A hybrid heuristic, based on Iterated Local Search and GENIUS, for the Vehicle Routing Problem with Simultaneous Pickup and Delivery

Marcone Jamilson Freitas Souza*

Universidade Federal de Ouro Preto,
Campus Universitário,
Morro do Cruzeiro 35.400-000 Ouro Preto,
Minas Gerais, Brazil
E-mail: marcone.freitas@gmail.com
E-mail: marcone@iceb.ufop.br
*Corresponding author

Marcio Tadayuki Mine,
Matheus de Souza Alves Silva,
Luiz Satoru Ochi and Anand Subramanian

Universidade Federal Fluminense,
Rua Passo da Pátria,
156 – Bloco E – 3^0 andar,
24.210-240 Niterói, Rio de Janeiro, Brazil
E-mail: mmine@ic.uff.br
E-mail: msalves@ic.uff.br
E-mail: satoru@ic.uff.br
E-mail: anand@ic.uff.br

Abstract: This work deals with the Vehicle Routing Problem with Simultaneous Pickup and Delivery (VRPSPD). The VRPSPD is a common problem in the area of reverse logistics, which aims to plan the transportation of products to customers, as well as the return of leavings or products used by them for recycling or to special depots. The VRPSPD is NP-hard, since it can be reduced to the classical Vehicle Routing Problem (VRP) when no client needs the pickup service. To solve it, we propose a hybrid heuristic algorithm, called GENILS, based on Iterated Local Search (ILS), Variable Neighbourhood Descent and GENIUS. The proposed algorithm was tested on three well-known sets of instances found in literature and it was competitive with the best existing approaches. Among the 72 test-problems of these sets, the GENILS was capable to improve the result of nine instances and to equal another 49.

Keywords: VRPSPD; vehicle routing problem with simultaneous pickup and delivery; metaheuristics; ILS; iterated local search; GENIUS; variable neighbourhood descent; reverse logistics.

Reference to this paper should be made as follows: Souza, M.J.F., Mine, M.T., Silva, M.S.A., Ochi, L.S. and Subramanian, A. (20111) 'A hybrid heuristic, based on Iterated Local Search and GENIUS, for the Vehicle Routing Problem with Simultaneous Pickup and Delivery', *Int. J. Logistics Systems and Management*, Vol. 10, No. 2, pp.142–157.

Biographical notes: Marcone Jamilson Freitas Souza received his PhD in Systems Engineering in 2000 from the Universidade Federal do Rio de Janeiro, Brazil. He is an Associate Professor in the Departamento de Ciência da Computação, Universidade Federal de Ouro Preto, Brazil. His research interests includes vehicle routing, public transport, scheduling, open-pit-mining, mathematical programming and metaheuristics.

Marcio Tadayuki Mine received his MSc in Computing in 2009 from the Universidade Federal Fluminense, Brazil. He is currently an Optimisation Analyst at GAPSO, Brazil, a company that develops decision support systems. His research interests includes vehicle routing, linear programming and metaheuristics.

Matheus de Souza Alves Silva received his MSc in Computing in 2009 from the Universidade Federal Fluminense, Brazil. He is currently an Optimisation Analyst at GAPSO, Brazil. His research interests includes combinatorial optimisation, metaheuristics, travelling salesman and covering tour problems.

Luiz Satoru Ochi received his PhD in Systems Engineering in 1989 from the Universidade Federal do Rio de Janeiro, Brazil. He is a Full Professor in the Instituto de Computação, Universidade Federal Fluminense. His research interests includes combinatorial optimisation, metaheuristics, parallel algorithms, data mining and vehicle routing.

Anand Subramanian received his MSc in Production Engineering in 2008 from the Universidade Federal da Paraíba, Brazil. He is a PhD student in the Instituto de Computação, Universidade Federal Fluminense. His research interests includes combinatorial optimisation, metaheuristics, integer programming, hybrid algorithms and vehicle routing.

An earlier version of this paper was presented at the International Conference on Industrial Logistics (ICIL 2010) hosted by Military Institute of Engineering with cooperation of Federal University of Espírito Santo (UFES) and COPPE/Federal University of Rio de Janeiro (UFRJ) between March 8th and 10th 2010.

1 Introduction

The Vehicle Routing Problem (VRP) was originally proposed by Dantzig and Ramser (1959) and it can be defined as follows. Given a set of N clients, each one with a demand d_i, and a homogeneous vehicle fleet with capacity Q, the objective is to design the vehicle routes in such a way that the clients' demands are completely attended in a single visit and the sum of the travelled costs are minimised.

In the late 1980s, Min (1989) proposed an important variant of the VRP: the Vehicle Routing Problem with Simultaneous Pickup and Delivery (VRPSPD), in which the pickup and delivery services must be performed simultaneously. This model is a basic problem in the field of Reverse Logistics. Its main goal is to manage the transportation of products to customers, as well as the return of residues or products utilised by these clients for recycling or specialised depots. The Reverse Logistic can be observed, for instance, in the postal logistics or in the distribution planning of the beverages industry.

The VRPSPD is a NP-hard problem since it can be reduced to the classical VRP when all clients do not need pickup services. Therefore, heuristic approaches have been frequently applied to solve the problem.

