Computers & Operations Research 40 (2013) 2519–2531

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A hybrid algorithm for a class of vehicle routing problems

Anand Subramanian [a,*], Eduardo Uchoa [b], Luiz Satoru Ochi [c]

[a] Universidade Federal da Paraíba, Departamento de Engenharia de Produção, Centro de Tecnologia, Campus I - Bloco G, Cidade Universitária, João Pessoa-PB 58051-970, Brazil
[b] Universidade Federal Fluminense - Departamento de Engenharia de Produção, Rua Passo da Pátria 156, Bloco E - 4° andar, São Domingos, Niterói-RJ 24210-240, Brazil
[c] Universidade Federal Fluminense - Instituto de Computação, Rua Passo da Pátria 156, Bloco E - 3°andar, São Domingos, Niterói-RJ 24210-240, Brazil

ARTICLE INFO

Available online 24 January 2013

Keywords:
Vehicle Routing Problems
Matheuristic
Iterated Local Search
Set Partitioning

ABSTRACT

In this work we propose a hybrid algorithm for a class of Vehicle Routing Problems with homogeneous fleet. A sequence of Set Partitioning (SP) models, with columns corresponding to routes found by a metaheuristic approach, are solved, not necessarily to optimality, using a Mixed Integer Programming (MIP) solver, that may interact with the metaheuristic during its execution. Moreover, we developed a reactive mechanism that dynamically controls the dimension of the SP models when dealing with large size instances. The algorithm was extensively tested on benchmark instances of the following Vechicle Routing Problem (VRP) variants: (i) Capacitated VRP; (ii) Asymmetric VRP; (iii) Open VRP; (iv) VRP with Simultaneous Pickup and Delivery; (v) VRP with Mixed Pickup and Delivery; (vi) Multi-depot VRP; (vii) Multi-depot VRP with Mixed Pickup and Delivery. The results obtained were quite competitive with those found by heuristics devoted to specific variants. A number of new best solutions were obtained.

1. Introduction

The Vehicle Routing Problem (VRP) is a classical Combinatorial Optimization (CO) problem that was proposed in the late 1950s and it is still one of the most studied in the field of Operations Research (OR). The great interest in the VRP is due to its practical importance, as well as the difficulty of solving it.

However, solving the VRP is far from a simple task since the problem is $\mathcal{NP}$-hard [1], which implies that no algorithm capable of finding optimal solutions in polynomial time is known. There has been lot of advances in the development of exact algorithms for dealing with the VRP, particularly those based on mathematical programming techniques. Unfortunately, to date, even the best exact algorithms can be very time consuming and seldom solve VRP instances with more than 150 customers. Combining (meta)heuristic and exact methods appears to be a very promising alternative in solving many CO problems. The interest in hybrid approaches has rapidly grown especially due to several encouraging results obtained by the fusion of these two methods (see [2]). The interaction between mathematical programming techniques and metaheuristics led to a new class of optimization algorithms called *matheuristics*. Nevertheless, the application of these kinds of approaches have not received much attention yet from the VRP literature (see [3–5]).

Most VRP heuristics usually focus on a particular type of problem. A relatively small number of works have suggested unified heuristic procedures for dealing with several variants (see, for example, [6–9]). Seen from a practical point of view, these non-specific approaches are highly relevant. For instance, VRP commercial packages must be prepared to face real-life problems of different classes. Cordeau et al. [10] even state that when talking about attributes for good heuristics, one should take into account not only the solution quality (accuracy) and computational time (speed), but also the simplicity and flexibility factors.

Given the above, one of the interests of this work is to propose a general hybrid algorithm for solving different VRPs. However, because of the huge number of existing variations it becomes virtually impossible to tackle all of them here. Therefore, it was thought advisable to turn attention only to a subset of variants, namely the following ones: (i) Capacitated VRP (with or without route duration limits), (ii) Asymmetric CVRP, (iii) Open VRP, (iv) VRP with Simultaneous Pickup and Delivery, (v) VRP with Mixed Pickup and Delivery, (vi) Multi-depot VRP, (vii) Multi-depot VRP with Mixed Pickup and Delivery.

The developed hybrid algorithm combines an exact procedure based on the Set Partitioning (SP) formulation with an Iterated Local Search (ILS) based heuristic. This strategy is quite similar to the classical two-phase petal algorithm (see [11]). The idea is to store a pool of routes generated during the heuristic execution and then solve a SP problem in order to extract the best combination of routes. However, unlike traditional petal algorithms and other SP based approaches to VRPs [12,13,4], the proposed hybrid algorithm includes some enhanced strategies.

* Corresponding author. Tel.: +55 83 3246 4527.
E-mail addresses: anand@ct.ufpb.br, anandsubraman@gmail.com (A. Subramanian), uchoa@producao.uff.br (E. Uchoa), satoru@ic.uff.br (L.S. Ochi).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.01.013

The first one is the cooperation between a Mixed Integer Programming (MIP) solver and the ILS heuristic (while solving the SP problem). This scheme was successfully applied in a previous work [14] to solve the Heterogeneous Fleet VRP (HFVRP) but its efficiency in terms of scalability was limited to approximately 200 customers. To overcome this limitation we introduce a new second strategy, which includes a reactive mechanism that dynamically controls the dimension of the SP models when dealing with large size instances that still allows for taking advantage of the exact procedure. As a result, new improved solutions were found for instances with up to 480 customers.

The remainder of this paper is organized as follows. Section 2 briefly describes the VRPs considered in this work. Section 3 explains the proposed hybrid algorithm. Section 4 contains the results obtained and a comparison with those reported in the literature. Section 5 presents the concluding remarks of this work.

2. A brief description of the VRPs considered in the present work

In this section we present a formal description of the VRPs considered here and we also point the best known algorithms, to our knowledge, for each variant. A complete literature review regarding such variants can be found in [15].

2.1. Capacitated Vehicle Routing Problem (CVRP)

The CVRP is considered to be the classical version of the VRP. A formal definition of the problem is as follows. Let $G=(V,E)$ be a complete graph with a set of vertices $V=\{0,\ldots,n\}$, where the vertex 0 represents the depot and the remaining ones the customers. Each edge $\{i,j\}\in E$ has a non-negative cost c_{ij} and each customer $i\in V'=V\backslash\{0\}$ has a demand d_i. Let $C=\{1,\ldots,m\}$ be the set of homogeneous vehicles with capacity Q. The CVRP consists in constructing a set of up to m routes in such a way that: (i) every route starts and ends at the depot; (ii) all demands are accomplished; (iii) the vehicle's capacity is not exceeded; (iv) a customer is visited by only a single vehicle; (v) the sum of costs is minimized. Some versions of this problem include route duration constraints. In such cases, there might be a travel time t_{ij} for each edge $\{i,j\}\in E$ and a service time s_i for each customer $i\in V'$. Among the best known heuristic algorithms are those Pisinger and Røpke [6], Mester and Bräysy [16], Nagata and Bräysy [17], Zachariadis and Kiranoudis [18] and Vidal et al. [9].

2.2. Asymmetric Capacitated Vehicle Routing Problem (ACVRP)

The ACVRP is a generalization of the CVRP where the cost between a pair of vertices is not necessarily symmetric, i.e., c_{ij} need not be equal to c_{ji}, $\forall i,j\in V$. Although this variant is more likely to be found in practice when compared to the CVRP (due to the existence of one-way streets in most urban zones), there are very few works that dealt with the ACVRP in the literature (see [19,3,20]).

2.3. Open Vehicle Routing Problem (OVRP)

The OVRP is a variant of the CVRP where the vehicles need not return to the depot after visiting the last customer of a given route. Any OVRP instance can be converted to an ACVRP instance by simply setting $c_{i0}=0,\forall i\in V$. Most authors also state that the primary objective is to minimize the number of vehicles, while the secondary objective is to minimize the sum of the travel costs. The most competitive heuristics are those of Pisinger and Røpke [6], Fleszar et al. [21], Repoussis et al. [22] and Zachariadis and Kiranoudis [23].

2.4. Vehicle Routing Problem with Simultaneous Pickup and Delivery (VRPSPD)

The VRPSPD is a generalization of the CVRP in which a customer $i\in V'$ has both a delivery demand d_i and also a pickup demand p_i. The heuristics of Subramanian et al. [24], Zachariadis and Kiranoudis [25] and Souza et al. [26] together produced the best known results.

2.5. Vehicle Routing Problem with Mixed Pickup and Delivery (VRPMPD)

The VRPMPD (a.k.a. the VRP with mixed backhauls) is a particular case of the VRPSPD, in which customers either have a pickup or a delivery demand but not both, i.e., if $d_i>0$, then $p_i=0$ and vice versa. The best known heuristics are those of Røpke and Pisinger [7] and Gajpal and Abad [27].

2.6. Multi-depot Vehicle Routing Problem (MDVRP)

Let G be the set of depots. The MDVRP is a generalization of the CVRP where more than one depot may be considered, that is, $|G|\geq 1$. Also, the vehicle must start and end at the same depot. Typically, the number of vehicles per each depot is given as an input data. Pisinger and Røpke [6] and Vidal et al. [9] developed the best heuristic approaches for the MDVRP.

2.7. Multi-depot Vehicle Routing Problem with Mixed Pickup and Delivery (MDVRPMPD)

The MDVRPMPD generalizes the VRPMPD by allowing $|G|\geq 1$ depot(s). The best known algorithm is the one of Røpke and Pisinger [7].

3. The hybrid algorithm

The proposed hybrid algorithm, called ILS-RVND-SP, essentially combines an ILS based heuristic, called ILS-RVND, and a SP approach. In this section we present a description of both the methods and how we merged them to efficiently tackle the seven variants considered in this work.

3.1. The ILS-RVND heuristic

In this section we briefly explain the general idea of the ILS-RVND heuristic. A highly detailed description of ILS-RVND can be found in [15,28]. An earlier version of this heuristic was applied in a parallel fashion by Subramanian et al. [24] to solve the VRPSPD and it is still remains as one of the best heuristic approaches, in terms of solution quality, proposed for the problem. Modified versions of ILS-RVND were also successfully applied to solve single-vehicle routing problems such as the Minimum Latency Problem (a.k.a. the Delivery Man Problem or the Cumulative Traveling Salesman Problem) [29] and the Traveling Salesman Problem with Mixed Pickup and Delivery [30].

The ILS-RVND heuristic is a multi-start procedure that uses insertion heuristics in the constructive phase, a Variable Neighborhood Descent with Random neighborhood ordering (RVND) in the local search phase and simple moves as perturbation mechanisms.

