International Conference
on Industrial Engineering and Systems Management

IESM'2011

May 25 - May 27
METZ - FRANCE

A hybrid algorithm for the Vehicle Routing Problem with Time Windows[⋆]

Sabir RIBAS[a], Anand SUBRAMANIAN[a], Igor Machado COELHO[a], Luiz Satoru OCHI[a], Marcone Jamilson Freitas SOUZA[b]

[a]*Instituto de Computação, Universidade Federal Fluminense – Niterói, RJ – Brazil*

[b]*Departamento de Computação, Universidade Federal de Ouro Preto – Ouro Preto, MG – Brazil*

Abstract

The Vehicle Routing Problem with Time Windows is a particular case of the classical Vehicle Routing Problem in which the demands of each customer should be met within an established time window. Due to the combinatorial complexity of the problem its resolution by pure exact methods is, in many cases, computationally impractical. This fact motivates the development of heuristic algorithms, which are usually faster but do not guarantee the best solution for the problem. This work proposes a hybrid algorithm that combines the metaheuristic Iterated Local Search, the Variable Neighborhood Descent procedure and an exact Set Partitioning model. The latter mathematical procedure is periodically activated in order to find the best combination of the routes generated along the execution of the algorithm. The computational results demonstrate that the proposed hybrid approach is quite competitive, since out of the 56 test problems considered, the algorithm was found capable to improve the best known heuristic/hybrid solution in 12 cases and to equal the result of another 27.

Key words: Vehicle Routing Problem, Hybrid Algorithms, Iterated Local Search

1 Introduction

The Vehicle Routing Problem with Time Windows (VRPTW) is a well known optimization problem and it has received a lot of attention in operational research literature. In this problem, a fleet of vehicles must leave the depot, serve customer demands, and return to the depot, at minimum cost, without violating the capacity of the vehicles as well as the time window specified by each customer.

There are two main reasons (operational and theoretical) for investing in research to develop new algorithms for the efficient resolution of this problem. From the practical/operational point of view, the costs related to transporting people or merchandise are generally high, with a tendency to increase, motivated by the actual expansion of commerce of all types [3]. Researchers calculate that 10% to 15% of the final cost of the merchandise commercialized in the world is due to its transport [10]. From the theoretical aspect, since the VRP and most of its variants, including the VRPTW, are NP-hard problems [13], the efficient resolution of these problems represents a challenge for researchers, who, in general, opt for heuristic approaches. The size of this challenge is demonstrated by the great number of articles dealing with this type of problem.

[⋆] This paper was not presented to another journal. Corresponding author Sabir Ribas. Tel. +55 21 2629-5665. Fax +55 21 2629-5666.

Email addresses: `sribas@ic.uff.br` (Sabir RIBAS), `anand@ic.uff.br` (Anand SUBRAMANIAN), `imcoelho@ic.uff.br` (Igor Machado COELHO), `satoru@ic.uff.br` (Luiz Satoru OCHI), `marcone@iceb.ufop.br` (Marcone Jamilson Freitas SOUZA).

The VRPTW has been dealt with various objectives and, in the present work, the aim is to minimize the total traveling distance which is one of the most commonly found in literature. Rochat [20], Backer [4], Riise [19], Kilby [9], Ombuki [17], Alvarenga [3] and Oliveira [7] also adopted the same objective.

Given the complexity of the problem, its resolution using pure exact methods is often an extremely arduous task due the large amount of computational time required. This fact has motivated the development of new heuristic algorithms for solving VRPTW. It is noteworthy to mention that such algorithms aims at finding near-optimal solutions using less computational effort.

The algorithm proposed in this article for solving VRPTW combines the concepts of Iterated Local Search metaheuristic, the Variable Neighborhood Descent method and an exact Set Partitioning model, which periodically determines the best combination of routes generated during the execution of the algorithm. The developed algorithm was tested on a set of 56 instances containing 100 customers that was proposed by Solomon [23]. These test-problems had been widely adopted in the literature and it is used to measure the efficiency of exact, heuristic and hybrid methods.

The remainder of this work is organized as follows. Section 2 defines the VRPTW. Section 3 describes the proposed algorithm. Section 4 contains the computational results. Section 5 presents the concluding remarks.

2 Problem definition

The VRPTW can be defined in a complete directed graph $G = (V, A)$ in which $V = \{0, \ldots, n + 1\}$ is the set of vertices and $A = \{(i, j)|i, j \in V\}$ is the set of arcs. Each arc (i, j) is associated to a time t_{ij} and a traveling cost c_{ij}, both non-negative.