Min (1989) proposed a three phase method for solving the distribution planning of a public library. The first phase consists in grouping the clients in clusters by means of the average linkage method (Anderberg, 2007). The second phase assigns the vehicles to the respective routes. The third phase consists in solving each cluster by applying a Travelling Salesman Problem (TSP) heuristic that attributes, iteratively, a penalty to the arcs in which the vehicle capacity is exceeded in order to generate a feasible solution.

Halse (1992) proposed a two phase heuristic for solving the VRPSPD which firstly consists of assigning the clients to the vehicles followed by an application of a 3-opt improvement heuristic.

Dethloff (2001) developed a Cheapest Insertion based method in which the clients are added to the routes according to the following criteria:

- distance
- residual capacity
- clients' distance to the depot.

Vural (2003) developed two Genetic Algorithm (GA) approaches. The first one codifies the individuals using the Random Keys method while the second one was implemented as an improvement heuristic based on the GA structure developed by Topcuoglu and Sevilmis (2002).

Gökçe (2004) employed an Ant Colony approach that uses a 2-opt local search procedure in the pos-optimisation phase.

Nagy and Salhi (2005) developed an approach which combines different strategies used to solve the classical VRP such as: 2-opt, 3-opt, shift, swap, reverse and procedures to repair' infeasible solutions.

Dell'Amico et al. (2006) utilised a branch-and-price technique by employing two approaches: dynamic programming and state space relaxation.

Crispim and Brandão (2005) proposed a hybrid approach by combining the Tabu Search (TS) and Variable Neighbourhood Descent (VND) metaheuristics. The sweep method was used to generate an initial solution while shift and swap movements were utilised in the local search phase.

Röpke and Pisinger (2006) developed a heuristic inspired on the Large Neighbourhood Search (LNS) approach. The LNS is a local search based on two ideas for defining and exploring the neighbourhood structures of large complexity.

The first idea is to fix one part of the solution and hence define the solution space. The second one performs a search by applying constraint programming, integer programming and others.

Montané and Galvão (2006) utilised the TS metaheuristic considering four neighbourhood structures: shift, swap, cross and 2-opt. Both the first improvement and the best improvement approaches were adopted.

Chen (2006) dealt with the VRPSPD by combining the Simulated Annealling (SA) and TS metaheuristics while Chen and Wu (2006) developed a methodology based on the record-to-record travel approach, which in turn is a variation of the SA.

Constructive and improvement heuristics, as well as an algorithm based on the TS metaheuristic were presented by Bianchessi and Righini (2007). These approaches employed the node-exchange-based and arc-exchange-based movements.

Wassan et al. (2008) proposed a reactive version of the TS metaheuristic. The sweep method was used to generate a initial solution while shift, swap and reverse movements were utilised in the local search phase.

Subramanian et al. (2008) developed an algorithm based on the Iterated Local Search (ILS) approach, which uses the VND procedure in the local search. A constructive heuristic inspired on the algorithm proposed by Dethloff (2001) was used to generate an initial solution. The VND explores the solution space by employing shifts, swap and cross movements. In case of improvement of the current solution an intensification is performed in the modified routes by means of the Or-opt, 2-opt, exchange and reverse movements. The perturbation mechanisms applied were ejection chain, double-swap and double-bridge. The ejection chain consists in transferring a client from one route to an adjacent one. The double-swap consists in two consecutive swap movements. The double-bridge consists in removing four arcs and inserting another four in such away that a new route is generated. A detailed description of this algorithm, as well as a new mathematical formulation for the VRPSPD can be found in Subramanian (2008).

Zachariadis et al. (2009) proposed a hybrid heuristic for the VRPSPD which combines the TS and Guided Local Search metaheuristics.

In order to compare the literature approach, Dethloff (2001) proposed a set of 40 instances with 50 clients. Salhi and Nagy (1999) presented 28 instances with 50–199 clients, but half of them consider route duration constraints. Finally, Montané and Galvão (2006) proposed 18 instances involving 100, 200 and 400 clients.

To the best of our knowledge, the best results found in the literature for these instances belong to:

- Chen and Wu (2006): one instance of Salhi and Nagy (1999)
- Röpke and Pisinger (2006): 26 instances of Dethloff (2001)
- Wassan et al. (2008): 6 instances of Salhi and Nagy (1999)
- Zachariadis et al. (2009): 6 instances of Salhi and Nagy (1999) and 27 instances of Dethloff (2001)
- Subramanian et al. (2008): all instances of Dethloff (2001), Montané and Galvão (2006) and 17 of Salhi and Nagy (1999).

This work presents a new heuristic algorithm to solve the VRPSPD. The proposed algorithm, called GENILS, combines the ILS, VND and an adaptation of the GENIUS heuristic. It differs from the one developed by Subramanian et al. (2008) because the GENILS includes the GENIUS heuristic and the 3-opt and 4-opt procedures. The results obtained show that these strategies appeared to be efficient in the resolution of the problem.

The remainder of the paper is organised as follows. Section 2 describes the VRPSPD. Section 3 deals with the algorithm GENILS. Section 4 contains the results obtained by the proposed algorithm. Section 5 presents the concluding remarks and future works.