The insertion strategies are the Sequential Insertion Strategy (SIS) and the Parallel Insertion Strategy (PIS), while the insertion criteria are based on the Nearest Feasible Insertion Criterion (NFIC) and on a Modified Cheapest Feasible Insertion Criterion (MCFIC). At each iteration, the method randomly chooses a strategy and a criterion. In the case of NFIC, the cost of inserting a customer k after a customer i is simply given by c_{ik}. As for MCFIC, the cost of inserting a customer k between customers i and j is given by $(c_{ik}+c_{kj}-c_{ij})+\gamma(c_{0k}+c_{k0})$. The parameter γ controls the level of incentive of inserting customers located far from the depot.

The RVND is composed of well-known VRP inter-route neighborhood structures, namely those based on λ-interchanges [31] and Cross-exchange [32]; and also by specific ones, namely ShiftDepot and SwapDepot. With respect to the λ-interchanges, we consider Shift(λ,0), $\lambda \in \{1,2\}$, and Swap(λ_1,λ_2), $\lambda_1,\lambda_2 \in \{1,2\}$. As a result, five distinct neighborhood structures can be identified, i.e., Shift(1,0), Shift(2,0), Swap(1,1), Swap(1,2) and Swap(2,2). In Shift(λ,0), λ consecutive customers are moved from a route r_1 to a route r_2. In Swap(λ_1,λ_2), λ_1 consecutive customers from a route r_1 are interchanged with λ_2 consecutive customers from a route r_2. The Cross operator in our case consists of interchanging a segment from a route r_1 with a segment from a route r_2. ShiftDepot and SwapDepot were incorporated in the present work and they consist, respectively, of moving and swapping routes from a depot to another one. The best improvement strategy was adopted and the neighborhoods are explored exhaustively. Every time a route is modified due to an inter-route move an intra-route local search is performed using classical Traveling Salesman Problem neighborhood structures, more precisely, Reinsertion, Or-opt2 [33], Or-opt3 [33], 2-opt [34] and Exchange. Reinsertion consists of transferring a customer from its current position to another one in the same route. Or-opt2 and Or-opt3 make use of the same rationale but involve two and three consecutive customers, respectively. In 2-opt, two nonadjacent arcs are removed and another two are added in such a way that a new route is generated. Exchange is the intra-route version of Swap(1,1).

The perturbation mechanisms consist of performing multiple Swap(1,1) or Shift(1,1) moves. The Shift (1,1) consists of moving a customer from a route r_1 to a route r_2 and vice versa.

The ILS-RVND structure was slightly modified in order to store routes during its execution. Every time a local search is performed, the routes associated to the local optimal solution s may be added to a pool of routes (*RoutePool*). The method decides whether to add or not such routes based on the average number of customers per route (n/v) and on the deviation between the current best solution s^* and s (see Section 3.3). If this deviation, given by $(f(s)-f(s^*))/f(s^*)$, where $f(\cdot)$ is the cost function, is smaller than a threshold value (*tolerance*) then the routes of s are added to *RoutePool*. The input parameters of ILS-RVND are *MaxIter*, *MaxIterILS*, s_0, *RoutePool*, v and *tolerance*. The first parameter indicates the number of iterations, the second one is the maximum number of consecutive perturbations without improvements and the third one is an initial solution. Of course, if s_0 is provided then the procedure that generates an initial solution is skipped.

3.2. A set partitioning approach

Let $\mathcal{R}$ be the set of all possible routes of all vehicle types, $\mathcal{R}_i \subseteq \mathcal{R}$ be the subset of routes that contain customer $i \in V'$. Define y_j as the binary variable associated to a route $j \in \mathcal{R}$, and c_j as its cost. Consider the following basic SP formulation F1:

$$\text{Min} \quad \sum_{j \in \mathcal{R}} c_j y_j \tag{1}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{R}_i} y_j = 1 \quad \forall i \in V', \tag{2}$$

$$y_j \in \{0,1\} \quad \forall j \in \mathcal{R}. \tag{3}$$

The objective function (1) minimizes the sum of the costs by choosing the best combination of routes. Constraints (2) state that a single route from the subset $\mathcal{R}_i$ visits customer $i \in V'$. Since the enumeration of set $\mathcal{R}$ is an impractical task, ILS-RVND-SP only considers a subset of this set, usually limited to a few thousand routes. Formulation F1 is mainly suitable for variants such as the Fleet Size and Mix VRP [14] because the number of vehicles of each type is not predefined. Let $\mathcal{R}_u \subseteq \mathcal{R}$ be the set of routes associated with vehicle type $u \in M$ or with depot $u \in G$ and let m_u be an upper bound on the number of vehicles of a given type or available at a given depot. In order to deal with MDVRPs one can add the following constraints:

$$\sum_{j \in \mathcal{R}_u} y_j \le m_u \quad \forall u \in G. \tag{4}$$

Let v be the number of vehicles. For the remaining variants, one must include the constraint that ensure that the number of routes in the solution is equal to the number of vehicles available, i.e.,

$$\sum_{j \in \mathcal{R}} y_j = v. \tag{5}$$

It is important to mention that there are some instances of VRPs with homogeneous fleet that do not specify the number of vehicles, but ILS-RVND-SP fixes this value by using the number of vehicles of the current best solution. Although the solution space is reduced, this helps the problem to be solved more efficiently.

The pseudocode of the SP procedure is illustrated in Algorithm 1. Input parameter *MaxSPTime* corresponds to the time limit imposed to the Mixed Integer Programming (MIP) solver. It is assumed that the MIP solver uses a branch-and-bound or a branch-and-cut procedure. The algorithm starts by verifying if the number of vehicles should be minimized (e.g. OVRP) and if the number of vehicles of s^* is larger than the estimated lower bound on the number of vehicles ($v_{min} = \lceil (\sum_{i \in V'} d_i)/Q \rceil$). If so, solution s^* is stored in s' and the number of vehicles is decreased by one unit (lines 2–3). Next, the *SP_Model* is created (line 4) according to the VRP variant and the *Cutoff* value is initialized (line 5). The SP problem is given to a MIP solver (line 6), which calls the ILS-RVND heuristic whenever an incumbent solution is found (Procedure IncumbentCallback). If the solution s^* is improved in the IncumbentCallback, the *Cutoff* value is updated, but s^* is not given back to the solver since it may contain a route that does not belong to the subset of routes $\mathcal{R}$ of the SP model. The solver is interrupted if: (i) an optimal solution is found; (ii) $LB > Cutoff$; (iii) *MaxSPTime* is exceeded. If the primary objective is to minimize the number of vehicles and the solution s^* is infeasible, then the number of vehicles is incremented by one unit, s^* is restored and the MIP solver is called again (lines 7–10).

Algorithm 1. SP.

1: Procedure SP(s^*, *RoutePool*, *MaxSPTime*, v);
2: **if** v must be minimized **and** $v > v_{min}$ **then**
3: $v \leftarrow v-1$; $s' \leftarrow s^*$;
4: *SP_Model* ← CreateSetPartitioningModel(*RoutePool*, v);
5: *Cutoff* ← $f(s^*)$; {Only if $v = v_{min}$. Otherwise, *Cutoff* ← ∞}
6: s^* ← MIPSolver(*SP_Model*,s^*, *Cutoff*, *MaxSPTime*, IncumbentCallback(s^*));
7: **if** v must be minimized and **and** s^* *is infeasible* **then**
8: $s^* \leftarrow s'$; $v \leftarrow v+1$;
9: Update *SP_Model* {Increasing one *vehicle*};
10: s^* ← MIPSolver(*SP_Model*,s^*,*Cutoff*,*MaxSPTime*, IncumbentCallback(s^*));
11: **return** s^*;
12: **end** SP.

3.3. The hybrid algorithm

One of the challenges of designing a unified hybrid solution approach is to ensure that the MIP model is computationally tractable, regardless of the instance. For example, a SP model that exceeds the time limit only to solve its linear relaxation (e.g. due to an excessive number of routes) is not a suitable improving mechanism. On the other hand, a SP model that contains relatively few routes, is easily solved, but seldom finds improved solutions. Hence, it is necessary that the SP models generated throughout the algorithm find a balance between computational tractability and improvement potential. Experiments carried out in many instances with distinct characteristics indicated that the following simple pieces of data are crucial for estimating the dimension (number of routes) of a properly balanced SP model: (i) number of customers and (ii) the average number of customers per route.

With respect to (i), we developed two strategies for ILS-RVND-SP. The first one, called ILS-RVND-SP-a, is executed when the number of customers is less than or equal to a parameter $\mathcal{N}$. The idea of ILS-RVND-SP-a is very straightforward: the SP procedure is run only once at the end of the algorithm, after the ILS-RVND heuristic, as performed in [14]. The second one, called ILS-RVND-SP-b, is executed when $n > \mathcal{N}$. In this case, the SP procedure is called after each iteration of the ILS-RVND heuristic. Both strategies are completely independent, as well as some of their parameters, namely: *MaxIter*-a, *MaxIter*-b, *MaxIterILS*-a, *MaxIterILS*-b, *TDev*-a, *TDev*-b. The parameter *TDev* is described next.

With respect to (ii), we do the following. Let $\mathcal{A}$ be a parameter. It has been observed that when the ratio between the number of customers and the number of vehicles is smaller than $\mathcal{A} = 11$, the SP models tend to become harder. In such cases, we only add the routes of a solution to the SP model if its deviation when compared to the incumbent solution is smaller than a given threshold *TDev*. However, this parameter is difficult to tune, especially in ILS-RVND-SP-b. To overcome this issue we implemented a reactive approach that dynamically adjusts its value throughout the execution of the algorithm, as will be further explained.

The pseudocode of ILS-RVND-SP and ILS-RVND-SP-a will be omitted since they are quite simple. Algorithm 2 shows the pseudocode of ILS-RVND-SP-b. Firstly, *tolerance* (threshold deviation) is set to a given value according to the average number of customers per route (lines 2–5). In the main loop (lines 7–24), the ILS-RVND heuristic is executed with a single iteration (line 8) and the SP procedure is repeatedly called while there is any improvement over the best current solution (lines 10–21). When no improvement is observed, the non-permanent routes (short-term memory), in this case those generated on that particular iteration, are removed from *RoutePool* (line 16). After each call to the SP procedure, the algorithm may update the value of *tolerance*, in case $n/v < \mathcal{A}$, according to the following conditions. If the SP model is solved at the root node, meaning that the problem is easy, then *tolerance* is increased by one-tenth of *TDev*-b (lines 17–18). If the time limit is exceeded then *tolerance* is decreased by one-tenth of *TDev*-b (lines 19–20). If there is any improvement at the end of a given iteration, the incumbent solution s^* is updated and the associated routes are permanently added (long-term memory) to *RoutePool* (lines 22–24). Such routes are never removed from the pool.

Algorithm 2. ILS-RVND-SP-b.

