Procedia Computer Science
Volume 80, 2016, Pages 1212–1222
ICCS 2016. The International Conference on Computational Science

A VNS-based heuristic for solving the vehicle routing problem with time windows and vehicle preventive maintenance constraints

Amine Dhahri[1], Anis Mjirda[2], Kamel Zidi[3], and Khaled Ghedira[4]

[1] National School of Computer Sciences, University of Manouba, 2010, Tunisia
aminedhahri@gmail.com
[2] Technological Research Institute Railenium, France
anis.mjirda@gmail.com
[3] Tabuk University, Community College, Saudi Arabia
zidi@ut.edu.sa
[4] National School of Computer Sciences, University of Manouba, 2010, Tunisia
khaled.ghedira@isg.rnu.tn

Abstract

We address a vehicle routing problem with time windows (VRPTW) that also contains vehicle with preventive maintenance constraints (VRPTW-PM) and propose a MIP mathematical formulation as well as a general variable neighborhood search metaheuristic (VNS) to solve with large instance the problematic situation. First we create a initial solution using Solomon heuristic then we minimize the number of used routes, and then the total travelled distance by all vehicles is minimized. Computational results show the efficiency of the proposed approach.

Keywords:

1 Introduction

Vehicle routing problem (VRP) is a well-known NP-hard combinatorial optimization problem. It was proposed by Dantzig and Ramser in [1] and it is identified as a plan to follow for supplying a set of customers using a set of vehicles, at the same time the cost of allocating vehicles to customers must be minimized. A review of the VRP and resolution approaches can be found in [6]. Vehicle routing problem with time windows (VRPTW) is a variant of VRP in which the delivery beginning time is bounded by a time slot. In this paper, we propose extension of the well-known VRPTW by adding vehicle preventive maintenance constrained. Indeed, a fleet of vehicles should be checked from time to time to avoid unpleasant interruption of services. For that we propose a VNS metaheuristic to solve this problem. The majority of studies related to the planning of the supply chain are placed in an environment where all vehicles are available.

doi:10.1016/j.procs.2016.05.473

However, in reality this is not the case. Indeed, physical resources such as vehicles may be unavailable for various reasons such as the daily driving limit. This problematic situation is treated in [8]. Other deterministic instances of vehicle-based disruption can be due to vehicle preventive maintenance inspections scheduled by the maintenance department, which is our case. Vehicles can therefore be unavailable during certain periods, where dates and times are fixed and known beforehand (deterministic case). The problem addressed in this paper is a variant of the VRPTW. In this case, the availability involves a vehicle k that requires maintenance activities at time τ after leaving the depot and before making a delivery on its route to the final customer. We only look at the case where one vehicle is available because it is unusual to have more than one vehicle that requires maintenance activities at the same time. In this situation, we need to prepare a plan reflecting the constraints and objectives of the evolved environment while minimizing the negative impact of disruption. Extra vehicles (EVs) may have to be used to finish the provision of service to the unserved customers. The objective is to minimize the number of used vehicles as well as the total travelled distance by all the vehicles. We describe the the mathematical programming formulation in the next section.

2 Mathematical formulation

The VRPTW-PM is defined as follows: given a graph (V,A) where $V = \{0, ..., n + 1\}$ a set of vertices and $A = \{(i;j)|i, j; \in V\}$ an arc set. A set K of identical vehicles with capacity Q, localized initially at the depot 0 should attend n customers. For each customer i, there is known demand denoted by q_i and a service time windows (e_i, l_i) is defined, where e_i is the earliest time that service can begin and l_i is the latest time that service can begin and service time s_i. We denote by b_{ik} the time that vehicle k starts delivering customer i. Each arc (i, j) has a non-negative distance d_{ij} and we denotes by T_k the time that vehicle k starts a preventive maintenance activities (PM), and τ_k the time that PM take. b_{ik} is the set of variables to decide when to serve a customer i with vehicle k. The objective is to find a set of routes for vehicles k delivering the demand q_i for each customer i and satisfying the time windows constraints and PM constraints while minimizing the total traveling distance. To model the problem we introduce the following variables.

