a strong performance in benchmark instances, as well as in other instances, with also good results by the VNS in other instances. Further researches can be conducted in multiple fronts. New metaheuristics of different strategies, such as genetic methods, can be evaluated. Exact methods can also be further explored and combined with metaheuristics as a seed.

References

1. Arnold, F., Sörensen, K.: What makes a VRP solution good? The generation of problem-specific knowledge for heuristics. *Comput. Oper. Res.* **106**, 280–288 (2019)
2. Ball, M.O., Golden, B., Assad, A., Bodin, L.: Planning for truck fleet size in the presence of a common-carrier option. *Decis. Sci.* **14**(1), 103–120 (1983)
3. Braekers, K., Ramaekers, K., Van Nieuwenhuysse, I.: The vehicle routing problem: state of the art classification and review. *Comput. Ind. Eng.* **99**, 300–313 (2016)
4. Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. *Manag. Sci.* **6**(1), 80–91 (1959)
5. Feo, T.A., Resende, M.G.: A probabilistic heuristic for a computationally difficult set covering problem. *Oper. Res. Lett.* **8**(2), 67–71 (1989)
6. Feo, T.A., Resende, M.G.: Greedy randomized adaptive search procedures. *J. Global Optimiz.* **6**(2), 109–133 (1995)

7. Festa, P., Resende, M.G.C.: Hybrid GRASP heuristics. In: Abraham, A., Hassanien, A.E., Siarry, P., Engelbrecht, A. (eds.) *Foundations of Computational Intelligence Volume 3*. SCI, vol. 203, pp. 75–100. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01085-9_4
8. Golden, B., Assad, A., Levy, L., Gheysens, F.: The fleet size and mix vehicle routing problem. *Comput. Oper. Res.* **11**(1), 49–66 (1984)
9. Hansen, P., Mladenovic, N.: A tutorial on variable neighborhood search. *Groupe d'Études et de Recherche en Analyse des Décisions, HEC Montréal* (2003)
10. Hansen, P., Mladenović, N., Pérez, J.A.M.: Variable neighbourhood search: methods and applications. *Ann. Oper. Res.* **175**(1), 367–407 (2010). <https://doi.org/10.1007/s10479-009-0657-6>
11. Koç, Ç., Bektaş, T., Jabali, O., Laporte, G.: A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows. *Comput. Oper. Res.* **64**, 11–27 (2015)
12. Laporte, G.: Fifty years of vehicle routing. *Transp. Sci.* **43**(4), 408–416 (2009)
13. Lieb, R.C., Lieb, K.J.: The north American third-party logistics industry in 2013: the provider CEO perspective. *Transp. J.* **54**(1), 104–121 (2015)
14. Liu, F.H., Shen, S.Y.: The fleet size and mix vehicle routing problem with time windows. *J. Oper. Res. Soc.* **50**(7), 721–732 (1999)
15. Manguino, J.L.V.: Heuristic methods applied to the mixed fleet vehicle routing problem with time windows and step costs per distance range. Ph.D. Thesis, Escola Politécnica, Universidade de São Paulo, São Paulo (2020). www.teses.usp.br
16. Manguino, J.L.V., Ronconi, D.P.: Step cost functions in a fleet size and mix vehicle routing problem with time windows. *Ann. Opera. Res.* (2020, submitted)
17. Marasco, A.: Third-party logistics: a literature review. *Int. J. Prod. Econ.* **113**(1), 127–147 (2008)
18. Mladenović, N., Hansen, P.: Variable neighborhood search. *Comput. Oper. Res.* **24**(11), 1097–1100 (1997)
19. Palomo-Martínez, P.J., Salazar-Aguilar, M.A., Laporte, G., Langevin, A.: A hybrid variable neighborhood search for the orienteering problem with mandatory visits and exclusionary constraints. *Comput. Oper. Res.* **78**, 408–419 (2017)
20. Potvin, J.Y., Rousseau, J.M.: An exchange heuristic for routeing problems with time windows. *J. Oper. Res. Soc.* **46**(12), 1433–1446 (1995). <https://doi.org/10.1057/jors.1995.204>
21. Resende, M.G., Ribeiro, C.C.: Greedy randomized adaptive search procedures: advances, hybridizations, and applications. In: Gendreau, M., Potvin, J.Y. (eds.) *Handbook of Metaheuristics*. ISOR, vol. 146, pp. 283–319. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-1665-5_10
22. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **35**(2), 254–265 (1987)
23. Taillard, É., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.Y.: A tabu search heuristic for the vehicle routing problem with soft time windows. *Transp. Sci.* **31**(2), 170–186 (1997)
24. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: Heuristics for multi-attribute vehicle routing problems: a survey and synthesis. *Eur. J. Oper. Res.* **231**(1), 1–21 (2013)
25. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: A unified solution framework for multi-attribute vehicle routing problems. *Eur. J. Oper. Res.* **234**(3), 658–673 (2014)