2 Problem description

The VRPSPD is a variant of the classical VRP. In this problem there is a depot with a homogeneous fleet with capacity Q and a set of N clients geographically dispersed. Each client $i \in N$ is associated with quantities d_i and p_i which represent, respectively, the delivery and pickup demands. The objective is to design a set of routes in such a way that sum of the travel costs are minimised and the following constraints are satisfied:

- each route must start and end at the depot
- all clients must be visited exactly once and by only one vehicle
- the delivery and pickup demands must be fully attended
- the vehicle load cannot be exceeded.

In some variants of VRPSPD, one should include travel duration limits (distance or time) to accomplish a route.

Figure 1 illustrates an example of the VRPSPD. In this figure, the clients are represented by integer numbers within the interval [1, $|N|$] while the depot is represented by 0. Each pair $[d_i/p_i]$ denotes, respectively, the delivery and pickup demands of a client i. In addition, there are three routes to be executed by vehicles with capacity $Q = 150$. In one of them, the vehicle leaves the depot and visits the clients 10, 8, 19, 9 and 2, returning to the depot at the end. In the first visit, the vehicle delivers 10 units and collects another 5 units, while in the last visit the vehicle delivers 13 units and collects another 30.

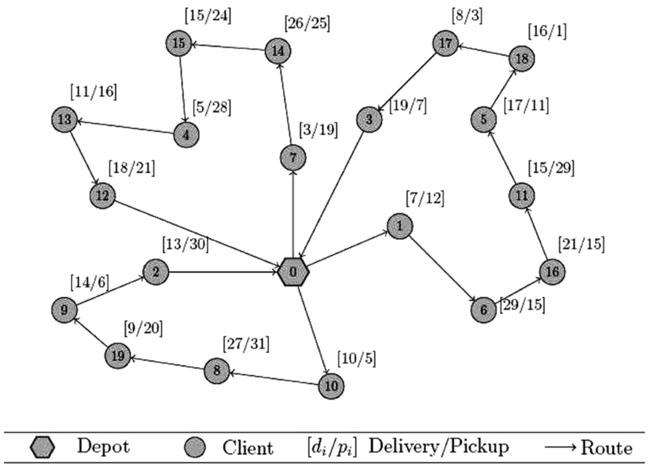
3 Methodology

This section presents the methodology developed to solve the VRPSPD. Section 3.1 describes how a initial solution is generated. Section 3.2 illustrates the neighbourhood structures utilised to explore the solution space. Section 3.3 shows how a solution is evaluated. Section 3.4 presents the GENILS algorithm.

3.1 Constructive heuristic

Three insertion based heuristics are employed to generate an initial solution.

Figure 1 VRPSPD example



The first one, called CI-1R, is an adaptation of the cheapest insertion and the solution is built route by route.

The second one, called CI-NR, was proposed by Subramanian et al. (2008) and it is based on the insertion heuristic of Dethloff (2001).

The last one, VRGENIUS, is an adaptation of the GENIUS heuristic (Gendreau et al., 1992) originally proposed for the TSP and it is divided into two steps: construction (VRGENI) and improvement (VRUS). The VRGENI is a generalised insertion based method whose main characteristic is that the evaluation of the possible insertions of a client i is not necessarily limited to a position between two consecutive clients. The VRUS consists, at each iteration, in removing one client from the solution and re-inserting it in another position with a view of improving the current solution. This procedure ends when there is no more possibility of improvements. It is important to mention that both removal and insertion of a client is performed by applying 2-opt and 3-opt moves. The efficiency of these procedures relies on the fact that the solution space to be explored is restricted to the number of neighbours of each client. This number is determined by a parameter p.

3.2 Neighbourhood structures

In order to explore the solution space of the problem, the following seven neighbourhood operators are applied:

- Shift(1,0): one client i is transferred from a route r_1 to a route r_2
- Shift(2,0): two consecutive clients i and j are transferred from a route r_1 to a route r_2
- Swap(1,1): one client i from a route r_1 is permuted with a client j from a route r_2
- Swap(2,1): two consecutive clients i and j from a route r_1 are permuted with a client k from a route r_2

- Swap(2,2): two consecutive clients i and j from a route r_1 are permuted with another two consecutive clients k and l from a route r_2
- M2-opt: two non-adjacent arcs are removed and another two are inserted in such a way that a new route is formed
- kOr-opt: k consecutive clients are removed and re-inserted in another position of the route. This movement is a generalisation of the Or-opt movement proposed by Or (1976).

It is important to emphasise that only feasible movements are realised.

3.3 Evaluation function

A solution s is evaluated by the function f presented in equation (1), which determines the total travelled cost.

$$f(s) = \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

where

A: set of arcs (i, j), $i, j \in N$
c_{ij}: travelled cost between i and j, $i, j \in N$
x_{ij}: indicates if the arc $(i, j) \in A$ is used $(x_{ij} = 1)$ in the solution or not $(x_{ij} = 0)$.

3.4 GENILS algorithm

A hybrid algorithm, called GENILS, is proposed to solve the VRPSPD. This algorithm uses the method described in Section 3.1 for generating an initial solution and combines the Iterated Local Search – ILS (Stützle and Hoos, 1999), Variable Neighbourhood Descent – VND (Hansen and Mladenović, 2001) and an adaptation of the GENIUS heuristic (Gendreau et al., 1992). The pseudocode of the GENILS is presented in Algorithm 1.