$$z_{ijk} = \begin{cases} 1 & \text{if } T_k \in [b_{ik}, b_{jk}] \\ 0 & \text{else} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \text{ travel from customer } c_i \text{ to } c_j \\ 0 & \text{else} \end{cases}$$

$$\min \quad z = \sum_{i \in V} \sum_{i \in V} \sum_{k=1}^{v} c_{ij} x_{ijk} \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{j \in N} x_{ijk} = 1 \quad \forall i \in V \tag{2}$$

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

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

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

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

$$b_{ik} + s_i + t_{ij} + z_{ijk}\tau_k \leq M * (1 - x_{ijk}) + b_{jk} \tag{7}$$
$$i, j = 0, 1, .., n, i \neq j \;\; \forall k \in K \;\; \forall (i,j) \in A$$

$$e_i \leq b_{ik} \leq l_i \quad \forall k \in K, \quad \forall i \in N \tag{8}$$

$$x_{ijk} \in \{0, 1\} \tag{9}$$

$$z_{ijk} \in \{0, 1\} \tag{10}$$

The objective function (1) represents the total cost to be minimized, it includes minimizing the total travel distance. Constraints (2) ensure that customer i is delivered only by one vehicle k. Constraints (3)-(5) are flow constraints of the vehicle k; that is, each vehicle leaves the depot, visits the customers, and then, returns to the depot. Constraints (6) guarantee that the vehicle capacity is not exceeded. Constraints (7) and (8) signify the time constraints; to make sure that time window is not violated, where M_{ij} are large constants. For determining the value of decision variable z_{ijk} when solving the MIP model for a given benchmark problem and due to the dependence of b_{ik}, we must introduce a logical constraint when defining constraint (7) as; if $T_k \in [b_{ik}, b_{jk}]$ then $z_{ijk} = 1$ else $z_{ijk} = 0$. According to [5], M_{ij} can be replaced by $max\{e_i + d_i + c_{ij} - e_j, 0\}\,\forall (i,j) \in A$. Constraints (9) and (10) define the domain of the decision variables x and z.

3 Variable neighborhood search for the VRPTW-PM

Proposed by Hansen and Mladenovic in [7], variable neighborhood search VNS is a metaheuristic based on the concept of systematic change of neighborhoods during the search, knowing that there are several rules of crucial importance to govern this change. The basic VNS method has several steps. Firstly it requires the definition of different and several neighborhood structures, denoted $\mathcal{N}_1, \ldots, \mathcal{N}_{k\,\max}$ generally from different sizes and cardinalities. A relation order between these structures is defined in terms of the problem. From an initial solution x, a neighbor of x is selected randomly using the neighborhood structure $\mathcal{N}_1$. If the solution obtained after the application of a local search method (which is defined) is better than the initial solution, then VNS resumes the search from this solution. Otherwise, a solution is selected using a neighborhood of upper range, (in our case, it is the neighborhood $\mathcal{N}_2$). Local search is applied to this solution, and so on. This method allows escaping from a local optimum if it is a local optimum for a neighborhood for a range smaller than the *max*. General VNS (GVNS) uses a Variable Neighborhood Descent (VND) as its local search subroutine.

3.1 Feasibility checking

To check the feasibility of a given solution we must check the arrival time b_{jk} of the vehicle k that must supply the customer j, e_i, b_{jk}, $d_{(i,j)}$, z_{ijk} and π_k are described in section 2 ;

$$b_{jk} \quad = \begin{cases} e_i & \text{if } b_{ik} + s_i + d_(i,j) \leq e_i + z_{ijk}\pi_k \\ b_{ik} + s_i + d_(i,j) + z_{ijk}\pi_k & \text{otherwise} \end{cases}$$