1: Procedure ILS-RVND-SP-b(*MaxIter*−b, *MaxIterILS*−b, *RoutePool*, v, *TDev*−b, *MaxSPTime*);
2: **if** $n/v < \mathcal{A}$ **then**
3:   $tolerance \leftarrow TDev - b$;
4: **else**
5:   $tolerance \leftarrow 1$;
6: $iter \leftarrow 0$; $s^* \leftarrow \emptyset$; $s_0 \leftarrow NULL$;
7: **while** $iter < MaxIter - b$ **do**
8:   $s \leftarrow$ ILS-RVND(1, *MaxIterILS*−b, s_0, *RoutePool*, v, *tolerance*);
9:   *improvement* ← true;
10:   **while** *improvement* **do**
11:     $s' \leftarrow SP(s, RoutePool, MaxSPTime, v)$;
12:     **if** $f(s') < f(s)$ **then**
13:       $s \leftarrow s'$;
14:     **else**
15:       *improvement* ← false;
16:       Remove non-permanent routes from *RoutePool*;
17:     **if** $n/v < \mathcal{A}$ **and** $Time > MaxSPTime$ **then**
18:       $tolerance \leftarrow tolerance - 0.1 \times TDev - b$;
19:     **if** $n/v < \mathcal{A}$ **and** Problem solved at the root node **then**
20:       $tolerance \leftarrow tolerance + 0.1 \times TDev - b$;
21:     $iter \leftarrow iter + 1$;
22:   **if** $f(s) < f(s^*)$ **or** s^* is empty **then**
23:     $s^* \leftarrow s$;
24:     Add routes associated to s^* permanently to the pool;
25: **return** s^*;

4. Computational results

The algorithm ILS-RVND-SP was coded in C++ and the tests were executed on an Intel® Core™ i7 with 2.93 GHz and 8 GB of RAM running under Ubuntu Linux 64 bits. CPLEX 12.2 was used as a MIP solver. The computational experiments were carried out using a single thread and the algorithm was executed 10 times for each instance.

Table 1 shows the values of the parameters used by ILS-RVND-SP, which were calibrated after preliminary experiments. The most crucial parameters are $\mathcal{N}$ and $\mathcal{A}$. The values adopted for the remaining ones are not so critical, which is reflected in the round numbers chosen.

In the following tables, *Instance* denotes the test-problem, n is the number of customers, $|G|$ is the number of depots, v is the number of vehicles available per depot, *BKS* represents the Best Known Solution (BKS) reported in the literature, *Best Sol.*, *Avg. Sol.* and *Time* (s) indicate, respectively, the best solution, the average solution and the average computational time in seconds associated to the corresponding work, *Gap* denotes the gap, given by

Table 1
Values of the parameters used by ILS-RVND-SP.

Parameter	Value
$\mathcal{N}$	150
$\mathcal{A}$	11
MaxIter-a	50
MaxIter-b	100
MaxIterILS-a	$n+0,5 \times v$
MaxIterILS-b	2000
Tdev-a	0.05
Tdev-b	0.005
γ	Random value of the set $\{0.00, 0.05, 0.10, \ldots, 1, 70\}$ [24]
MaxSPTime (s)	60

$100 \times ((z_{\text{ILS-RVND-SP}} - z_{\text{BKS}})/z_{\text{BKS}})$, between the best solution found by ILS-RVND-SP and the BKS, *Avg. Gap* corresponds to the gap between the average solution found by ILS-RVND-SP and the best known solution, *Scaled time* (s) is the approximate average scaled time in seconds of each machine using the factors suggested by the benchmarks of Dongarra [35], when solving a system of equations of order 1000, with respect to our i7 2.93 GHz (5839 Mflop/s). The BKSs are highlighted in boldface and the improved solutions are underlined.

4.1. CVRP

The developed hybrid algorithm was tested on the instances of the A, B, E, M, P series and all known optimal solutions were easily determined. Table 2 only shows the results obtained on the three open instances of the M-series, namely: M-n151-k12, M-n200-k16 and M-n200-k17. The proposed algorithm was found capable of improving the result of the second one and to equal the BKSs of the first and third ones. Table 3 contains the results found on the instances suggested in [36] and a comparison with those reported in [13], [6] (ALNS 50K) and [16,17,9]. ILS-RVND-SP was successful to equal the BKS in 13 of the 14 instances and the average gap between the average solutions found by ILS-RVND-SP and the BKSs was 0.08%. Table 4 illustrates a comparison, in terms of average solution between the results obtained by ILS-RVND-SP and those found in [6] (ALNS 50K) and [17,18] for the instances proposed in [37]. It can be seen that the ILS-RVND-SP outperformed the algorithm developed in [6], but it is not as effective as those presented in [17,9] in terms of average solution quality. Yet, the average gap between the average solutions found by ILS-RVND-SP and the BKSs was only 0.55%, a value smaller than the one obtained by the general heuristic proposed in [6]. On the other hand, in spite of obtaining slightly lower quality solutions, we believe that the proposed algorithm is simpler than those developed in [17,9,18].

4.2. ACVRP

ILS-RVND-SP was tested in the ACVRP instances suggested in [19]. The capacity of the vehicle is the same ($Q=1000$) and the number of customers varies between 33 and 70. Pessoa et al. [20] also considered the same data set of [19] but with different capacities (150, 250 and 500). The instances with $Q=150$ are not considered, since they can be easily solved using F1 (most feasible routes contain very few customers and it is practical to perform a complete enumeration), thus leading to a total of 24 instances. Table 5 shows the results found for the ACVRP instances. All the known optimal solutions were consistently found by ILS-RVND-SP. Regarding the two instances where the optimal solutions is

Table 2
Results found for the open instances of the M-series.

Instance	n	v	BKS	ILS-RVND-SP				
				Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
M-n151-k12	150	12	**1015**[a]	**1015**	1015.5	0.00	0.05	37.12
M-n200-k16	199	16	**1285**[b]	**1278**	1285.8	−0.54	0.06	772.01
M-n200-k17	199	17	**1275**[a]	**1275**	1279.9	0.00	0.38	513.00
					Avg.	−0.18	0.17	440.71

[a] Value presented in [38].
[b] Value obtained in [5].

Table 3
Results found for the CVRP instances proposed in [36].

Instance	n	v	BKS	Rochat and Taillard		Pisinger and Røpke		Mester and Bräysy		Nagata and Bräysy		Vidal et al.		ILS-RVND-SP				
				Best Sol.	Time	Best Sol.	Time[a] (s)	Best Sol.	Time[b] (s)	Best Sol.	Time[c] (s)	Best. Sol.	Time[d] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
C1	50	5	**524.61**[e]	**524.61**	–	**524.61**	21	**524.61**	0.2	**524.61**	4.3	**524.61**	25.8	**524.61**	**524.61**	0.00	0.00	1.48
C2	75	10	**835.26**	**835.26**	–	**835.26**	36	**835.26**	5.5	**835.26**	22.3	**835.26**	57.6	**835.26**	**835.26**	0.00	0.00	13.52
C3	100	8	**826.14**	**826.14**	–	**826.14**	78	**826.14**	1.0	**826.14**	17.1	**826.14**	76.2	**826.14**	**826.14**	0.00	0.00	12.49
C12	100	10	**819.56**	**819.56**	–	**819.56**	73	**819.56**	0.2	**819.56**	8.1	**819.56**	50.4	**819.56**	**819.56**	0.00	0.00	5.23
C11	120	7	**1042.11**	**1042.11**	–	**1042.11**	113	**1042.11**	1.1	**1042.11**	21.0	**1042.11**	69.0	**1042.11**	**1042.11**	0.00	0.00	20.66
C4	150	12	**1028.42**	**1028.42**	–	1029.56	160	**1028.42**	10.2	**1028.42**	75.2	**1028.42**	172.2	**1028.42**	1028.73	0.00	0.03	53.48
C5	199	17	**1291.29**	1291.45	–	1297.12	219	**1291.29**	2160.0	1291.45	302.1	1291.45	356.4	1291.45	1293.18	< 0.01	0.13	625.17
C6	50	6	**555.43**	**555.43**		**555.43**	21	**555.43**	4.2	**555.43**	5.1	**555.43**	28.8	**555.43**	556.49	0.00	0.19	0.93
C7	75	11	**909.68**	**909.68**	–	**909.68**	36	**909.68**	0.8	**909.68**	38.9	**909.68**	65.4	**909.68**	910.00	0.00	0.03	5.05
C8	100	9	**865.94**	**865.94**	–	**865.94**	78	**865.94**	0.8	**865.94**	23.5	**865.94**	68.4	**865.94**	**865.94**	0.00	0.00	7.61
C14	100	11	**866.37**	**866.37**	–	**866.37**	73	**866.37**	1.7	**866.37**	13.2	**866.37**	71.4	**866.37**	**866.37**	0.00	0.00	7.12
C13	120	11	**1541.14**	**1541.14**	–	1542.86	113	**1541.14**	13.5	**1541.14**	106.9	**1541.14**	169.8	**1541.14**	1544.07	0.00	0.19	80.24
C9	150	14	**1162.55**	**1162.55**	–	1163.68	160	**1162.55**	25.8	**1162.55**	135.6	**1162.55**	151.8	**1162.55**	1164.11	0.00	0.13	82.50
C10	199	18	**1395.85**	**1395.85**	–	1405.88	219	1401.12	52.2	**1395.85**	390.6	**1395.85**	493.2	**1395.85**	1402.03	0.00	0.44	496.07
															Avg.	0.00	0.08	100.83
Scaled Time (s)					–			54.48	68.08			49.62		72.24				100.83

[a] Average of 10 runs on a Pentium IV 3.0 GHz (3181 Mflop/s).
[b] Average of 10 runs on a Pentium IV 2.8 GHz (2444 Mflop/s).
[c] Average of 10 runs on an Opteron 2.4 GHz (3485 Mflop/s).
[d] Average of 10 runs on an Opteron 2.4 GHz scaled for a Pentium IV 3.0 GHz.
[e] Optimality proved.

Table 4
Results found for the CVRP instances proposed in [37].