The GENILS algorithm starts by generating three initial solutions, s^A, s^B, and s^C, each one of them by applying the methods described in Section 3.1. These solutions are improved by the VND and the best solution is used as an initial solution s. In order to scape from the local optimum s, the solution is perturbed and a new solution s' is generated. Next, this perturbed solution is improved by the VND local search and a local optimum s'' is obtained. This perturbed solution becomes the new current solution in case s'' is better then s; otherwise, it is discarded and a new perturbation is performed from the solution s. This procedure is repeated until the maximum number of iterations without improvement of the current solution ($iter_{\max}$) is archived.

The perturbations are performed by one of the three following mechanisms chosen at random:

- *Multiple shifts*. Consists in performing k Shift movements (described in Section 3.2) successively. The value of k is randomly chosen among 1, 2 or 3.

Algorithm 1: GENILS

```
Input: iter_max
Output: Solution s
s^A ← generate a solution by applying the CI-IR method
s^B ← generate a solution by applying the CI-NR method
s^C ← generate a solution by applying the VRGENIUS method
s^A ← VND(s^A)
s^B ← VND(s^B)
s^C ← VND(s^C)
s ← s' | f(s') = min{f(s^A), f(s^B), f(s^C)}
iter ← 0
while iter ≤ iter_max do
    iter ← iter + 1
    s' ← perturb(s)
    s'' ← VND(s')
    if f(s'') < f(s) then
        s ← s''
        iter ← 0
    end
end
return s
```

- *Multiple swaps*. It follows the same idea of the previous perturbation, but Swap movements are applied.
- *Ejection chain*. This movement was proposed by Rego and Roucairol (1996). First, a subset composed by m routes $R = \{r_1, r_2, \ldots, r_m\}$ is arbitrarily chosen. Next, a client is transferred from the route r_1 to the route r_2, then another client is transferred from a route r_2 to the route r_3 and so on until a client is transferred from the route r_m to the route r_1. In this perturbation, the clients are randomly selected.

The VND with random neighbourhood ordering explores the solution space by means of the movements described in Section 3.2. The pseudocode of VND is presented in Algorithm 2.

As can be seen in Algorithm 1, an intensification phase is also embedded into to the VND procedure and it is performed using intra-route movements based on the following neighbourhood structures: Shift(1,0), Shift(2,0), Swap(1,1), 2-Opt, Swap(2,1), Swap(2,2) and kOr-opt with $k = 3, 4, 5$.

In addition, this intensification phase also includes another two local search procedures, called G3-opt e G4-opt, which are based on the GENIUS heuristic. These movements are adaptations of the 3-opt and 4-opt neighbourhoods. The adaptation consists in evaluating an arc insertion $(v_i; v_j)$ only if the clients v_i and v_j are relatively close. In view of this, define $N_p(v)$ as the set of p neighbourhoods closest to the client v in a given route r of the solution s, where p is a parameter. Consider also the following definitions: N^r, set of all clients in route r; v_i: client $v_i \in N^r$; v_{h+1} and v_{h-1}: clients in route r which, respectively, succeeds and precedes the client $v_h \in N^r$; v_j: client $v_j \in N^p(v_i)$; v_k: client $v_k \in N_p(v_{i+1})$ in the path between v_j to v_i; v_l: client $v_l \in N^p(v_{j+1})$ in the path between v_i to v_j. The G3-opt works as follows: at each iteration, the arcs $(v_i; v_{i+1})$, $(v_j; v_{j+1})$ and $(v_k; v_{k+1})$ are removed and the arcs $(v_i; v_j)$, $(v_{i+1}; v_k)$ and $(v_{j+1}; v_{k+1})$ are inserted in the route r, in such a way that the solution s is improved and its cost is the least

Algorithm 2: VND

```
Input: t distinct neighborhoods described in section 3.2, Initial solution s
Output: Refined solution s
RN ← set of t neighborhoods, in a random ordering
i ← 1
while i ≤ t do
    Let s0 the best neighbor in RN^(i)(s) neighborhood
    if f(s0) < f(s) then
        s ← s0
        i ← 1
        { Intensification on the modified routes of s }
        s ← Local Search with Shift(1,0)
        s ← Local Search with Shift(2,0)
        s ← Local Search with Swap(1,1)
        s ← Local Search with M2-opt
        s ← Local Search with Swap(2,1)
        s ← Local Search with Swap(2,2)
        s ← Local Search with G3-opt
        s ← Local Search with G4-opt
        s ← Local Search with kOr-opt, k = 3, 4, 5
        s ← Reverse
    end
    else
        i ← i + 1
    end
end
return s
```

possible. It should be pointed out that both directions of the route r are examined. This procedure is repeated until is no longer possible to improve the solution s. The G4-opt procedure is similar to the G3-opt with the difference that, at each iteration, the arcs $(v_i; v_{i+1})$, $(v_{l-1}; v_l)$, $(v_j; v_{j+1})$ and $(v_{k-1}; v_k)$ are removed and the arcs $(v_i; v_j)$, $(v_l; v_{j+1})$, $(v_{k-1}; v_{l-1})$ and $(v_{i+1}; v_k)$ are inserted. Finally, the reverse movement, which consists of inverting the route if there is reduction in the maximum load of the vehicle, is applied.