3.2 Initial Solution

In this work we used the Nearest Neighbor algorithm for which Solomon in [2] proposed a cost function to generate an initial solution. The nearest neighbor heuristic starts by finding the closest (in terms of a measure) unrouted customer to the depot. At every subsequent iteration, the heuristic searches for the closest customer to the last customer added to the route. This search is performed among all the customers. The mentioned cost function is defined as follows:

$$C_{ij} = \gamma_1 D_{ij} + \gamma_2 T_{ij} + \gamma_3 V_{ij} \tag{11}$$

$$\gamma_1 + \gamma_2 + \gamma_3 = 1, \gamma_1 \geq 0, \gamma_2 \geq 0, \gamma_3 \geq 0$$

where i is a customer in the partial solution and j is not. With D_{ij} the Euclid distance from customer i to customer j. T_{ij} the Time difference between completing i and starting j and V_{ij} the Time remaining until last possible start of j following i.

3.3 Neighborhood structures

Generation of the whole neighborhood structure is the most important part of the VNS metaheuristic. In this paper six adjacent structures were designed. There is Intra-route and Inter-route neighborhood structures. Neighborhood structures are,

- $Two-opt$ ($\mathcal{N}_1$): The basic idea is replacing two arcs in a route by two others of the same route while inverting the direction of the route.
- $Insert-Move(k)$ ($\mathcal{N}_2(k)$): The Insert-Move neighbourhood consists in changing the position of a segment of k customers.
- $Swap-move(k)$ ($\mathcal{N}_3$): The basic idea is to relocate a set of consecutive of k-consecutive vertices, this is achieved by the relocation of three arcs in the original route by another while maintaining the same orientation of route.
- $Two-opt^*$ ($\mathcal{N}_4$): The basic idea of the Two-opt* is to combine two routes to insert the last customers of a given route after the first customer of another route while maintaining the orientation of the route.
- $Relocate(k)$ ($\mathcal{N}_6(k)$): The basic idea is to change the location of a customer in another route. In our case we exchanged the location of k customers located consecutively in the original route.
- $Exchange(k,k1)$ ($\mathcal{N}_5(k,k1)$): This operator enables the exchange of a value of k and $k1$ customers between two different routes.

3.4 Shaking

In order to diversify the search in solution space, we introduced a shaking procedure that uses different neighborhood structure depending on the value of k. In table 1, we give a description about the used neighborhood structure for each value of k. The column segment length represent the number of consecutive customer. We note that the shaking phase may lead to infeasible solution but we accept only feasible ones.

k	Neighborhood Structure	Segment length
1	Insert-Move	1
2	Insert-Move	2
3	Relocate	1
4	Relocate	2
5	Relocate	3
6	swap	(1,1)
7	swap	(2,2)
8	Exchange	(1,1)
9	Exchange	(2,2)
10	Exchange	(1,2)

Table 1: The set of neighborhood structure used in the shaking phase

3.5 Route minimizing procedure

In the literature, the objective function is defined as hierarchy of objectives for the VRPTW, in which we minimize the number of vehicles at first then the total travel distance. We define a procedure of minimizing the number of routes based on relocating procedure described in algorithm 1a and exchange neighbourhood procedure presented in algorithm 1b. We propose a VNS with a local search in wich we apply algorithm 1a and algorithm 1b. The procedure stops when there is no routes to eliminate; Algorithm 1

```
for r1 the route that have the minimum number of customer do
  for r2 =1...$v$ do
    if r1 $\neq$ r2 then
      for each customer k $\in$ r1 do
        for each position in r2 do
          if $r2$ after insertion of $k$ in position $i$ is feasible then ($s \leftarrow s'$)
        end for
      end for
    end if
  end for
end for
```

(a) modified relocate procedure

```
for r1 the route that have the minimum number of customer do
  for r2 =1...$v$ do
    if r1 $\neq$ r2 then
      for each customer k $\in$ r1 do
        for each customer k1 $\in$ r2 do
          if $r2$ after insertion of $k$ and $r1$ after insertion of $k1$ are feasible then ($s \leftarrow s'$)
        end for
      end for
    end if
  end for
end for
```