Instance	n	v	BKS	Pisinger and Røpke		Nagata and Bräysy		Vidal et al.		Zachariadis and Kiranoudis		ILS-RVND-SP				
				Avg. Sol.[a]	Time[b] (s)	Avg. Sol.[a]	Time[c] (s)	Avg. Sol.[a]	Time[d] (s)	Avg. Sol.[a]	Time[e] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
G17	240	22	**707.76**[f,g]	710.59	304	707.78	582.4	708.09	423.6	708.94	962.3	**707.76**	707.81	0.00	0.01	937.59
G13	252	26	**857.19**[f,g]	874.24	285	858.42	921.9	859.64	561.6	860.44	1189.3	**857.19**	860.00	0.00	0.33	910.35
G9	255	14	**579.71**[g]	590.33	437	581.46	1043.3	581.79	973.2	584.66	929.4	583.24	585.21	0.61	0.95	1720.76
G18	300	27	**995.13**[f,g]	1007.84	387	995.91	1465.9	998.44	993.6	997.74	1718.6	995.65	997.85	0.05	0.27	2297.62
G14	320	30	**1080.55**[f,g]	1103.53	393	1080.84	1239.3	1082.41	847.2	1083.55	1187.4	**1080.55**	1082.15	0.00	0.15	1513.32
G10	323	16	**736.26**[g]	751.36	616	739.56	1617.5	739.86	1551.6	739.86	1271.4	741.96	744.17	0.77	1.07	3229.35
G19	360	33	**1365.60**[g]	1377.88	449	1366.70	2115.6	1367.83	1674.6	1370.77	1824.2	1366.29	1367.25	0.05	0.12	2917.31
G15	396	33	**1337.92**[g]	1366.23	468	1344.32	1872.2	1343.52	2349.0	1344.41	1658.8	1347.13	1349.23	0.69	0.85	3265.68
G11	399	18	**912.84**[g]	926.57	761	916.27	2337.5	916.44	2736.6	919.52	1392.2	921.46	922.93	0.94	1.11	5978.97
G20	420	38	**1818.32**[g]	1834.70	488	1821.65	2824.7	1822.02	2293.8	1829.57	1199.3	1821.16	1823.52	0.16	0.29	4997.31
G16	480	37	**1612.50**[g]	1645.67	549	1622.26	2616.2	1621.02	3496.2	1623.42	1848.5	1624.55	1627.76	0.75	0.95	4835.12
G12	483	19	**1102.69**[g]	1125.22	911	1108.21	3561.9	1106.73	5740.2	1110.65	1282.3	1113.30	1116.52	0.96	1.25	10 410.70
G5	200	5	**6460.98**	6482.49	629	**6460.98**	164.7	**6460.98**	153.6	**6460.98**	989.6	**6460.98**	**6460.98**	0.00	0.00	978.69
G1	240	9	**5623.47**[g]	5662.57	93	5632.05	3393	5627.00	700.8	5637.99	907.7	5657.74	5671.65	0.61	0.86	994.59
G6	280	7	**8412.88**[h]	8543.30	876	8413.41	830.3	8412.90	502.8	8412.90	1091.6	8412.90	8412.90	0.00	0.00	2455.56
G2	320	10	**8404.61**[g]	8487.94	672	8440.25	1726.2	8446.65	1245.0	8457.92	1249.4	8447.92	8449.82	0.52	0.54	2659.68
G7	360	9	**10 102.70**[g]	10 265.15	941	10 186.93	2179.7	10 157.63	1376.4	10 192.47	1885.5	10 195.58	10 195.60	0.92	0.92	4410.84
G3	400	10	**11 036.22**	11 052.72	1015	**11 036.22**	2606.8	**11 036.22**	1679.4	**11 036.22**	1164.0	**11 036.22**	**11 036.22**	0.00	0.00	6064.98
G8	440	10	**11 635.30**[g]	11 766.07	1011	11 691.54	5776.7	11 646.58	2440.2	11 674.43	1657.4	11 710.47	11 774.40	0.65	1.20	7541.75
G4	480	10	**13 592.88**[f]	13 748.50	1328	13 618.55	3841.6	13 624.52	2620.2	13 632.59	1019.0	13 624.53	13 624.53	0.23	0.23	10 644.50
				Avg. Gap (%)	1.34	Avg. Gap (%)	0.27	Avg. Gap (%)	0.26	Avg. Gap (%)	0.42		Avg.	0.40	0.55	3938.23
Scaled Time (s)					343.57		1274.79		935.93		632.98					3938.23

[a] Average of 10 runs.
[b] Average of 10 runs on a Pentium IV 3.0 GHz (3181 Mflop/s).
[c] Average of 10 runs on an Opteron 2.4 GHz (3485 Mflop/s).
[d] Average of 10 runs on an Opteron 2.4 GHz scaled for a Pentium IV 3.0 GHz.
[e] Average of 10 runs on a T5500 1.66 GHz (2791 Mflop/s).
[f] Found in [17].
[g] Found in [9].
[h] Found in [16].

Table 5
Results found for the ACVRP instances proposed in [19,20].

Instance	n	v	BKS	ILS-RVND-SP				
				Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
A034-02f	34	2	**1406**[a]	**1406**	**1406.00**	0.00	0.00	0.57
A034-04f	34	4	**1773**[a]	**1773**	**1773.00**	0.00	0.00	0.47
A034-08f	34	8	**2672**[a]	**2672**	**2672.00**	0.00	0.00	0.47
A036-03f	36	3	**1644**[a]	**1644**	**1644.00**	0.00	0.00	0.64
A036-05f	36	5	**2110**[a]	**2110**	**2110.00**	0.00	0.00	0.61
A036-10f	36	10	**3338**[a]	**3338**	**3338.00**	0.00	0.00	5.67
A039-03f	39	3	**1654**[a]	**1654**	**1654.00**	0.00	0.00	0.69
A039-06f	39	6	**2289**[a]	**2289**	**2289.00**	0.00	0.00	0.60
A039-12f	39	12	**3705**[a]	**3705**	**3705.00**	0.00	0.00	0.77
A045-03f	45	3	**1740**[a]	**1740**	**1740.00**	0.00	0.00	1.06
A045-06f	45	6	**2303**[a]	**2303**	**2303.00**	0.00	0.00	0.88
A045-11f	45	11	**3544**[a]	**3544**	**3544.00**	0.00	0.00	2.46
A048-03f	48	3	**1891**[a]	**1891**	**1891.00**	0.00	0.00	1.26
A048-05f	48	5	**2283**[a]	**2283**	**2289.50**	0.00	0.28	1.79
A048-10f	48	10	**3325**[a]	**3325**	**3325.60**	0.00	0.02	1.29
A056-03f	56	3	**1739**[a]	**1739**	**1740.00**	0.00	0.06	2.09
A056-05f	56	5	**2165**[a]	**2165**	**2165.00**	0.00	0.00	3.53
A056-10f	56	10	**3263**[a]	**3263**	**3264.50**	0.00	0.05	2.26
A065-03f	65	3	**1974**[a]	**1974**	**1974.00**	0.00	0.00	3.08
A065-06f	65	6	**2567**[a]	**2567**	2571.70	0.00	0.18	3.30
A065-12f	65	12	**3902**[a]	**3902**	3904.90	0.00	0.07	3.41
A071-03f	71	3	**2054**[a]	**2054**	**2054.00**	0.00	0.00	4.46
A071-05f	71	5	**2475**[b]	**2457**	**2457.90**	−0.73	−0.69	6.72
A071-10f	71	10	**3486**[b]	**3486**	3492.90	0.00	0.20	5.61
					Avg.	−0.03	0.01	2.24

[a] Optimality proved.
[b] Value presented in [20].

Table 6
Results found for the OVRP instances proposed in [36,39,45].

Instance	n	v_{min}	BKS	$vbest$	Pisinger and Røpke			Fleszar et al.			Repoussis et al.			Zachariadis and Kiranoudis			ILS-RVND-SP					
					Best Sol.	v	Time[a] (s)	Best Sol.	v	Time[b] (s)	Best Sol.	v	Time[c] (s)	Best Sol.	v	Time[d] (s)	Best Sol.	v	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
C1	50	5	**416.06**[e]	5	**416.06**	5	23	**416.06**	5	1.0	**416.06**	5	98	**416.06**	5	28	**416.06**	5	**416.06**	0.00	0.00	1.78
F11	71	4	**177.00**[e]	4	**177.00**	4	104	178.66	4	6.2	**177.00**	4	264	**177.00**	4	132	**177.00**	4	177.21	0.00	0.12	4.41
C2	75	10	**567.14**[e]	10	**567.14**	10	53	**567.14**	10	2.3	**567.14**	10	143	**567.14**	10	72	**567.14**	10	**567.14**	0.00	0.00	6.44
C3	100	8	**639.74**[e]	8	641.76	8	128	641.40	8	9.5	**639.74**	8	330	**639.74**	8	97	**639.74**	8	639.81	0.00	0.01	15.67
C12	100	10	**534.24**[e]	10	**534.24**	10	118	534.40	10	6.7	**534.24**	10	363	**534.24**	10	47	**534.24**	10	**534.24**	0.00	0.00	5.74
C11	120	7	**682.12**	7	**682.12**	7	141	**682.12**	7	10.7	**682.12**	7	318	**682.12**	7	76	**682.12**	7	**682.12**	0.00	0.00	23.54
F12	134	7	**769.55**	7	770.17	7	359	769.66	7	75.4	**769.55**	7	753	**769.55**	7	278	**769.55**	7	770.00	0.00	0.06	40.51
C4	150	12	**733.13**	12	733.13	12	279	737.82	12	45.4	**733.13**	12	613	**733.13**	12	204	**733.13**	12	**733.13**	0.00	0.00	27.77
C5	199	16	**893.39**	16	896.08	16	237	905.96	16	17.1	894.11	16	1272	**893.39**	16	332	**883.50**	16	895.55	−1.11	0.24	1579.45
C6	50	5	**412.96**	6	**412.96**	6	31	**412.96**	6	75.8	**412.96**	6	215	–	–	–	**412.96**	6	**412.96**	0.00	0.00	1.24
C7	75	10	**583.19**	10	**583.19**	10	33	596.47	10	22.3	584.15	10	367	–	–	–	**583.19**	10	582.07	0.00		7.29
C8	100	8	**644.63**	9	645.16	9	114	**644.63**	9	587.6	**644.63**	9	510	–	–	–	**644.63**	9	644.95	0.00	0.05	9.21
C14	100	10	**591.87**	11	**591.87**	11	75	**591.87**	11	389	**591.87**	11	411	–	–	–	**591.87**	11	**591.87**	0.00	0.00	22.76
C13	120	7	**904.04**	11	909.80	11	116	**904.94**	11	1820.1	910.26	11	890	–	–	–	**899.16**	11	**904.02**	−0.54	0.00	37.35
C9	150	12	**757.84**	13	**757.84**	13	185	760.06	13	1094.1	764.56	13	933	–	–	–	757.91	13	759.38	0.01	0.20	38.93
C10	199	16	**875.67**	17	**875.67**	17	224	**875.67**	17	1252.4	888.46	17	1678	–	–	–	**874.71**	17	877.68	−0.11	0.23	472.91
																			Avg.	−0.11	0.06	143.44
Scaled time (s)					75.59			100.75			25.68			61.38								143.44 (189.48[f])
O1	200	5	**6018.52**	5	–	–	–	–	–	–	**6018.52**	5	161.55[g]	**6018.52**	5	635	**6018.52**	5	**6018.52**	0.00	0.00	1270.81
O2	240	9	**4557.38**	9	–	–	–	–	–	–	4583.70	9	219.09[g]	**4557.38**	9	832	**4544.46**	9	**4551.74**	−0.28	−0.12	1247.37
O3	280	7	**7731.00**	7	–	–	–	–	–	–	7733.77	7	263.05[g]	**7731.00**	7	921	**7721.16**	7	**7728.77**	−0.13	−0.03	2008.76
O4	320	10	**7253.20**	10	–	–	–	–	–	–	7271.24	10	297.71[g]	**7253.20**	10	1009	**7215.48**	10	**7229.56**	−0.52	−0.33	2663.24
O5	360	8	**9193.15**	8	–	–	–	–	–	–	9254.15	8	487.85[g]	**9193.15**	8	1590	**9180.93**	8	9205.01	−0.13	0.13	5702.38
O6	400	9	**9793.72**	9	–	–	–	–	–	–	9821.09	9	433.53[g]	**9793.72**	9	1108	**9773.83**	9	**9784.52**	−0.20	−0.09	5065.72
O7	440	10	**10 347.70**	10	–	–	–	–	–	–	10 363.4	10	552.90[g]	**10 347.70**	10	1094	**10 326.57**	10	**10 342.1**	−0.20	−0.05	6140.55
O8	480	10	**12 415.36**	10	–	–	–	–	–	–	12 428.2	10	590.78[g]	**12 415.36**	10	1273	**12 389.43**	10	**12 393.4**	−0.21	−0.18	6653.41
																			Avg.	−0.21	−0.08	3844.03
Scaled time (s)							–			–			89.14			505.60						3844.03