4 Computational results

This section presents the computational results obtained by the GENILS to solve the VRPSPD. The algorithm was coded in C++ using the Microsoft Visual C++, version 2005 and it was implemented in a Intel Core 2 Duo with 1.66 GHz and 2 GB of RAM memory running Windows Vista Home Premium 32 bits.

In order to validate the algorithm, the three set of test-problems presented in Section 1 were utilised, namely those of: Salhi and Nagy (1999), Dethloff (2001) and Montané and Galvão (2006). In the test-problems of Salhi and Nagy (1999), only those without the route duration constraints were considered. The maximum number of iterations of the GENILS adopted was 10,000.

Tables 1–3 compare the performance of the GENILS with the different algorithms proposed in the literature. In these tables, the column *Problem* indicates the test-problem considered, *Best* is the best value found by the respective author(s) and *Time* is the execution time, in seconds, of the respective algorithm. The *Gap* column shows the percentual deviation of the average solutions of the GENILS with respect to the best known solutions. The *Gap* is calculated by the equation (2).

$$\text{Gap} = 100 \times (\text{Average} - \text{Best})/\text{Best}. \tag{2}$$

Table 1 Results obtained by the GENILS in the test-problems of Dethloff (2001)

	Röpke and Pisinger		*Zachariadis et al.*		*Subramanian et al.*		*GENILS*		
Problem	*Best*	$Time^1$ (s)	*Best*	$Time^2$ (s)	*Best*	$Time^3$ (s)	*Best*	$Time^4$ (s)	Gap^* (%)
SCA3-0	636.10	232.00	636.06	2.83	**635.62**	0.90	**635.62**	6.77	0.00
SCA3-1	**697.80**	170.00	**697.84**	2.12	**697.84**	1.12	**697.84**	8.49	0.00
SCA3-2	**659.30**	160.00	**659.34**	2.58	**659.34**	1.19	**659.34**	8.13	0.00
SCA3-3	680.60	182.00	**680.04**	3.13	**680.04**	1.13	**680.04**	8.45	0.00
SCA3-4	**690.50**	160.00	**690.50**	2.68	**690.50**	1.32	**690.50**	8.09	0.00
SCA3-5	**659.90**	178.00	**659.90**	2.56	**659.90**	1.17	**659.90**	8.19	0.00
SCA3-6	**651.10**	171.00	**651.09**	4.40	**651.09**	1.23	**651.09**	8.21	0.00
SCA3-7	666.10	162.00	**659.17**	2.98	**659.17**	1.69	**659.17**	6.76	0.00
SCA3-8	**719.50**	157.00	**719.47**	3.98	**719.47**	1.08	**719.47**	8.85	0.00
SCA3-9	**681.00**	167.00	**681.00**	3.86	**681.00**	1.03	**681.00**	8.63	0.00
SCA8-0	975.10	98.00	**961.50**	3.21	**961.50**	2.52	**961.50**	5.65	0.00
SCA8-1	1052.40	95.00	1050.20	3.55	**1049.65**	2.98	**1049.65**	5.67	0.00
SCA8-2	**1039.60**	83.00	**1039.64**	4.67	**1039.64**	3.42	**1039.64**	5.92	0.00
SCA8-3	991.10	94.00	**983.34**	3.29	**983.34**	3.44	**983.34**	4.58	0.00
SCA8-4	**1065.50**	84.00	**1065.49**	2.68	**1065.49**	2.74	**1065.49**	5.98	0.00
SCA8-5	**1027.10**	96.00	**1027.08**	4.50	**1027.08**	3.44	**1027.08**	6.62	0.00
SCA8-6	972.50	93.00	**971.82**	2.67	**971.82**	2.48	**971.82**	6.57	0.00
SCA8-7	1061.00	92.00	1052.17	4.32	**1051.28**	5.39	**1051.28**	5.56	0.00
SCA8-8	**1071.20**	85.00	**1071.18**	3.43	**1071.18**	2.05	**1071.18**	5.57	0.00
SCA8-9	**1060.50**	86.00	**1060.50**	4.12	**1060.50**	3.10	**1060.50**	5.62	0.00

Table 1 Results obtained by the GENILS in the test-problems of Dethloff (2001) (continued)