At this moment, it is necessary to precisely define the term *traveling cost*. In practice, the traveling cost can take into consideration many factors, such as: distance, time, vehicle wear and consumption during the trip, among others. However, when dealing with the theoretical problems involving time windows, it is common to convert all the relevant measures into time units for standardization and method comparison purposes. Therefore, in this work the definition of *traveling cost* is: distance converted into time units.

In general, a set K of identical vehicles with capacity Q should attend n customers, represented by vertices $1, \ldots, n$. Consider that $N = V - \{0, n+1\}$ represents a set of customers. To attend these customers, the vehicles should leave the depot, visit them, and then, return to the depot. For the sake of convenience, the depot is represented by two vertices: the vertex 0 representing the source and the vertex $n + 1$ representing the sink. Each customer i, has a demand q_i that must be attended by a single vehicle. In addition, all the vertices have a time window $[e_i, l_i]$; that is, the service of the vertex i must be started within this interval. If the vehicle arrives at the customer i before the instant e_i, it should await the opening of the window. The vehicle cannot arrive at i after the instant l_i, since this would violate the time constraint of the problem. This type of constraint is known in the literature as hard time windows. For each vertex, there is a service time d_i. The objective is to minimize the route's total cost $c(s)$; in other words, to minimize the sum of all the costs of the trip $\sum_{(i,j)\in s} c_{ij}$ that are associated with the arcs (i, j) present in the solution s.

The VRPTW mathematical formulation, also used by Cordeau et al [6], is represented by the expressions (1)-(9) which follow. In these expressions, the binary variable x_{ijk} assumes a value of 1, if the vehicle k traverses arc (i, j); and 0, otherwise.

$$\textit{Minimize} \sum_{k\in K} \sum_{i\in V} \sum_{j\in V} c_{ij} x_{ijk} \tag{1}$$

subject to:

$$\sum_{k\in K}\sum_{j\in V} x_{ijk} = 1 \qquad \forall i \in N \tag{2}$$

$$\sum_{j\in V} x_{0jk} = 1 \qquad \forall k \in K \tag{3}$$

$$\sum_{i\in V} x_{ijk} - \sum_{i\in V} x_{jik} = 0 \qquad \forall k \in K, \forall j \in N \tag{4}$$

$$\sum_{i\in V} x_{i(n+1)k} = 1 \qquad \forall k \in K \tag{5}$$

$$\sum_{i\in N} q_i \sum_{j\in V} x_{ijk} \le Q \qquad \forall k \in K \tag{6}$$

$$b_{ik} + d_i + t_{ij} - (1 - x_{ijk})M_{ij} \le b_{jk} \qquad \forall k \in K, \forall (i,j) \in A \tag{7}$$

$$e_i \le b_{ik} \le l_i \qquad \forall k \in K, \forall i \in V \tag{8}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in K, \forall (i,j) \in A \tag{9}$$

The objective function (1) represents the total cost to be minimized. Constraints (2) assure that only one vehicle k leaves the customer i. Constraints (3)-(5) guarantee the path continuity of the vehicle k; that is, each vehicle leaves the depot, visits the customers, and then, returns to the depot. Constraints (6) make sure that each vehicle k only attends the set of customers whose total demands do not surpass its capacity Q. Constraints (7)-(8) assure route feasibility as regards the time window, where b_{ik} represents the time from which the vehicle k begins to serve the customer i and M_{ij} are large constants. According to Cordeau et al. [6], M_{ij} can be replaced by $\max\{b_i + d_i + t_{ij} - e_j, 0\}$ $\forall (i,j) \in A$. Constraints (9) define the domain of the decision variables.

3 Proposed methodology

This section details the proposed hybrid algorithm. Section 3.1 presents the data structure used to represent a VRPTW solution, while Section 3.2 describes the penalty-based function that evaluates a solution for the problem. Next, Section 3.3 demonstrates the procedure used to construct the initial solution; and Section 3.4 describes the utilized neighborhood structures. Finally, Section 3.5 presents the proposed algorithm.

3.1 Solution representation

A route r is defined by a sequence of integer numbers that corresponds to the identifiers of the customers in r. A solution s contains a set of routes.