[a] Average of 10 runs on a Pentium IV 3.0 GHz (3181 Mflop/s).
[b] Best run on a Pentium M 2.0 GHz (1738 Mflop/s).
[c] Best run on a scaled to a Pentium II 400 MHz (262 Mflop/s).
[d] Average of 10 runs on a T5500 1.66 GHz (2791 Mflop/s).
[e] Optimality proved.
[f] Average of 10 runs considering the following instances: C1, F11, C2, C3, C12, C11, F12, C4 and C5.
[g] Best run on a Pentium IV 2.8 GHz.

Table 7
Results found for the VRPSPD instances proposed in [41].

Instance	n	v	BKS	Gajpal and Abad		Zachariadis et al.		Subramanian et al.		ILS-RVND-SP				
				Best Sol.	Time[a] (s)	Best Sol.	Time[b] (s)	Best Sol.	Time[c] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
CMT1X	50	3	**466.77**[d]	**466.77**	5.00	469.80	2.1	**466.77**	2.3	**466.77**	**466.77**	0.00	0.00	2.08
CMT1Y	50	3	**466.77**[d]	**466.77**	5.00	469.80	3.8	**466.77**	2.3	**466.77**	**466.77**	0.00	0.00	1.97
CMT2X	75	6	**684.21**	**684.21**	41.25	**684.21**	5.4	**684.21**	6.4	**684.21**	684.78	0.00	0.08	12.79
CMT2Y	75	6	**684.21**	684.94	22.25	**684.21**	6.8	**684.21**	6.4	**684.21**	684.59	0.00	0.06	10.83
CMT3X	100	5	**721.27**[d]	721.40	377.50	**721.27**	11.9	**721.27**	12.1	**721.27**	721.46	0.00	0.03	17.69
CMT3Y	100	5	**721.27**[d]	721.40	43.75	**721.27**	11	**721.27**	12.3	**721.27**	721.50	0.00	0.03	17.61
CMT12X	100	5	**662.22**	663.01	36.25	**662.22**	9.3	**662.22**	10.3	**662.22**	663.44	0.00	0.18	9.07
CMT12Y	100	5	**662.22**	663.50	39.25	**662.22**	4.8	**662.22**	10.8	**662.22**	663.12	0.00	0.14	9.34
CMT11X	120	4	**833.92**	839.66	57.25	**833.92**	21.2	**833.92**	18.9	846.23	848.65	1.48	1.77	51.82
CMT11Y	120	4	**833.92**	840.19	52.75	**833.92**	14.4	**833.92**	19.0	846.23	848.74	1.48	1.78	48.63
CMT4X	150	7	**852.46**	854.12	131.75	**852.46**	29.6	**852.46**	30.9	**852.46**	853.02	0.00	0.07	98.03
CMT4Y	150	7	**852.46**	855.76	140.25	**852.46**	27.4	**852.46**	31.6	**852.46**	852.73	0.00	0.03	80.63
CMT5X	199	10	**1029.25**	1034.87	377.50	1030.55	62.8	**1029.25**	71.5	**1029.25**	1029.52	0.00	0.03	1786.74
CMT5Y	199	10	**1029.25**	1037.34	393.50	1030.55	47.7	**1029.25**	69.6	**1029.25**	**1029.25**	0.00	0.00	1726.18
CMT6X	50	7	**555.43**	**555.43**	14.00	–	–	–	–	**555.43**	557.35	0.00	0.35	1.04
CMT6Y	50	7	**555.43**	**555.43**	13.75	–	–	–	–	**555.43**	557.10	0.00	0.30	1.08
CMT7X	75	13	**900.12**	**900.12**	47.75	–	–	–	–	**900.12**	901.02	0.00	0.10	4.55
CMT7Y	75	13	**900.54**	**900.54**	46.25	–	–	–	–	**900.12**	901.08	−0.05	0.06	4.87
CMT8X	100	10	**865.50**	**865.50**	80.75	–	–	–	–	**865.50**	**865.50**	0.00	0.00	7.36
CMT8Y	100	10	**865.50**	**865.50**	77.75	–	–	–	–	**865.50**	**865.50**	0.00	0.00	7.74
CMT14X	100	11	**821.75**	**821.75**	78.50	–	–	–	–	**821.75**	**821.75**	0.00	0.00	5.42
CMT14Y	100	11	**821.75**	**821.75**	74.75	–	–	–	–	**821.75**	**821.75**	0.00	0.00	5.48
CMT13X	120	12	**1542.86**	**1542.86**	160.25	–	–	–	–	**1542.86**	1543.54	−0.03	0.04	68.72
CMT13Y	120	12	**1542.86**	**1542.86**	160.25	–	–	–	–	**1542.86**	1544.42	0.00	0.10	73.49
CMT9X	150	16	**1161.54**	**1161.54**	300.00	–	–	–	–	**1160.68**	1161.77	−0.07	0.02	64.43
CMT9Y	150	16	**1161.54**	**1161.54**	291.75	–	–	–	–	**1160.68**	1162.59	−0.07	0.09	80.86
CMT10X	199	20	**1386.29**	**1386.29**	773.50	–	–	–	–	**1373.40**	**1379.19**	−0.93	−0.51	552.81
CMT10Y	199	20	**1395.04**	**1395.04**	757.50	–	–	–	–	**1373.40**	**1377.03**	−1.55	−1.29	547.39
											Avg.	0.01	0.12	189.24
Scaled time (s)					55.65		8.83		–					189.24 (276.67[e])

[a] Best run on a Xeon 2.4 GHz (1978 Mflop/s).
[b] Average of 10 runs on a T5500 1.66 GHz (2791 Mflop/s).
[c] Average of 50 runs on a cluster with 32 SMP nodes, where each node consists of two Intel Xeon 2.66 GHz (wall clock).
[d] Optimality proved.
[e] Average of 10 runs considering the following instances: CMT1X, CMT1Y, CMT2X, CMT2Y, CMT3X, CMT3Y,CMT12X, CMT12Y, CMT11X. CMT11Y, CMT4X, CMT4Y, CMT5X and CMT5Y.

Table 8
Results found for the VRPSPD instances proposed in [43].

Instance	n	v	BKS	Souza et al.		Zachariadis et al.		Subramanian et al.		ILS-RVND-SP				
				Best Sol.	Time[a] (s)	Best Sol.	Time[b] (s)	Best Sol.	Time[c] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
r101	100	12	**1009.95**[d]	**1009.95**	35.7	**1009.95**	28.7	**1009.95**	15.8	**1009.95**	1010.08	0.00	0.01	65.42
r201	100	3	**666.20**[d]	**666.20**	39.6	**666.20**	31.4	**666.20**	16.0	**666.20**	**666.20**	0.00	0.00	15.71
c101	100	16	**1220.18**[d]	**1220.18**	18.3	**1220.18**	18.5	**1220.18**	10.4	**1220.18**	1220.43	0.00	0.02	12.93
c201	100	5	**662.07**[d]	**662.07**	16.6	**662.07**	23.5	**662.07**	8.8	**662.07**	**662.07**	0.00	0.00	9.77
rc101	100	10	**1059.32**[d]	**1059.32**	12.8	**1059.32**	23.8	**1059.32**	11.1	**1059.32**	**1059.32**	0.00	0.00	16.89
rc201	100	3	**672.92**[d]	**672.92**	24.0	**672.92**	21.2	**672.92**	7.3	**672.92**	**672.92**	0.00	0.00	11.42
r1_2_1	200	23	**3357.64**	**3357.64**	175.8	3375.19	84.6	3360.02	66.2	**3353.80**	**3355.04**	−0.11	−0.08	1142.05
r2_2_1	200	5	**1665.58**[d]	**1665.58**	103.4	**1665.58**	72.7	**1665.58**	45.3	**1665.58**	**1665.58**	0.00	0.00	1425.88
c1_2_1	200	28	**3629.89**	3636.74	117.6	3641.89	57.0	**3629.89**	87.4	**3628.51**	3636.53	−0.04	0.18	2874.50
c2_2_1	200	9	**1726.59**	**1726.59**	127.8	1726.73	67.3	**1726.59**	65.0	**1726.59**	**1726.59**	0.00	0.00	1365.93
rc1_2_1	200	23	**3306.00**	3312.92	299.3	3316.94	83.4	**3306.00**	71.7	**3303.70**	3306.73	−0.07	0.02	1293.53
rc2_2_1	200	5	**1560.00**[d]	**1560.00**	77.5	1560.00	74.4	**1560.00**	44.7	**1560.00**	**1560.00**	0.00	0.00	1361.87
r1_4_1	400	54	**9605.75**[e]	9627.43	2928.3	9668.18	421.5	9618.97	481.6	**9519.45**	**9539.56**	−0.90	−0.69	9177.90
r2_4_1	400	10	**3551.38**	3582.08	768.6	3560.73	352.0	**3551.38**	459.2	**3546.49**	**3549.49**	−0.14	−0.05	9086.79
c1_4_1	400	63	**11 098.21**	**11 098.21**	1510.4	11 125.14	384.6	11 099.54	546.2	**11 047.19**	**11 075.60**	−0.46	−0.20	8016.83
c2_4_1	400	15	**3546.10**	3596.37	569.0	3549.20	341.1	**3546.10**	488.6	**3539.50**	**3543.65**	−0.19	−0.07	10 691.30
rc1_4_1	400	52	**9520.06**	9535.46	2244.2	**9520.06**	412.7	9536.77	513.4	**9447.53**	**9478.12**	−0.76	−0.44	10 867.10
rc2_4_1	400	11	**3403.70**	3422.11	3306.8	3414.90	264.7	**3403.70**	422.6	**3403.70**	**3403.70**	0.00	0.00	8326.18
											Avg.	−0.15	−0.07	3653.44
Scaled time (s)					329.35		73.53		–					3653.44

[a] Best run on an Intel Core 2 Duo 1.66 GHz (2791 Mflop/s).
[b] Average of 10 runs T5500 1.66 GHz ((2791 Mflop/s)).
[c] Average of 50 runs on a cluster with 32 SMP nodes, where each node consists of two Intel Xeon 2.66 GHz (wall clock).
[d] Optimality proved.
[e] Found in [24].

not known, the proposed algorithm was capable of improving the BKS in one of them and to equal the best result in the other one.