	Röpke and Pisinger		*Zachariadis et al.*		*Subramanian et al.*		*GENILS*		
Problem	*Best*	*Time*[1] (s)	*Best*	*Time*[2] (s)	*Best*	*Time*[3] (s)	*Best*	*Time*[4] (s)	*Gap**(%)
CON3-0	**616.50**	171.00	**616.52**	3.89	**616.52**	2.02	**616.52**	6.77	0.00
CON3-1	**554.50**	190.00	**554.47**	2.97	**554.47**	1.83	**554.47**	7.76	0.00
CON3-2	521.40	176.00	519.26	3.32	**518.00**	2.10	**518.00**	9.28	0.00
CON3-3	**591.20**	177.00	**591.19**	2.78	**591.19**	1.34	**591.19**	9.18	0.00
CON3-4	**588.80**	173.00	589.32	3.12	**588.79**	1.79	**588.79**	6.29	0.00
CON3-5	**563.70**	179.00	**563.70**	3.45	**563.70**	1.71	**563.70**	9.16	0.00
CON3-6	**499.10**	195.00	500.80	2.98	**499.05**	1.93	**499.05**	7.33	0.00
CON3-7	**576.50**	226.00	**576.48**	2.40	**576.48**	1.52	**576.48**	6.96	0.00
CON3-8	**523.10**	174.00	**523.05**	5.02	**523.05**	1.51	**523.05**	8.75	0.00
CON3-9	**578.20**	163.00	580.05	3.14	**578.24**	1.58	**578.24**	6.87	0.00
CON8-0	**857.20**	86.00	**857.17**	3.40	**857.17**	3.74	**857.17**	6.36	0.00
CON8-1	**740.90**	81.00	**740.85**	3.73	**740.85**	2.82	**740.85**	4.88	0.00
CON8-2	716.00	84.00	713.14	2.87	**712.89**	2.46	**712.89**	6.95	0.00
CON8-3	**811.10**	91.00	**811.07**	3.82	**811.07**	2.82	**811.07**	5.87	0.00
CON8-4	**772.30**	87.00	**772.25**	2.98	**772.25**	3.37	**772.25**	5.01	0.00
CON8-5	755.70	94.00	756.91	5.76	**754.88**	3.30	**754.88**	5.82	0.00
CON8-6	693.10	96.00	**678.92**	4.00	**678.92**	3.04	**678.92**	5.67	0.00
CON8-7	814.80	94.00	**811.96**	2.46	**811.96**	2.73	**811.96**	4.71	0.00
CON8-8	774.00	94.00	**767.53**	4.21	**767.53**	3.42	**767.53**	5.23	0.00
CON8-9	809.30	92.00	**809.00**	3.87	**809.00**	3.60	**809.00**	5.86	0.00

[1] CPU time in a Pentium IV 1.5 GHz.
[2] CPU time in a Pentium IV 2.4 GHz.
[3] CPU time in a Intel Core 2 Quad 2.5 GHz.
[4] CPU time in a Intel Core 2 Duo 1.66 GHz.

Table 2 Results obtained by the GENILS in the test-problems of Salhi and Nagy (1999)

	Wassan et al.		*Zachariadis et al.*		*Subramanian et al.*		*GENILS*		
Problem	*Best*	*Time*[1] (s)	*Best*	*Time*[2] (s)	*Best*	*Time*[3] (s)	*Best*	*Time*[4] (s)	*Gap** (%)
CMT1X	468.30	48	469.80	2.89	**466.77**	1.10	**466.77**	7.82	0.00
CMT1Y	**458.96**	69	469.80	3.85	466.77	1.08	466.77	7.61	1.68
CMT2X	**668.77**	94	684.21	7.42	684.21	6.99	684.21	17.62	2.31
CMT2Y	**663.25**	102	684.21	8.02	684.21	5.84	684.21	20.10	3.16
CMX3X	729.63	294	**721.27**	11.62	721.40	6.80	721.40	59.61	0.02
CMT3Y	745.46	285	**721.27**	13.53	721.40	7.37	**721.27**	58.72	0.00
CMT12X	**644.70**	242	662.22	11.80	662.22	8.02	662.22	22.89	2.72
CMT12Y	**659.52**	254	662.22	7.59	662.22	7.32	663.50	22.33	0.60
CMT11X	861.97	504	**838.66**	17.78	839.39	12.58	846.23	48.85	0.90
CMT11Y	**830.39**	325	837.08	14.26	841.88	14.80	836.04	287.30	0.68
CMT4X	876.50	558	**852.46**	27.75	**852.46**	50.72	**852.46**	134.26	0.00
CMT4Y	870.44	405	852.46[a]	31.20	852.46	46.06	862.28	266.76	1.17[b]
CMT5X	1044.51	483	**1030.55**	51.67	**1030.55**	53.51	1033.51	768.94	0.29
CMT5Y	1054.46	533	**1030.55**	58.81	1031.17	58.74	1036.14	398.75	0.54

[1] CPU time in a Sun-Fire-V440 computer with UltraSPARC-IIIi 1062 MHz processor.
[2] CPU time in a Pentium IV 2.4 GHz.
[3] CPU time in a Intel Core 2 Quad 2.5 GHz.
[4] CPU time in a Intel Core 2 Duo 1.66 GHz.
[a] A best result of value 852.35 was found by Chen and Wu (2006).
[b] *Gap* in relation to the value found by Chen and Wu (2006).