3.2 Evaluation function

A solution s is evaluated by the function f, given by the equation (10), which must be minimized:

$$f(s) = \sum_{r\in s} g(r) = \sum_{r\in s} (c(r) + w_l.l(r) + w_e.e(r)) \tag{10}$$

where: g is a function that evaluates routes; $c(r)$ is the cost of the route r; $l(r)$ corresponds to the lateness time for r; $e(r)$ is the load excess in the route r; w_l and w_e are penalties per unit of delay and excess load, respectively, and those values were empirically fixed in $w_l = 200$ and $w_e = 300$. Notice that when s is feasible, the value given by f will only correspond to the travel cost, since in this case: $l(r) = e(r) = 0$.

3.3 Constructive procedure

To obtain an initial solution for the VRPTW, a cheapest insertion method, called *CI-POP*(), that explores the Proximate Optimality Principle [18] was developed. According to this principle, in an optimal sequence of choices, each sub-sequence should also be optimal. It is worth mentioning that although this principle deals with optimal cases, in the developed algorithm there is no guarantee that the optimal solution will be obtained, or even parts of the optimal solution. Thus, this principle is only employed to generate better initial solutions. The pseudo-code of the constructive method is presented in Algorithm 1.

Algorithm 1: CI-POP()

```
Let $s_0$ be a solution with $|K|$ empty routes
foreach customer $c \in \{1, \ldots, n\}$ do
    $best\_cost \leftarrow \infty$
    foreach route $r \in s_0$ do
        foreach position $p$ of route $r$ which it is possible to insert customer $c$ do
            $cost \leftarrow$ cost of the insertion of customer $c$ at position $p$ of route $r$
            if $cost < best\_cost$ then
                $best\_r \leftarrow r$ ; $best\_p \leftarrow p$ ; $best\_cost \leftarrow cost$
            end
        end
    end
    Add customer $c$ at position $best\_p$ in the route $best\_r$ of solution $s_0$
    if $(c \mod \lceil n/5 \rceil) = 0$ then
        Refine the partial solution $s_0$ by the heuristic $h$
    end
end
return $s_0$
```

Let $|K|$ be the maximum number of available vehicles. Initially, the constructive algorithm creates $|K|$ empty routes (line 1 of Algorithm 1) and a list of candidates to be inserted in the set of routes (line 2). The idea of the procedure is to iteratively insert each candidate customer in the best location (line 12). A local search is periodically performed in the partial solution (line 14). More specifically, the parameters of line 13 were calibrated in such a way that five local searches occur during the construction; for example, if there is a total of 100 customers, the local search is performed for every twenty customers added to the partial solution. In this case, the local search is performed using the RVND (see Section 3.5.2). The procedure terminates when all customers have been added.

3.4 Neighborhood structures

In order to explore the solution space, 10 neighborhood structures are used, where six of these modify two routes at each movement performed (inter-route), while the other four modify only a single route (intra-route). The inter-route neighborhood structures are generated by the following movements: *Shift*$(1, 0)$, *Shift*$(2, 0)$, *Shift*$(3, 0)$, *Swap*$(1, 1)$, *Swap*$(2, 1)$ and *Swap*$(2, 2)$. A movement of the neighborhood structure *Shift*$(k, 0)$ involves transferring k adjacent customers from route r_1 to another route r_2; and a movement of the type, *Swap*(k, l), interchanges k adjacent customers from route r_1 to l other adjacent customers from another route r_2.

As for those neighborhood structures that only modify one route at a time, the following movements are used: *Exchange*, *Shift'*(1), *Shift'*(2) and *Shift'*(3). The *Exchange* neighborhood involves the permutation between two customers of the same route and it can be seen as an intra-route version of the *Swap*$(1, 1)$ neighborhood. The other three neighborhoods can be considered as intra-route versions of the *Shift*$(1, 0)$, *Shift*$(2, 0)$ e *Shift*$(3, 0)$ neighborhoods, respectively.

3.5 Proposed algorithm

The proposed algorithm, called Intensified Iterated Local Search (IILS-SP), involves the construction of an initial solution according to the procedure presented in Section 3.3, followed by a local search that combines adapted versions of the Iterated Local Search (ILS) and Variable Neighborhood Descent (VND) methods with an exact approach based on the mathematical formulation of the Set Partitioning (SP). The pseudo-code of IILS-SP is presented in Algorithm 2. Let s_0 be an initial solution; s^* the best solution obtained during the procedure execution; s' a perturbed solution; and, s'' a local optimal solution obtained by the application of the RVND to the perturbed solution. The following sections detail each part of this algorithm.