4.3. OVRP

Table 6 presents the results found by ILS-RVND-SP in the set of instances proposed in [36,39] and in the set of instances suggested in [40], as well as a comparison with those pointed out in [6] (ALNS 50K) and [21–23]. Regarding the set of instances introduced in [36,39], ILS-RVND-SP was capable of obtaining the BKS in 12 cases and to improve another three solutions, but it failed to find one BKS. Furthermore, ILS-RVND-SP also failed to always obtain solutions with the minimum number of vehicles on instance C7. The average gap between the average solutions obtained by ILS-RVND-SP and the BKSs, disregarding instance C7, was 0.06%. As for the eight instances suggested in [40], ILS-RVND-SP equaled the result of one instance and improved the results of the remaining ones. The average gap between the average solutions produced by ILS-RVND-SP and the BKSs was −0.08%.

It is worth mentioning that on the last set of instances, the ILS-RVND algorithm itself is sufficiently capable of obtaining, on average, competitive results, in terms of computational time and solution quality (see [15]), when compared to the best results reported in [23]. Nevertheless, the solutions obtained by ILS-RVND-SP are still much better than both of them, despite the larger computational time.

4.4. VRPSPD

Table 7 contains the results obtained in the set of instances suggested in [41] and a comparison with those reported in [27,42,24]. It can be verified that the ILS-RVND-SP equaled 21 BKSs and improved another five. Table 8 presents the results found on the instances proposed in [43] and also those reported in [24–26].

Table 9
Results found for the VRPMPD instances proposed in [41].

Instance	n	v	BKS	Røpke and Pisinger		Gajpal and Abad		ILS-RVND-SP				
				Best Sol.	Time[a] (s)	Best Sol.	Time[b] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
CMT1H	50	4	**465.02**[c]	**465**	51	**465.02**	5.6	**465.02**	465.03	0.00	0.00	2.07
CMT1Q	50	6	**489.74**[c]	**490**	41	**489.74**	6.0	**489.74**	489.74	0.00	0.00	1.52
CMT1T	50	7	**520.06**[c]	**520**	34	**520.06**	7.0	**520.06**	**520.06**	0.00	0.00	1.60
CMT2H	75	5	**662.63**	**663**	78	**662.63**	22.0	**662.63**	**662.63**	0.00	0.00	5.16
CMT2Q	75	7	**732.76**	**733**	65	**732.76**	26.2	**731.26**	**731.40**	−0.20	−0.19	8.03
CMT2T	75	9	**782.77**[c]	**783**	57	**782.77**	26.0	**782.77**	**782.77**	0.00	0.00	7.56
CMT3H	100	3	**700.94**[c]	**701**	186	**701.31**	35.6	**700.94**	**700.94**	0.00	0.00	17.65
CMT3Q	100	4	**747.15**[c]	**747**	128	**747.15**	39.8	**747.15**	**747.15**	0.00	0.00	9.70
CMT3T	100	5	**798.07**[c]	**798**	109	**798.07**	42.6	**798.07**	**798.07**	0.00	0.00	28.76
CMT12H	100	6	**629.37**[c]	**629**	150	**629.37**	32.8	**629.37**	**629.37**	0.00	0.00	13.93
CMT12Q	100	8	**729.25**[c]	**729**	108	**729.46**	42.0	**729.25**	**729.25**	−0.03	−0.03	17.37
CMT12T	100	9	**787.52**[c]	**788**	96	**787.52**	52.0	**787.52**	**787.52**	0.00	0.00	6.79
CMT11H	120	4	**818**	**818**	303	820.35	45.8	**818.05**	818.06	0.01	0.01	63.18
CMT11Q	120	6	**939.36**[c]	**939**	196	**939.36**	66.2	**939.36**	**939.36**	0.00	0.00	20.35
CMT11T	120	7	**998.80**	1000	164	**998.80**	70.2	**998.80**	998.81	0.00	0.00	19.91
CMT4H	150	6	**829**	**829**	345	831.39	125.4	**828.12**	831.59	−0.11	0.31	80.24
CMT4Q	150	9	**913.93**	918	244	**913.93**	153.0	915.27	915.27	0.15	0.15	58.92
CMT4T	150	11	**990.39**	1000	212	**990.39**	166.8	**990.39**	**990.39**	0.00	0.00	50.42
CMT5H	200	9	**992.37**	983	514	**992.37**	351.4	**978.74**	**978.74**	−1.37	−1.37	1531.73
CMT5Q	200	12	**1118**[d]	1119	381	1134.72	451.8	**1104.87**	**1105.79**	−1.17	−1.09	1627.78
CMT5T	200	15	**1227**	**1227**	333	1232.08	460.8	**1218.77**	**1220.24**	−0.67	−0.55	1802.81
CMT6H	50	7	**555.43**	**555**	31	**555.43**	13.0	**555.43**	557.35	0.00	0.35	1.08
CMT6Q	50	7	**555.43**	**555**	30	**555.43**	12.8	**555.43**	557.15	0.00	0.31	1.08
CMT6T	50	7	**555.43**	**555**	31	**555.43**	11.6	**555.43**	556.64	0.00	0.22	1.15
CMT7H	75	13	**900**	**900**	54	900.84	50.0	**900.54**	900.84	0.06	0.09	4.47
CMT7Q	75	14	**900.69**	**901**	53	**900.69**	46.8	**900.69**	902.62	0.00	0.21	4.90
CMT7T	75	14	**903.05**	**903**	52	**903.05**	39.0	**903.05**	**903.05**	0.00	0.00	4.77
CMT8H	100	10	**865.50**	**866**	95	**865.50**	85.6	**865.50**	**865.50**	0.00	0.00	7.78
CMT8Q	100	10	**865.50**	**866**	93	**865.50**	74.4	**865.50**	**865.50**	0.00	0.00	7.50
CMT8T	100	10	**865.54**	**866**	95	**865.54**	65.6	**865.54**	**865.54**	0.00	0.00	7.18
CMT14H	100	11	**821.75**	**822**	89	**821.75**	81.6	**821.75**	**821.75**	0.00	0.00	5.37
CMT14Q	100	11	**821.75**	**822**	85	**821.75**	72.4	**821.75**	**821.75**	0.00	0.00	5.47
CMT14T	100	11	**826.77**	**827**	86	**826.77**	64.6	**826.77**	**826.77**	0.00	0.00	6.30
CMT13H	120	12	**1542.86**	**1543**	125	**1542.86**	164.2	**1542.86**	1544.54	0.00	0.11	73.82
CMT13Q	120	12	**1542.97**	**1543**	120	**1542.97**	157.8	**1542.86**	1544.05	−0.01	0.07	69.87
CMT13T	120	12	**1542.97**	1544	127	**1542.97**	152.8	**1542.86**	1544.11	−0.01	0.07	73.59
CMT9H	150	16	**1161**[d]	1166	177	1161.63	306.4	**1160.68**	1162.17	−0.03	0.10	77.95
CMT9Q	150	16	**1161.51**	1162	171	**1161.51**	289.6	**1161.24**	1161.69	−0.02	0.02	80.64
CMT9T	150	16	**1162.68**	1164	178	**1162.68**	261.0	**1162.55**	1164.37	−0.01	0.15	83.29
CMT10H	199	20	**1383.78**	1393	296	**1383.78**	791.0	**1372.52**	**1377.23**	−0.81	−0.47	550.45
CMT10Q	199	20	**1386.54**	1389	288	**1386.54**	730.2	**1374.18**	**1379.47**	−0.89	−0.51	537.66
CMT10T	199	20	**1395**[d]	1402	291	1400.22	658.6	**1381.04**	**1388.17**	−1.00	−0.49	501.65
									Avg.	−0.15	−0.06	178.13
Scaled time (s)					51.31		51.28					178.13

[a] Average of 10 runs on a Pentium IV 1.5 GHz (1311 Mflop/s).
[b] Best run on a Xeon 2.4 GHz (1978 Mflop/s).
[c] Optimality proved.
[d] Found in [7] using another version of their algorithm.

ILS-RVND-SP found 12 BKSs improved another six results. Furthermore, it is worth noting that the proposed algorithm had a satisfactory performance on the large size instances, always producing, on average, competitive results. The average gap between the average solutions produced by ILS-RVND-SP and the BKSs in the first and second group of instances was 0.12% and -0.07% respectively.

4.5. VRPMPD

Table 9 illustrates the results found by ILS-RVND-SP on the VRPMPD instances introduced in [41] and a comparison with those reported in [7] (6R—normal learning) and [27]. ILS-RVND-SP obtained the BKS in 25 instances and it managed to improve

Table 10
Results found for the old MDVRP instances proposed in [44].