(b) modified exchange procedure

Figure 1: Used heuritics for minimizing the number of route

Algorithm 1 Route minimizing procedure

```
Initialization: initial solution $x$, set of neighbourhood structure $\mathcal{N}_k, k = 1,2$ and $3$ ($\mathcal{N}_2(1), \mathcal{N}_2(2)$ and $\mathcal{N}_3$). see 3.3
repeat
  $K \leftarrow 1$
  repeat
    Shaking: generate randomly a solution $y$ from the $k_{th}$ neighborhood $y \in \mathcal{N}_{k(x)}$
    Local Search: apply 1a and 1b with $y$ initial solution to obtain a local optimum denoted $y_1$
    if the local optimum $y_1$ is better than $x$ then
      (1) ($x \leftarrow y_1$)
      (2) continue the search with $\mathcal{N}_1 (k = 1)$
    else
      $k \leftarrow k + 1$.
    end if
  until $k < k_{max}$
until there is no route to eliminate
```

3.6 Variable Neighbourhood Descent

An important element is to enhance the initial solution using the defined neighborhood structures. In this section, we present the VND algorithm which will be used as a local search subroutine in our proposed GVNS algorithm. The VND algorithm requiere as input an initial solution which will be denoted by x ans the set of the defined neighborhood structures. At each iteration, a local search procedure with the first improvement strategy is applied on a solution for a given neighborhood structure. The search continue in the next neighborhood structure whether there is an improvement or not. Algorithm stops when there is no more improvement. Different steps of the VND procedure is give in algorithm 2.

Algorithm 2 Variable Neighborhood Descent

```
Require: an initial solution $x$.
  $x_1 \leftarrow$ First improvement on $x$ using $\mathcal{N}_1$
  $x_2 \leftarrow$ First improvement on $x1$ using $\mathcal{N}_6(1)$
  $x_3 \leftarrow$ First improvement on $x2$ using $\mathcal{N}_6(2)$
  $x_4 \leftarrow$ First improvement on $x3$ using $\mathcal{N}_2(1)$
  $x_4 \leftarrow$ First improvement on $x3$ using $\mathcal{N}_2(2)$
  $x_6 \leftarrow$ First improvement on $x5$ using $\mathcal{N}_3$
  $x_7 \leftarrow$ First improvement on $x6$ using $\mathcal{N}_5(1,1)$
  $x_8 \leftarrow$ First improvement on $x7$ using $\mathcal{N}_5(1,2)$
  $x_9 \leftarrow$ First improvement on $x8$ using $\mathcal{N}_5(2,2)$
  if $f(x_9) < f(x)$ or Acceptance Criterion is satisfied 3.7 then
    $x \leftarrow x_9$
  end if
```

3.7 Acceptance Criterion

After shaking and VND phases have been completed, the resulting solution should be compared with the current one. In order to decide whether it will be selected or not, we adopted an acceptance criterion. In the basic VNS, only solutions that improve the cost will be chosen, but we can easily find them in a local optimum. In most cases, therefore, it is essential to have a strategy that does not accept any improvement solution under certain conditions. Instead, we put a system that is inspired by the simulated annealing (SA) [3]. More specifically, the solution obtained after the local search procedure is always accepted if the quality of the solution obtained is better and is accepted with a cost higher than the current solution with a probability $\exp(-(f(x') - f(x)/T))$ with $f(x)$ being of the solution. The temperature T decreases linearly in n steps during the search Thus, in each n iteration, T is reduced by an amount $T * n/T_{max}$, T_{max} represents the total number of iterations. Tests showed a satisfying performance when setting $n = 100$ and using an initial temperature value of $T_0 = 10$.