Table 3 Results obtained by the GENILS in the test-problems of Montané and Galvão (2006)

Problem	*Montané and Galvão*		*Zachariadis et al.*		*Subramanian et al.*		*GENILS*		
–	*Best*	*Time*[1] (s)	*Best*	*Time*[2] (s)	*Best*	*Time*[3] (s)	*Best*	*Time*[4] (s)	*Gap** (%)
r101	1042.62	12.20	1019.48	10.50	1010.90	10.51	**1009.95**	35.65	−0.09
r201	671.03	12.02	**666.20**	8.70	**666.20**	6.24	**666.20**	39.62	0.00
c101	1259.79	12.07	1220.99	10.20	1220.26	12.73	**1220.18**	18.34	−0.01
c201	666.01	12.40	**662.07**	5.70	**662.07**	4.18	**662.07**	16.62	0.00
rc101	1094.15	12.30	**1059.32**	12.90	**1059.32**	9.48	**1059.32**	12.79	0.00
rc201	674.46	12.07	**672.92**	10.50	**672.92**	4.21	**672.92**	24.03	0.00
r1_2_1	3447.20	55.56	3393.31	61.80	3371.29	95.79	**3357.64**	175.81	−0.40
r2_2_1	1690.67	50.95	1673.65	47.40	**1665.58**	24.13	**1665.58**	103.44	0.00
c1_2_1	3792.62	52.21	3652.76	66.30	3640.20	95.17	**3636.74**	117.62	−0.10
c2_2_1	1767.58	65.79	1753.68	60.90	1728.14	41.94	**1726.59**	127.81	−0.09
rc1_2_1	3427.19	58.39	3341.25	45.30	3327.98	76.30	**3312.92**	299.30	−0.45
rc2_2_1	1645.94	52.93	1562.34	62.40	**1560.00**	34.28	**1560.00**	77.48	0.00
r1_4_1	10027.81	330.42	9758.77	315.30	9695.77	546.39	**9627.43**	2928.31	−0.71
r2_4_1	3685.26	324.44	3606.72	273.60	**3574.86**	231.73	3582.08	768.60	0.20
c1_4_1	11676.27	287.12	11207.37	283.50	11124.30	524.35	**11098.21**	1510.44	−0.23
c2_4_1	3732.00	330.20	3630.72	336.00	**3575.63**	293.18	3596.37	569.01	0.58
rc1_4_1	9883.31	286.66	9697.65	145.80	9602.53	550.90	**9535.46**	2244.18	−0.70
rc2_4_1	3603.53	328.16	3498.30	345.00	**3416.61**	291.15	3422.11	3306.84	0.16

[1] CPU time in a Athlon XP 2.0 GHz.
[2] CPU time in a Pentium IV 2.4 GHz.
[3] CPU time in a Intel Core 2 Quad 2.5 GHz.
[4] CPU time in a Intel Core 2 Duo 1.66 GHz.

As can be seen in these tables, in the test-problems developed by Dethloff (2001), the GENILS obtained all the best known solutions. In the 14 test-problems of Salhi and Nagy (1999), the proposed algorithm found 4 of the best known solutions, with a maximum gap of 3.16% in the remaining problems. It is important to remark that none of the algorithms has a clear superiority in the terms of solution quality in these set of instances. The best performance of the GENILS was in the test-problems of Montané and Galvão (2006), in which the proposed algorithm improved 9 of the best known solutions and equaled another 6, while in the 3 remaining test-problems the value of the maximum gap was 0.58%.

Comparing the GENILS with the other algorithms, it can be verified that proposed algorithm had a performance quite similar to the one developed by Subramanian et al. (2008). Indeed, in the test-problems of Dethloff (2001) and Montané and Galvão (2006), both the algorithms taken together produced all the best results. In this second group of test-problems, the GENILS was superior in 9 problems and inferior in 3. On the other hand, in Salhi and Nagy (1999) test-problems, the GENILS was superior in 2 cases and inferior in another 5.

A comparison in terms of execution time was not performed because the results found by the other algorithms were obtained using different machines.

5 Conclusions and future works

This work dealt with the VRPSPD. In order to solve it, a hybrid heuristic algorithm, called GENILS, was proposed. Adaptations of the Cheapest Insertion and the GENIUS heuristic were employed to generate an initial solution. To improve this solution, an ILS based procedure, that uses the VND method in the local search, was developed. The VND explores the solution space of the a solution by applying the following movements: Shift(1,0), Shift(2,0), Swap, Swap(2,1), Swap(2,2), M2-Opt and kOr-Opt. In case of improvement an intensification is performed in the modified routes using the following procedures: G3-opt, G4-opt (both based on the GENIUS heuristic) and Reverse.

From the results obtained, it can be observed that the proposed algorithm is competitive with the best approaches presented in the literature. Indeed, in a set of well-known test-problems, the GENILS was found capable to produce all the best results reported in the literature; in another one, 9 new solutions were generated and another 3 were equalled; and in a third set of instances, 3 best known solutions were equalled and the value of the maximum gap in the remaining problems of this set was of 3.16%. In addition, the GENILS obtained solutions with a variability smaller than 1% in 67 of the 72 test-problems, which corresponds to 93% of the cases. One interesting behaviour of the algorithm is the fact of obtaining a better performance in most test-problems of Montané and Galvão (2006), illustrating its potential in solving real-life applications, where it is common to deal with large-scale problems.

As for future work, one can improve the G3-opt e G4-opt procedures by considering the recombination of multiple routes. In addition, it is strategic to combine the GENILS algorithm with the TS metaheuristic, where the latter can be triggered by replacing the VND, e.g., after a certain number of ILS iterations. This has to do with the fact that the TS is the base algorithm of those developed by

Wassan et al. (2008) and Zachariadis et al. (2009), where both have obtained most of the best known results in the test-problems of Salhi and Nagy (1999), and it was in this set that the GENILS had its worst performance.