Instance	n	v	$\lvert G\rvert$	BKS	Pisinger and Røpke		Vidal et al.		ILS-RVND-SP				
					Avg. Sol.[a]	Time[b] (s)	Avg. Sol.[a]	Time[c] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
p01	50	4	4	**576.87**[d]	**576.87**	29	**576.87**	13.8	**576.87**	**576.87**	0.00	0.00	2.80
p02	50	2	4	**473.53**[d]	**473.53**	28	**473.53**	12.6	**473.53**	**473.53**	0.00	0.00	2.27
p03	75	3	2	**641.19**[e]	**641.19**	64	**641.19**	25.8	**641.19**	**641.19**	0.00	0.00	7.25
p12	80	5	2	**1318.95**[f]	1319.13	75	**1318.95**	31.2	**1318.95**	**1318.95**	0.00	0.00	6.14
p04	100	8	2	**1001.04**[g]	1006.09	88	1001.23	116.4	**1001.04**	**1001.04**	0.00	0.00	51.76
p05	100	5	2	**750.03**[e]	752.34	120	**750.03**	63.6	**750.03**	750.21	0.00	0.02	31.54
p06	100	6	3	**876.50**[f]	883.01	93	**876.50**	68.4	**876.50**	**876.50**	0.00	0.00	25.70
p07	100	4	4	**881.97**[g]	889.36	88	884.43	93.0	**881.97**	**881.97**	0.00	0.00	21.88
p15	160	5	4	**2505.42**[e]	2519.64	253	**2505.42**	115.2	**2505.42**	**2505.42**	0.00	0.00	48.59
p18	240	5	6	**3702.85**[e]	3736.53	419	**3702.85**	271.2	**3702.85**	**3702.85**	0.00	0.00	1019.76
p21	360	5	9	**5474.84**[e]	5501.58	582	5476.41	600.0	**5474.84**	**5474.84**	0.00	0.00	2544.57
p13	80	5	2	**1318.95**[f]	**1318.95**	60	**1318.95**	34.2	**1318.95**	**1318.95**	0.00	0.00	3.06
p14	80	5	2	**1360.12**[e]	**1360.12**	58	**1360.12**	33.0	**1360.12**	**1360.12**	0.00	0.00	19.11
p16	160	5	4	**2572.23**[f]	2573.95	188	**2572.23**	118.2	**2572.23**	**2572.23**	0.00	0.00	247.77
p17	160	5	4	**2709.09**[e]	**2709.09**	179	**2709.09**	128.4	**2709.09**	2710.21	0.00	0.04	1448.47
p19	240	5	6	**3827.06**[f]	3838.76	315	**3827.06**	252.0	**3827.06**	3827.55	0.00	0.01	1214.57
p20	240	5	6	**4058.07**[e]	4064.76	300	**4058.07**	262.2	**4058.07**	**4058.07**	0.00	0.00	544.80
p08	249	14	2	**4372.78**[h]	4421.03	333	4397.42	600.0	4379.46	4393.70	0.15	0.48	1244.57
p09	249	12	3	**3858.66**[h]	3892.50	361	3868.59	570.0	3859.54	3864.22	0.02	0.14	1431.88
p10	249	8	4	**3631.11**[h]	3666.85	363	3636.08	589.2	3631.37	3634.72	0.01	0.10	1422.66
p11	249	6	5	**3546.06**[g]	3573.23	357	3548.25	428.4	**3546.06**	3546.15	0.00	<0.01	1217.35
p22	360	5	9	**5702.16**[e]	5722.19	462	**5702.16**	600.0	**5702.15**	5705.84	0.00	0.06	846.01
p23	360	5	9	**6078.75**[g]	6092.66	443	**6078.75**	600.0	**6078.75**	**6078.75**	0.00	0.00	1019.15
					Avg. Gap (%)	0.40	Avg. Gap (%)	0.07		Avg.	0.01	0.04	627.03
Scaled time (s)						77.44		82.87					627.03

[a] Average of 10 runs.
[b] Average of 10 runs on a Pentium IV 3.0 GHz (3181 Mflop/s).
[c] Average of 10 runs on an Opteron 2.4 GHz scaled for a Pentium IV 3.0 GHz.
[d] Optimality proved.
[e] First found in [44].
[f] First found in [46].
[g] First found in [6].
[h] First found in [9].

Table 11
Results found for the new MDVRP instances proposed in [44].

Instance	n	v	$\lvert G\rvert$	BKS	Pisinger and Røpke		Vidal et al.		ILS-RVND-SP				
					Avg. Sol.[a]	Time[b] (s)	Avg. Sol.[a]	Time[c] (s)	Best Sol.	Avg. Sol.	Gap (%)	Avg. Gap (%)	Time (s)
pr01	48	2	4	**861.32**[d]	**861.32**	30	**861.32**	10.2	**861.32**	**861.32**	0.00	0.00	1.24
pr07	72	3	6	**1089.56**[d]	1089.56	58	**1089.56**	20.4	**1089.56**	**1089.56**	0.00	0.00	3.87
pr02	96	4	4	**1307.34**[e]	1308.17	103	**1307.34**	45.6	**1307.34**	1308.53	0.00	0.09	12.39
pr03	144	6	4	**1803.80**[f]	1810.66	214	1803.80	114.6	**1803.81**	1804.09	0.00	0.02	55.04
pr08	144	6	6	**1664.85**[e]	1675.74	207	1665.05	123.0	**1664.85**	1665.08	0.00	0.01	393.98
pr04	192	8	4	**2058.31**[f]	2073.16	296	2059.36	313.2	**2058.31**	2060.93	0.00	0.13	779.30
pr09	216	9	6	**2133.20**[f]	2144.84	350	2134.17	366.0	**2133.20**	2135.37	0.00	0.10	1070.41
pr05	240	10	4	**2331.20**[f]	2350.31	372	2340.29	573.6	**2331.20**	2338.12	0.00	0.30	1337.10
pr06	288	12	4	**2676.30**[f]	2695.74	465	2681.93	600.0	2680.77	2685.23	0.17	0.33	2297.66
pr10	288	12	6	**2868.26**[f]	2905.43	455	2886.59	600.0	2874.28	2882.41	0.21	0.49	3009.53
					Avg. Gap (%)	0.52	Avg. Gap (%)	0.13		Avg.	0.04	0.15	896.05
Scaled time (s)						86.38		93.72					896.05

[a] Average of 10 runs.
[b] Average of 10 runs on a Pentium IV 3.0 GHz (3181 Mflop/s).
[c] Average of 10 runs on an Opteron 2.4 GHz scaled for a Pentium IV 3.0 GHz.
[d] First found in [44].
[e] First found in [6].
[f] First found in [9].

the result of another 12. The developed algorithm outperformed both the algorithms suggested in [7,27] in terms of solution quality. The average gap between the average solutions obtained by ILS-RVND-SP and the BKSs was −0.06%.

4.6. MDVRP

Tables 10 and 11 present a comparison, in terms of average solution, between the results found by ILS-RVND-SP and those determined in [6] (ALNS 50K) and [9] on the old and new set of instances presented in [44], respectively. The latter two clearly outperformed the first one in terms of solution quality. The average gap between the average solutions found by ILS-RVND-SP and the BKSs for the old and new benchmark sets was, respectively, 0.04% and 0.15%.

4.7. MDVRPMPD

Table 12 presents the results found by ILS-RVND-SP and those pointed out in [7] (6R—no learning) on the set of instances proposed in [41]. With respect to the solution quality, the developed algorithm clearly had a better performance, equaling 17 BKSs and improving the result of another 16. The average gap between the average solutions and the BKSs was −0.10%.

5. Concluding remarks

This work presented an algorithm that hybridizes an Iterated Local Search based heuristic and a Set Partitioning formulation. Its design favored the flexibility, allowing its application in the solution of several VRP variants. Moreover, we believe that the developed hybrid approach is relatively simple and easy to implement. The key aspects of the proposed methodology are the interaction between a solver and a metaheuristic approach while solving a given MIP model and an efficient scheme of dynamically controlling the size of the SP models when solving large size instances. These ideas can be employed to efficiently solve a large class of combinatorial optimization problems.

The ILS-RVND-SP algorithm was evaluated in hundreds of well-known instances of the variants considered in this work, with up to 480 customers. The same parameter tuning was adopted and the results obtained were quite competitive with those found by heuristics devoted to specific variants. Table 13 shows the summary of the results found by ILS-RNVD-SP. In this table *Avg. Gap* corresponds to the average gap between the average solutions and the BKSs, *#Instances* is the number of instances of a particular benchmark, *#Improv* denotes the number of solutions improved and *#Ties* represents the number of ties. It can be seen that 52 new best solutions were found and that the Avg. Gap was always smaller than 0.55%.

Table 12
Results found for the MDVRPMPD instances proposed in [41].

Instance	n	v	$\lvert G\rvert$	BKS	Røpke and Pisinger			ILS-RVND-SP				
					Best Sol.	Avg. Sol.[a]	Time[b] (s)	Best Sol.	Avg. Sol.	Gap %	Avg. Gap (%)	Time (s)
GJ01Q	50	4	4	**528**	**528**	**528**	36	**528.30**	**528.30**	0.06	0.06	3.12
GJ01T	50	4	4	**569**	**569**	**569**	34	**569.43**	**569.43**	0.08	0.08	2.98
GJ02H	75	4	2	**440**	**440**	**440**	51	**440.00**	**440.00**	0.00	0.00	2.95
GJ02Q	75	4	2	**450**	**450**	451	43	**449.72**	**449.72**	−0.06	−0.06	2.60
GJ02T	75	4	2	**464**	**464**	**464**	37	**464.13**	**464.13**	0.03	0.03	2.50
GJ03H	100	5	3	**581**	581	583	81	**579.45**	**579.45**	−0.27	−0.27	9.54
GJ03Q	100	5	3	**605**	**605**	608	71	**605.25**	**605.25**	0.04	0.04	8.95
GJ03T	100	5	3	**624**	**624**	626	65	**624.44**	**624.44**	0.07	0.07	10.94
GJ04H	100	2	8	**790**	**790**	797	112	**789.19**	**789.30**	−0.10	−0.09	31.69
GJ04Q	100	2	8	**875**	**875**	876	94	**874.78**	874.79	−0.03	−0.02	41.89
GJ04T	100	2	8	**962**	**962**	969	85	**962.25**	**962.65**	0.03	0.07	37.95
GJ05H	100	2	5	**678**	**678**	680	168	**676.81**	**676.91**	−0.18	−0.16	22.24
GJ05Q	100	2	5	**700**	702	705	133	**700.15**	**700.15**	0.02	0.02	19.55
GJ05T	100	2	5	**733**	**733**	738	118	**733.17**	733.18	0.02	0.02	39.31
GJ06H	100	3	6	**745**	747	751	116	**742.18**	**742.18**	−0.38	−0.38	32.28
GJ06Q	100	3	6	**794**	**794**	800	100	**793.85**	793.87	−0.02	−0.02	25.00
GJ06T	100	3	6	**851**	**851**	853	90	**850.82**	**850.82**	−0.02	−0.02	27.19
GJ07H	100	4	4	**733**	**733**	734	117	**732.73**	**732.73**	−0.04	−0.04	24.66
GJ07Q	100	4	4	**802**	803	807	94	**801.91**	801.94	−0.01	−0.01	38.58
GJ07T	100	4	4	**854**	855	862	88	**853.54**	**853.54**	−0.05	−0.05	20.64
GJ08H	249	2	14	**3327**	**3327**	3373	581	**3320.39**	3342.91	−0.20	0.48	1435.21
GJ08Q	249	2	14	**3762**	3774	3810	479	**3745.18**	3769.01	−0.45	0.19	1288.57
GJ08T	249	2	14	**4134**	4134	4170	431	**4110.78**	**4120.27**	−0.56	−0.33	1272.63
GJ09H	249	3	12	**3005**	3006	3028	646	**2990.92**	3005.52	−0.47	0.02	1478.35
GJ09Q	249	3	12	**3355**	**3355**	3393	535	**3351.18**	3361.23	−0.11	0.19	1362.18
GJ09T	249	3	12	**3677**	**3677**	3718	492	**3656.03**	**3661.62**	−0.57	−0.42	1316.84
GJ010H	249	4	8	**2927**	2930	2963	644	**2894.71**	**2905.23**	−1.10	−0.74	1452.52
GJ010Q	249	4	8	**3242**	3245	3267	513	**3220.79**	**3226.79**	−0.65	−0.47	1315.68
GJ010T	249	4	8	**3485**	**3485**	3524	472	**3470.70**	**3477.99**	−0.41	−0.20	1281.92
GJ011H	249	5	6	**2855**	2880	2905	609	**2842.79**	**2845.71**	−0.43	−0.33	1357.58
GJ011Q	249	5	6	**3155**	3165	3192	511	**3138.64**	**3143.33**	−0.52	−0.37	1267.12
GJ011T	249	5	6	**3390**	**3390**	3421	469	**3360.48**	**3367.63**	−0.87	−0.66	1181.56
					Avg. Gap (%)		0.66		Avg.	−0.22	−0.10	497.54
Scaled time (s)						55.48						497.54

Found in [7] using another version of their algorithm.