3.5.1 Intensified Iterated Local Search

Intensified Iterated Local Search is an extension of the *Iterated Local Search* – ILS [14] metaheuristic. ILS explores the solution space by applying perturbations to the current local optimal solution. This metaheuristic starts

Algorithm 2: IILS-SP()

```
1  $s_0 \leftarrow$ *CI-POP*()
2  $s^* \leftarrow$ RVND($s_0$)
3  **repeat**
4      $s' \leftarrow$ Perturbation($s^*$, *history*)
5      $s'' \leftarrow$ RVND($s'$)
6      **if** *AppropriatedMoment(history)* **then**
7          $s'' \leftarrow$ Intensification ($s''$)
8      **end**
9      $s^* \leftarrow$ AcceptanceCriterion($s''$, $s^*$, *history*)
10 **until** *stop criterion not satisfied*
11 **return** $s^*$
```

with the initial solution s_0 and applies a local search to it, obtaining s^*. Next, the method iteratively performs the following steps: (i) perturbs the current best solution s^*; (ii) obtains a solution s'; and (iii) performs a local search in s', obtaining a local optimal s''. If s'' is better than the current best solution s^*, then the method transforms s'' into the new current solution. Otherwise, the method performs another iteration. This procedure is repeated until the stopping criterion is met.

It is important to emphasize that ILS's success strongly depends on the perturbations performed. This way, the perturbation applied to a given solution should be proportioned in such a way that the resulting modification is sufficient to escape from local optima and to explore different regions of the search space, but keeping some characteristics of the current best solution, in order to avoid a complete random restart in next iterations.

In this work, a perturbation (line 4 of Algorithm 2) consists of applying $p + 2$ movements randomly chosen in the neighborhood *Shift*, presented in Section 3.4, where $p \in \{0, 1, 2, \dots\}$ represents the perturbation level. This way, the greater this value, the greater the number of modifications performed in the solution. Herein, *ILSmax* iterations without improvement are applied in the same perturbation level. When this value is achieved, the perturbation level is increased.

In this case, the local search of the IILS (lines 2 and 5 of Algorithm 2) is performed using the Variable Neighborhood Descent with random neighborhood ordering, denoted by RVND (see Section 3.5.2).

Finally, the proposed algorithm contains an intensification module (line 7 of Algorithm 2). This module is activated at appropriate moments of the search and invokes a mathematical programming procedure, based on Set Partitioning, to find the optimal set of routes among those generated during the search. More specifically, the partitioning model is applied to the set formed by all the routes belonging to the solutions generated after the local search phase of the IILS algorithm. That is, for each IILS iteration, the routes of the solution s'' (line 5 of Algorithm 2) are added to the set to be partitioned. This is done in such a way that there are no repeated routes in the set, which has an unlimited size. A description of this module is given in Section 3.5.3.

3.5.2 Variable Neighborhood Descent with random neighborhood exploration

The procedure Variable Neighborhood Descent (VND) [16] involves an exhaustive exploration of the solution space by means of systematic exchanges of the neighborhood structures. During the local search, only the solution that is better than the current best solution is accepted. When a better solution is found, the method restarts the search, beginning with the first neighborhood structure.

The method VND is based on three principles: (i) a local optimum for a given neighborhood structure does not necessarily correspond to a local optimum of another neighborhood structure; (ii) a global optimum corresponds to a local optimum for all neighborhood structures; and (iii) for many problems, the local optimum of a given neighborhood structure is close to the local optima of other neighborhood structures.

The latter principle, of empirical nature, indicates that a local optimum frequently gives some type of information about the global optimal. This is the case in which local and global optimum share a lot of variables with the same value.

The classical version of VND searches local optimal solutions following a fixed order of neighborhood structures. This strategy is widely applied and the results in literature confirm its efficiency. However, for the results presented in this work, a random order was used to explore the neighborhoods. This strategy is adopted with success in [24]. Here, this strategy is so-called RVND.