3.8 General Variable Neighborhood Search

The two phase proposed approach is presented in algorithm 3. First in step 1 we start by generate a initial solution created by algorithm 3.2 than the number of routes is minimized. Once we have a solution with minimum number of routes the total travelled distance is minimized in step 2 using a GVNS wish accept solution that not improve cost under a given probability. The pseudo of proposed approach is presented in 3

4 Experimental Results

In this section we describe results of applying the proposed approach to solve the VRPTW-PM using instances that are are generated based on Solomon benchmark[4]. The description of Solomon's instances is: the geographical data are randomly generated by a random uniform distribution in problem set R1, clustered in problem set C1, and a mix of randomly generated data and clusters in problem set RCl. All the test problems are 100-customer euclidean problems. Travel times between customers are equal to the corresponding distances truncated to one decimal place. The customer demands are relatively small compared to the vehicle capacity. In problem sets R1 and RC1 the time windows have a uniformly distributed, randomly generated center and a normally distributed random width. Proposed approaches are implemented using JAVA language and executed on a 2.53 GHz intel Core i5 with 4GB RAM. MIP model was solved with ILOG CPLEX 12 MIP solver. We denote by f the value of the objective function. In table 2, (2,10,20) denotes that vehicle 2 starts the maintenance activity at time 10 and takes 20 units of time. A benchmark of 20 customers (from customer 1 to 20) from Problem $R109$ is used to validate the MIP model and to compare GVNS with implementation of the MIP model with small benchmark instance. We denote by f the best result obtained in the 5 runs and by $\#v$ its number of used vehicles. The optimal solution without preventive maintenance for this instance is:

route 1: Total demand=50, Distance =67.344: 1-6-9-19-7-1
route 2: Total demand=67, Distance =105.49: 1-13-10-4-5-1
route 3: Total demand=79, Distance =105.84: 1-3-16-15-17-18-14-1
route 4: Total demand=60, Distance =81.43: 1-8-20-12-11-2-1
Total travel distance by 4 vehicles is 360.12

Algorithm 3 Pseudo code of proposed approach

```
step 1: minimizing the number of routes apply route minimization procedure to minimize
the number of route by applying algorithm 1
step 2: minimizing the total travelled distance
Initialization: Set of neighbourhood structure $\mathcal{N}_k, k = 1, ..., 10$ described in table 1, initial
solution $x$ with minimal number of route. Initial temperature $T = 10$, $i = 0$, $Time = 0$ in
seconds, $j = 0$
repeat
  $K \leftarrow 1$
  repeat
    Shaking: generate randomly a solution y from the $k_{th}$ neighborhood; $y \in \mathcal{N}_{k(x)}$
    Local Search: apply The VND method described in algorithm 2 with $\mathcal{N}_s$ and $y$ initial
    solution to obtain a local optimum denoted $y_1$
    Let $n \in [0, 1]$ a random number and $prob = \exp(-(f(y_1) - f(x)/T))$
    if the local optimum $y_1$ is better than $x$ or $n < prob$ then
      (1) $(x \leftarrow y_1)$
      (2) continue the search with $\mathcal{N}_1 (k = 1)$
    else
      $k \leftarrow k + 1$.
    end if
  until $k < 10$ or $T == 1$ or $Time < 2000$
  if $j = i + 100$ then
    $T = T - T * n/k$;
    $i = j$;
  end if
  $j \leftarrow j + 1$;
until $T == 1$ or $Time < 2000$ or $j < 4000$
```

We make a assumption that the optimal value for $R109$ benchmark problem which is 360.12 is the lower bound (LB) for VRPTW-PM for the same instance. Since the number of vehicle for the original plan is small, equal to 4, we set it automatically, equal to 4 or 5 for the implementation of VRPTW-PM.