[a] Average of 10 runs.

[b] Average of 10 runs on a Pentium IV 1.5 GHz (1311 Mflop/s).

Table 13
Summary of ILS-RVND-SP results.

Variant	#Instances	#Improv	#Ties	Avg. Gap (%)	Avg. Time (s)
CVRP[a]	3[b], 14[c], 20[d]	1[b], 0[c], 0[d]	2[b], 13[c], 5[d]	0.17[b], 0.08[c], 0.55[d]	17.41[b], 100.83[c], 3938.23[d]
ACVRP[a]	24[e]	1[e]	23[e]	0.01[e]	2.24[e]
OVRP[a]	16[f],8[g]	3[f], 7[g]	12[f], 1[g]	0.06[f], −0.08[g]	143.44[f], 3844.03[g]
VRPSPD[a]	28[h], 18[i]	5[h], 7[i]	21[h], 11[i]	0.12[h], −0.07[i]	189.24[h], 3653.44[i]
VRPMPD[a]	42[h]	12[h]	29[h]	−0.06[h]	178.13[h]
MDVRP[a]	23[j], 10[k]	0[j], 0[k]	20[j], 8[k]	0.04[j], 0.15[k]	627.03[j], 896.05[k]
MDVRPMPD[a]	33[h]	16[h]	17[h]	−0.10[h]	497.54[h]
Total	239	52	162		

[a] Core i7 2.93 GHz (single thread).
[b] M-series open instances.
[c] Christofides et al. [36].
[d] Golden et al. [37].
[e] Fischetti et al. [19] and Pessoa et al. [20].
[f] Christofides et al. [36] and Fisher [39].
[g] Li et al. [40].
[h] Salhi and Nagy [41].
[i] Montané and Galvão [43].
[j] Cordeau et al. (old) [44].
[k] Cordeau et al. (new) [44].

As for future work, we intend to extend the range of application of ILS-RVND-SP by tackling variants with additional constraints such as time windows, backhauls and site/time dependence.

Acknowledgments

This research was partially supported by the following Brazilian research agencies: CNPq, CAPES and FAPERJ.

References

[1] Garey M, Johnson D. Computers and intractability: a guide to the theory of NP-completeness. WH Freeman & Co.; 1979.
[2] Maniezzo V, Stützle T, Voß S. Matheuristicshybridizing metaheuristics and mathematical programming. Annals of information systems. New York: Springer; 2009.
[3] De Franceschi R, Fischetti M, Toth P. A new ILP-based refinement heuristic for vehicle routing problems. Mathematical Programming 2006;105(2–3):471–99.
[4] Alvarenga G, Mateus G, De Tomi G. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. Computers & Operations Research 2007;34(6):1561–84.
[5] Toth P, Tramontani A. An integer linear programming local search for capacitated vehicle routing problems. In: The vehicle routing problem: latest advances and new challenges. Springer; 2008. p. 275–95.
[6] Pisinger D, Røpke S. A general heuristic for vehicle routing problems. Computers & Operations Research 2007;34(8):2403–35.
[7] Røpke S, Pisinger D. A unified heuristic for a large class of vehicle routing problems with backhauls. European Journal of Operational Research 2006;171(3):750–75.
[8] Cordeau J-F, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society 2011;52:928–36.
[9] Vidal T, Crainic TG, Gendreau M, Lahrichi N, Rei W. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. Operations Research 2012;60(3):611–24.
[10] Cordeau J-F, Gendreau M, Laporte G, Potvin J-Y, Semet F. A guide to vehicle routing problem. Journal of the Operational Research Society 2002;53:512–22.
[11] Laporte G, Semet F. Classical heuristics for the capacitated VRP. In: The vehicle routing problem, monographs on discrete mathematics and applications. Philadelphia: SIAM; 2002. p. 109–28.
[12] Kelly JP, Xu J. A set-partitioning-based heuristic for the vehicle routing problem. INFORMS Journal on Computing 1999;11(2):161–72.
[13] Rochat Y, Taillard RD. Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1995;1:147–67.
[14] Subramanian A, Penna PHV, Uchoa E, Ochi LS. A hybrid algorithm for the heterogenous fleet vehicle routing problem. European Journal of Operational Research 2012;221:285–95.
[15] Subramanian A. Heuristic, exact and hybrid approaches for vehicle routing problems. PhD thesis. Universidade Federal Fluminense, Niterói, Rio de Janeiro, Brazil, available at 〈www.ic.uff.br/PosGraduacao/Teses/532.pdf〉; 2012.
[16] Mester D, Bräysy O. Active-guided evolution strategies for large-scale capacitated vehicle routing problems. Computers & Operations Research 2007;34(10):2964–75.
[17] Nagata Y, Bräysy O. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. Networks 2009;54(4):205–15.
[18] Zachariadis EE, Kiranoudis CT. A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. Computers & Operations Research 2010;37(12):2089–105.
[19] Fischetti M, Toth P, Vigo D. A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. Operations Research 1994;42(5):846–59.
[20] Pessoa AA, Uchoa E, Poggi de Aragão, M. Robust Branch-and-Cut-and-Price Algorithms for Vehicle Routing Problems. In: The vehicle routing problem: latest advances and new challenges. Springer; 2008. p. 297–325.
[21] Fleszar K, Osman IH, Hindi KS. A variable neighbourhood search algorithm for the open vehicle routing problem. European Journal of Operational Research 2009;195(3):803–9.
[22] Repoussis PP, Tarantilis CD, Bräysy O, Ioannou G. A hybrid evolution strategy for the open vehicle routing problem. Computers & Operations Research 2010;37(3):443–55.
[23] Zachariadis EE, Kiranoudis CT. An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. Computers & Operations Research 2010;37(4):712–23.
[24] Subramanian A, Drummond LMA, Bentes C, Ochi LS, Farias R. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. Computers & Operations Research 2010;37(11):1899–911.
[25] Zachariadis EE, Kiranoudis CT. A local search metaheuristic algorithm for the vehicle routing problem with simultaneous pick-ups and deliveries. Expert Systems with Applications 2011;38(3):2717–26.
[26] Souza MJF, Silva MSA, Mine MT, Ochi LS, Subramanian A. A hybrid heuristic, based on iterated local search and GENIUS, for the vehicle routing problem with simultaneous pickup and delivery. International Journal of Logistics Systems Management 2010;10(2):142–56.
[27] Gajpal Y, Abad P. An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup. Computers & Operations Research 2009;36(12):3215–23.
[28] Penna PHV, Subramanian A, Ochi LS. An iterated local search heuristic for the heterogeneous fleet vehicle routing problem, Journal of Heuristics, http://dx.doi.org/10.1007/s10732-011-9186-y, in press.
[29] Silva MM, Subramanian A, Vidal T, Ochi LS. A simple and effective metaheuristic for the minimum latency problem. European Journal of Operational Research 2012;221:513–20.
[30] Subramanian A, Battarra M. An iterated local search algorithm for the travelling salesman problem with pickups and deliveries. Journal of the Operational Research Society 2013;64:402–9.
[31] Osman IH. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Annals of Operations Research 1993;41(1–4):421–51.
[32] Taillard E, Badeau P, Gendreau M, Guertin F, Potvin J-Y. A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 1997;31:170–86.
[33] Or I. Traveling salesman-type combinational problems and their relation to the logistics of blood banking. PhD thesis. USA: Northwestern University; 1976.
[34] Croes GA. A method for solving traveling salesman problems. Operations Research 1958;6:791–812.
[35] Dongarra JJ. Performance of various computers using standard linear software. Technical Report CS-89-85. Computer Science Department, University of Tennessee; 2010.

[36] Christofides N, Mingozzi A, Toth P. The vehicle routing problem. In: Combinatorial optimization. Chichester, UK: Wiley; 1979. p. 315–38.
[37] Golden BL, Wasil EA, Kelly JP, Chao I-M. Metaheuristics in vehicle routing. In: Fleet management and logistics. Boston: Kluwer; 1998. p. 33–56.
[38] Fukasawa R, Longo H, Lysgaard J, Poggi de Aragão M, Reis M, Uchoa E, et al. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. Mathematical Programming 2006;106:491–511.
[39] Fisher M. Optimal solutions of vehicle routing problems using minimum K-trees. Operations Research 1994;42:626–42.
[40] Li F, Golden B, Wasil E. The open vehicle routing problem: algorithms, large-scale test problems, and computational results. Computers & Operations Research 2007;34(10):2918–30.
[41] Salhi S, Nagy G. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. Journal of the Operational Research Society 1999;50:1034–42.
[42] Zachariadis EE, Tarantilis CD, Kiranoudis CT. An adaptive memory methodology for the vehicle routing problem with simultaneous pick-ups and deliveries. European Journal of Operational Research 2010;202(2):401–11.
[43] Montané FAT, Galvão RD. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Computers & Operations Research 2006;33(3):595–619.
[44] Cordeau J-F, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 1997;30(2):105–19.
[45] Li F, Golden B, Wasil E. A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. Computers and Operations Research 2007;34:2734–42.
[46] Renaud J, Laporte G, Boctor FF. A tabu search heuristic for the multi-depot vehicle routing problem. Computers & Operations Research 1996;23(3):229–35.