3.5.3 Set partitioning model

The intensification phase of the proposed algorithm involves the exact resolution of a Set Partitioning Problem (SPP). Let $\mathcal{R}$ be the subset of routes generated by the IILS-algorithm and let y_j, $\forall j \in \mathcal{R}$, be the binary variables that indicate if the route $j \in \mathcal{R}$ is part of the solution ($y_j = 1$); or not ($y_j = 0$). Each route $j \in \mathcal{R}$ has an associated cost g_j. The parameter m_{ij} is equal to 1 if the customer $i \in N$ is attended by the route $j \in \mathcal{R}$; and 0, otherwise. The mathematical formulation is as follows.

$$Minimize \sum_{j \in \mathcal{R}} g_j y_j \tag{11}$$

subject to:

$$\sum_{j \in \mathcal{R}} m_{ij} y_j = 1 \qquad \forall i \in N \tag{12}$$

$$\sum_{j \in \mathcal{R}} y_j \leq |K| \tag{13}$$

$$y_j \in \{0, 1\} \qquad \forall j \in \mathcal{R} \tag{14}$$

The objective of this formulation is to find a set of routes that attend the constraints of the problem with a minimum cost (11). Constraints (12) guarantee that each customer is visited by exactly one route. Constraints (13) ensure that a solution contains up to $|K|$ routes. Constraints (14) define the domain of the variables.

Many combination optimal problems can be described as a SPP. For VRP problems in particular, this model has been adopted with success by various authors, among them: Agarwal [1], Desrosiers [8], Kohl [11] and Larsen [12]. The last two specifically address VRPTW, obtaining new optimal results for the Solomon [23] instances.

In this work, the SPP model was implemented using API Concert for C++ and solved by the CPLEX optimizer, version 12.

4 Computational results

The proposed algorithm (IILS-SP) was developed in C++ programming language and tested in a computer with an Intel Quad Core 2.4 GHz microprocessor with 8 GB of RAM memory and operating system Ubuntu Linux 9.10 64 bits (kernel 2.6.31).

IILS-SP was applied to solve the set of instances proposed by Solomon [23], which is well known and widely used in literature. This set has 56 instances with 100 customers, divided in six classes: C1, C2, R1, R2, RC1 and RC2. The instances contained in the C1 and C2 classes have customers that are geographically clustered, while in R1 and R2, the customers are randomly located. The RC1 and RC2 classes mix the clustering and randomly located customers. In addition, the C2, R2 and RC2 classes are designed for a longer term planning and have a greater vehicle capacity than the C1, R1 and RC1 classes. This means that each vehicle in C2, R2 and RC2 has a capacity to attend a greater number of customers.

For each of the 56 instances, five runs were performed using a 10-minute processing time limit for each run as stopping criterion [1]. The algorithm was empirically calibrated and the parameters were fixed as follows: (*i*) in the construction of an initial solution, as customers are being inserted, five local searches were performed as described in Section 3.3; (*ii*) the number of no-improvement iterations at a given level of perturbation of IILS was fixed as 20; (*iii*) the procedure is iteratively performed according to the *Multi-Start* [15] method, where at each iteration, an initial solution is constructed by a non-deterministic method described in the Section 3.3

[1] The computational results of this research are available at *http://www.decom.ufop.br/sabir/shared/2011iesm-vrptw-results.zip*

and a local search is performed by IILS-SP; and (*iv*) the maximum processing time for each execution of the mathematical solver in the intensification phase was limited to 5 seconds.

Tables 1-6 present to best-available results in literature for each of the analyzed instances, as well as the results obtained by the proposed algorithm. The first column indicates the name of the problems. The next three correspond to the number of vehicles ("$|K|$"), the total distance traveled in the best solution obtained for the problem ("Distance") and the first work that obtained it, respectively. The next group of columns present the results of the proposed method, which include: (*i*) the number of vehicles; (*ii*) the distance of the best solution obtained in 5 executions of the algorithm ("Best"); (*iii*) the average traveled distance obtained ("Average"); (*iv*) the deviation from the average with respect to the best solution ("gap"), which is in accordance with the following definition: $gap = (Average - Best)/Best$, where $Best$ is the best distance value found; and (*v*) the last column ("Impr.") presents the improvements obtained by IILS-SP with respect to the best solutions found in literature, in accordance with $Improvement = (BestLit - Best)/Best$. The best solutions are highlighted in boldface and the solutions improved by the IILS-SP algorithm are underlined.