Acknowledgements

This research was partially supported by CAPES, CNPq, FAPEMIG and FAPERJ.

References

Anderberg, M.R. (2007) 'Cluster analysis for applications', *Monographs and Textbooks on Probability and Mathematical Statistics*, Academic Press, Inc., New York.

Bianchessi, N. and Righini, G. (2007) 'Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery', *Computers and Operations Research*, Vol. 34, No. 2, pp.578–594.

Chen, J.F. (2006) 'Approaches for the vehicle routing problem with simultaneous deliveries and pickups', *Journal of the Chinese Institute of Industrial Engineers*, Vol. 23, No. 2, pp.141–150.

Chen, J.F. and Wu, T.H. (2006) 'Vehicle routing problem with simultaneous deliveries and pickups', *Journal of the Operational Research Society*, Vol. 57, No. 5, pp.579–587.

Crispim, J. and Brandão, J. (2005) 'Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls', *Journal of the Operational Research Society*, Vol. 56, No. 7, pp.1296–1302.

Dantzig, G.B. and Ramser, J.H. (1959) 'The truck dispatching problem', *Management Science*, Vol. 6, pp.80–91.

Dell'Amico, M., Righini, G. and Salanim, M. (2006) 'A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection', *Transportation Science*, Vol. 40, No. 2, pp.235–247.

Dethloff, J. (2001) 'Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up', *OR Spektrum*, Vol. 23, pp.79–96.

Gendreau, M., Hertz, A. and Laporte, G. (1992) 'New insertion and post optimization procedures or the traveling salesman problem', *Operations Research*, Vol. 40, pp.1086–1094.

Gökçe, E.I. (2004) *A Revised Ant Colony System Approach to Vehicle Routing Problems*, Master's Thesis, Graduate School of Engineering and Natural Sciences, Sabanci University, Turkey.

Halse, K. (1992) *Modeling and Solving Complex Vehicle Routing Problems*, PhD Thesis, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, Denmark.

Hansen, P. and Mladenović, N. (2001) 'Variable neighborhood search: principles and applications', *European Journal of Operations Research*, Vol. 130, pp.449–467.

Min, H. (1989) 'The multiple vehicle routing problems with simultaneous delivery and pick-up points', *Transportation Research A*, Vol. 23, No. 5, pp.377–386.

Montané, F.A.T. and Galvão, R.D. (2006) 'A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service', *Computers and Operations Research*, Vol. 33, No. 3, pp.595–619.

Nagy, G. and Salhi, S. (2005) 'Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries', *European Journal of Operational Research*, Vol. 162, pp.126–141.

Or, I. (1976) *Traveling Salesman-Type Combinational Problems and their Relation to the Logistics of Blood Banking*, PhD Thesis, Northwestern University, USA.

Rego, C. and Roucairol, C. (1996) 'Meta-heuristics theory and applications', *A Parallel Tabu Search Algorithm using Ejection Chains for the Vehicle Routing Problem*, Kluwer Academic Publishers, Boston, pp.661–675.

Röpke, S. and Pisinger, D. (2006) *A Unified Heuristic for a Large Class of Vehicle Routing Problems with Backhauls*, Technical Report 2004/14, University of Copenhagen, Denmark.

Salhi, S. and Nagy, G. (1999) 'A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling', *Journal of the Operational Research Society*, Vol. 50, pp.1034–1042.

Stützle, T. and Hoos, H.H. (1999) 'Analyzing the run-time behaviour of iterated local search for the tsp', *Proceedings of the Third Metaheuristics International Conference*, Angra dos Reis, Brazil, pp.449–453.

Subramanian, A. (2008) *Metaheurística Iterated Local Search aplicada ao problema de roteamento de veículos com coleta e entrega simultânea*, Master's Thesis, Universidade Federal da Paraíba, João Pessoa, Brazil.

Subramanian, A., Cabral, L.A. F. and Ochi, L.S. (2008) *An Efficient ILS Heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery*, Technical Report, Universidade Federal Fluminense, available at http://www.ic.uff.br/PosGraduacao/RelTecnicos/401.pdf

Topcuoglu, H. and Sevilmis, C. (2002) 'Task scheduling with conflicting objectives', in Yakhno, T.M. (Ed.): *ADVIS*, Lecture Notes in Computer Science, Springer, Berlin, Vol. 2457, pp.346–355.

Vural, A.V. (2003) *A GA based Meta-Heuristic for Capacited Vehicle Routing Problem with Simultaneous Pick-up and Deliveries*, Master's Thesis, Graduate School of Engineering and Natural Sciences, Sabanci University, Turkey.

Wassan, N.A., Wassan, A.H. and Nagy, G. (2008) 'A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries', *Journal of Combinatorial Optimisation*, Vol. 15, No. 4, pp.368–386.

Zachariadis, E.E., Tarantilis, C.D. and Kiranoudis, C.T. (2009) 'A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service', *Expert Systems with Applications*, Vol. 36, No. 2, pp.1070–1081.