Table 1
Results for C1 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\|K\|$	Distance	Work	$\|K\|$	Distance	Average	gap	
C101	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C102	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C103	10	**828.06**	RT95 [20]	10	**828.06**	**828.06**	0.00%	0.00%
C104	10	**824.78**	RT95 [20]	10	**824.78**	**824.78**	0.00%	0.00%
C105	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C106	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C107	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C108	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%
C109	10	**828.94**	RT95 [20]	10	**828.94**	**828.94**	0.00%	0.00%

Table 2
Results for C2 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\|K\|$	Distance	Work	$\|K\|$	Distance	Average	gap	
C201	3	**591.56**	RT95 [20]	3	**591.56**	**591.56**	0.00%	0.00%
C202	3	**591.56**	RT95 [20]	3	**591.56**	**591.56**	0.00%	0.00%
C203	3	**591.17**	RT95 [20]	3	**591.17**	**591.17**	0.00%	0.00%
C204	3	**590.60**	RT95 [20]	3	**590.60**	596.42	0.99%	0.00%
C205	3	**588.88**	RT95 [20]	3	**588.88**	**588.88**	0.00%	0.00%
C206	3	**588.49**	RT95 [20]	3	**588.49**	**588.49**	0.00%	0.00%
C207	3	**588.29**	RT95 [20]	3	**588.29**	**588.29**	0.00%	0.00%
C208	3	**588.32**	RT95 [20]	3	**588.32**	**588.32**	0.00%	0.00%

In summary, the best solutions found during the executions by the IILS-SP were: 100% (9/9) tied values for C1; 100% (8/8) tied values for C2; 33.3% (4/12) improved and 41.6% (5/12) tied values for R1; 27.3% (3/11) improved and 9.1% (1/11) tied values for R2; 37.5% (3/8) improved and 37.5% (3/8) tied values RC1; and 25% (2/8) improved and 12.5% (1/8) tied values for RC2. Overall, the values improved in 21.4% (12/56) of the cases, the values tied in 48.2% (27/56) and the values decreased in 30.4% (17/56).

The algorithm proved to be robust, since it presented relatively small gaps. In 80.4% (45/56) of the analyzed instances, gap was less that 1.0%. When this value was improved, the gap was always smaller than 4.16% (as in the R208). These results show that the algorithm produces final solutions with quite little variability in terms of solution quality. In addition, in some cases (R110, R202 e RC105) the proposed algorithm produced better results in average than those found in literature.

Table 7 presents the results of different researches that had as primary objective the minimization of the total distance traveled. The columns represent the algorithm whereas the lines show the average number of vehicles

Table 3
Results for R1 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\lvert K \rvert$	Distance	Work	$\lvert K \rvert$	Distance	Average	gap	
R101	20	**1642.88**	AM04 [2]	20	**1642.88**	**1642.88**	0.00%	0.00%
R102	18	**1472.62**	AM04 [2]	18	1472.81	1472.81	0.01%	-0.01%
R103	14	**1213.62**	RT95 [20]	14	**1213.62**	1214.40	0.06%	0.00%
R104	11	**986.10**	AL07 [3]	11	**982.30**	988.65	0.26%	0.39%
R105	15	**1360.78**	AL07 [3]	15	**1360.78**	**1360.78**	0.00%	0.00%
R106	13	**1241.52**	AL07 [3]	13	**1239.37**	1242.48	0.08%	0.17%
R107	11	**1076.13**	AL07 [3]	11	**1075.21**	1076.23	0.01%	0.09%
R108	10	**948.57**	AL07 [3]	10	951.22	955.39	0.72%	-0.28%
R109	13	**1151.84**	AL07 [3]	13	**1151.84**	1152.06	0.02%	0.00%
R110	11	**1080.36**	RT95 [20]	12	**1072.41**	**1072.41**	0.00%	0.74%
R111	12	**1053.50**	AL07 [3]	12	**1053.50**	1054.43	0.09%	0.00%
R112	10	**953.63**	RT95 [20]	10	956.36	961.66	0.84%	-0.29%

Table 4
Results for R2 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\lvert K \rvert$	Distance	Work	$\lvert K \rvert$	Distance	Average	gap	
R201	8	**1147.80**	OV08 [7]	8	**1147.80**	1149.94	0.19%	0.00%
R202	8	**1039.32**	OV08 [7]	8	**1034.35**	**1039.19**	0.47%	0.48%
R203	6	**874.87**	OV08 [7]	6	881.12	892.38	2.00%	-0.71%
R204	5	**735.80**	OV08 [7]	4	745.12	759.48	3.22%	-1.25%
R205	5	**954.16**	OV08 [7]	5	955.96	967.51	1.40%	-0.19%
R206	5	**884.25**	OV08 [7]	5	**879.89**	891.15	0.78%	0.50%
R207	4	**797.99**	OV08 [7]	4	808.23	820.73	2.85%	-1.27%
R208	4	**705.62**	OV08 [7]	3	724.98	734.94	4.16%	-2.67%
R209	5	**860.11**	OV08 [7]	5	**859.39**	866.29	0.72%	0.08%
R210	5	**910.98**	OV08 [7]	7	914.84	919.89	0.98%	-0.42%
R211	4	**755.82**	OV08 [7]	4	762.38	772.97	2.27%	-0.86%

Table 5
Results for RC1 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\lvert K \rvert$	Distance	Work	$\lvert K \rvert$	Distance	Average	gap	
RC101	15	**1623.58**	RT95 [20]	15	**1623.58**	1626.82	0.20%	0.00%
RC102	14	**1466.84**	AL07 [3]	14	**1461.23**	1472.15	0.36%	0.38%
RC103	11	**1261.67**	S98 [22]	11	1262.02	1273.51	0.94%	-0.03%
RC104	10	**1135.48**	C01 [6]	10	1135.52	1135.79	0.03%	0.00%
RC105	16	**1518.60**	AM04 [2]	16	**1518.58**	**1518.58**	0.00%	0.00%
RC106	13	**1377.35**	AM04 [2]	13	**1376.99**	1378.25	0.07%	0.03%
RC107	12	**1212.83**	AM04 [2]	12	**1212.83**	**1212.83**	0.00%	0.00%
RC108	11	**1117.53**	AM04 [2]	11	**1117.53**	1120.99	0.31%	0.00%

and the total distance traveled of the best solutions obtained for each class. For each algorithm, the average results with respect to Solomon's benchmarks are reported with respect to number of vehicles ("NV") and total distance ("TD"). CNV and CTD indicate, respectively, the cumulative number of vehicles and cumulative total distance over all the 56 instances. When observing the results of each group separately, the conclusion is that the algorithm values tied with those of the best results found in literature in the cluster groups of C1 and C2,

Table 6
Results for RC2 problem class

Problem	Literature best known			IILS-SP				Impr.
	$\|K\|$	Distance	Work	$\|K\|$	Distance	Average	gap	
RC201	9	**1266.11**	OV08 [7]	9	<u>**1265.90**</u>	1268.59	0.20%	0.02%
RC202	8	**1096.75**	OV08 [7]	8	<u>**1095.64**</u>	1099.17	0.22%	0.10%
RC203	5	**926.89**	OV08 [7]	5	937.45	949.94	2.49%	-1.13%
RC204	4	**786.38**	OV08 [7]	4	796.55	810.88	3.12%	-1.28%
RC205	7	**1157.55**	OV08 [7]	7	1157.66	1161.82	0.37%	-0.01%
RC206	6	**1056.21**	OV08 [7]	6	1069.00	1076.84	1.95%	-1.20%
RC207	6	**966.08**	OV08 [7]	6	**966.08**	982.07	1.66%	0.00%
RC208	4	**779.84**	OV08 [7]	5	785.07	788.78	1.15%	-0.67%

Table 7
Comparisons between different works that optimize the total distance traveled

Class		Work					This work
		RT95[20]	CA99[5]	SC00[21]	AL07[3]	OV08[7]	
C1	NV	10.00	10.00	10.00	10.00	10.00	10.00
	TD	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**
C2	NV	3.00	3.00	3.00	3.00	3.00	3.00
	TD	**589.86**	596.63	**589.86**	**589.86**	**589.86**	**589.86**
R1	NV	12.16	12.42	12.08	13.25	13.33	13.17
	TD	1208.50	1233.34	1211.53	1183.38	1186.94	**1181.03**
R2	NV	2.91	3.09	2.82	5.55	5.36	5.36
	TD	961.71	990.99	949.27	899.90	**878.79**	883.10
RC1	NV	11.87	12.00	11.88	12.88	13.25	12.75
	TD	1377.39	1403.74	1361.76	1341.67	1362.44	**1338.54**
RC2	NV	3.37	3.38	3.38	6.50	6.13	6.13
	TD	1119.59	1220.99	1097.63	1015.90	**1004.59**	1009.17
All classes	CNV	414	420	412	489	488	482
	CTD	57231	58927	56830	55134	55021	**54842**

and outperformed them in the groups of R1 and RC1. In the R2 and RC2 groups, although the results were close, they were not able to improve the values of the other groups. Therefore, when considering the overall scenario, IILS-SP outperformed all the others algorithms in terms of solution quality.

5 Conclusions

This paper presented a hybrid algorithm for the Vehicle Routing Problem with Time Windows. The proposed algorithm (IILS-SP) combines the metaheuristic Iterated Local Search, the Variable Neighborhood Descent procedure and an exact Set Partitioning model that, periodically, finds the best combination of the routes generated along the algorithm. Given the combinational nature of the problem, its resolution using pure exact approaches is, in many cases, computationally impractical. This fact motivates the development of heuristic algorithms to address this problem. The proposed algorithm combines the flexibility of heuristic methods and the power of mathematical programming.

The IILS-SP was tested in 56 well-known VRPTW instances and the results were compared with the best solutions found in literature. The computational results show that the proposed hybrid approach is quite competitive, since out of the 56 test problems considered, the algorithm improved the best known heuristic/hybrid solution in 12 cases and equaled the result of another 27.

Acknowledgements

The authors acknowledge CAPES, CNPq and FAPEMIG for supporting the development of this research.

References

[1] Y. Agarwal, K. Mathur, and H. M. Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19:731–740, 1989.

[2] G. B. Alvarenga and G. R. Mateus. A two-phase genetic and set partitioning approach for the vehicle routing problem with time windows. In *HIS '04: Proceedings of the Fourth International Conference on Hybrid Intelligent Systems*, pages 428–433, Washington, DC, USA, 2004. IEEE Computer Society.

[3] G. B. Alvarenga, G. R. Mateus, and G. de Tomi. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers and Operations Research*, 34:1561–1584, 2007.

[4] B. De Backer and V. Furnon. Meta-heuristics in constraint programming experiments with tabu search on the vehicle routing problem. In *MIC'97: Proceedings of the Second International Conference on Metaheuristics*, Sophia Antipolis, France, 1997.

[5] Y. Caseau and F. Laburthe. Heuristics for large constrained vehicle routing problems. *Journal of Heuristics*, 5:281–303, 1999.

[6] J. F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon, and F. Soumis. *The Vehicle Routing Problem*, chapter The VRP with Time Windows, pages 157–193. Paolo Toth and Daniele Vigo, SIAM Monographs on Discrete Mathematics and Applications, 2001.

[7] H. de OLIVEIRA and G. Vasconcelos. A hybrid search method for the vehicle routing problem with time windows. *Annals of Operations Research*, 2008.

[8] J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14(4):545–565, 1984.

[9] P. Kilby, P. Prosser, and P. Shaw. Guided local search for the vehicle routing problem with time windows. In *META-HEURISTICS advances and trends in local search paradigms for optimization*, pages 473–486, Boston: Kluwer Academic, 1999. S. Voss, S. Martello, I. H. Osman, & C. Roucairol (Eds.).

[10] G. F. King and C. F. Mast. Excess travel: causes, extent and consequences. *Transportation Research Record*, (1111):126–134, 1997.

[11] N. Kohl. *Exact methods for Time Constained Routing and Related Scheduling Problems.* PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1995.

[12] J. Larsen. *Parallelization of the vehicle routing problem with time windows.* PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1995.

[13] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 2006.

[14] H. R. Lourenco, O. C. Martin, and T. Stutzle. Iterated local search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, chapter 11. Kluwer Academic Publishers, Boston, 2003.

[15] R. Martí. Multi-start methods. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, chapter 12. Kluwer Academic Publishers, Boston, 2003.

[16] N. Mladenovic and P. Hansen. A variable neighborhood search. *Computers and Operations Research*, 24:1097–1100, 1997.

[17] B. Ombuki, B. J. Ross, and F. Hanshar. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24:17–30, 2006.

[18] M. G. C. Resende and C. C. Ribeiro. Grasp. In E. K. Burke and G. Kendall, editors, *Search Methodologies.* Springer (to appear), 2 edition, 2010. Available at: http://www.ic.uff.br/∼celso/artigos/grasp.pdf.

[19] A. Riise and M. Stølevik. Implementation of guided local search for the vehicle routing problem. Technical report, Department of Computer Science, Michigan State University, SINTEF Applied Mathematics, Norway, 1999.

[20] Y. Rochat and E. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1:147–167, 1995.

[21] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159:139–171, 2000.

[22] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. *Lecture notes in computer science*, pages 417–431, 1998.

[23] M. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window contraints. *Operational Research*, 35:254–265, 1987.

[24] A. Subramanian, L.M.A. Drummond, C. Bentes, L.S. Ochi, and R. Farias. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers and Operations Research*, 37:1899–1911, 2010.