As shown in table 2 there is a large gap between executing the optimal plan generated by the implementation of MIP model and the plan generated by adding a vehicle to supply unserved customers due to a maintenance activity (Easy Plan). In terms of solution quality, we can see that the proposed procedure performs consistently better than when using the easy plan knowing that in each problem an extra vehicle is required to serve all customers in their time windows except problems which are not affected by PM such as configuration (1,100,30). Also solutions produced by the GVNS approach are all the same then the optimal solution obtained by the implementation of the MIP model. For large instances(100-customers benchmark) and after a series of test we conclude that the MIP model cant find the optimal solution on a time limit equal to 7200 seconds so it is necessary to use a metaheuristic, GVNS in our case to get near-optimal solution in a reasonable execution time. We assume that the best known solution of VRPTW can be considered as a Lower Bound for VRPTW-PM. To see the effectiveness of GVNS approach for the VRPTW-PM with large instances(100 customers), a series of experiments is conducted using different scenarios of unavailability of vehicles with different starting period of the maintenance activities throughout the horizon plan as shown in table 3. The $\gamma = \{1, 2, 3\}$

Table 2: Comparison with the results of MIP model and GVNS with 20 customers from $R109$ Benchmark problem

Scenarios	VRPTW-PM (MIP)			Easy Plan		GVNS		
	$\#v$	f	Time (s)	$\#v$	f	$\#v$	f	Time (s)
(1,10,20)	4	367.77	27.86	5	391.74	4	367.77	5.23
(1,80,50)	4	381.20	19.68	5	391.74	4	381.20	3.00
(1,100,30)	4	360.12	7.82	5	375.93	4	360.12	0.12
(2,10,70)	4	406.82	102.31	5	360.12	4	406.82	20.48
(2,20,40)	4	369.57	0.66	5	360.12	4	369.57	0.00
(2,50,60)	4	396.31	21.18	5	386.98	4	396.31	2.40
(3,10,40)	4	402.55	29.01	5	414.80	4	402.55	3.09
(3,40,70)	4	401.26	22.67	5	430.49	4	401.26	12.21
(3,90,40)	4	372.60	17.49	5	414.80	4	372.60	12.80
(1,10,20) (2,20,10)	4	367.77	43.13	5	391.74	4	367.77	8.09
(1,40,35) (2,50,35)	5	379.81	21.80	6	391.74	5	379.81	0.61
(1,10,50) (4,10,70)	5	406.77	23.51	6	464.30	5	406.77	2.19
(2,60,40) (4,70,40)	5	379.81	9.93	6	417.26	5	379.81	1.04
(2,10,60) (3,10,40)	5	394.12	29.90	6	415.49	5	394.12	0.03
(1,100,35) (2,90,35)	4	372.60	22.55	6	375.93	4	372.60	3.2

values corresponds to unavailability of the vehicle during a period of time. When $\gamma = 1$, this is means that vehicle is unavailable at the beginning of the planning horizon; when $\gamma = 2$, vehicle is unavailble during the middle and when $\gamma = 3$ corresponds to an unavailability that starts at the end of horizon planning.

Table 3: Configuration for maintenance activity

	$\gamma = 1$	$\gamma = 2$	$\gamma = 3$
R1	(30,40)	(100,40)	(200,40)
R2	(50,40)	(500,40)	(900,40)
C1	(100,40)	(700,40)	(1000,40)
C2	(100,40)	(1500,40)	(2500,40)
RC1	(30,40)	(100,40)	(200,40)
RC2	(50,40)	(100,40)	(200,40)

The vehicle that must apply maintenance activity is the vehicle that made the longest distance based on the original plan. Knowing that the original plan is a solution for VRPTW generated by GVNS approach. The used instances consist of six different types (R1,C1,RC1,R2,C2,RC2), Two problems are selected for each type, which are R103, R111, C101, C104, RC104, RC107, R203, R204, C206, C207, RC203 and RC204. When testing problems with 100 customers, and in terms of solution quality, we can see that the proposed procedure performs consistently better than when using the easy plan.

Table 4: Computational results for VRPTW-PM with Solomon 100 customers Benchmark

Problem	values	Original Plan	Easy Plan			VRPTW-PM (VNS)		
			$\gamma = 1$	$\gamma = 2$	$\gamma = 3$	$\gamma = 1$	$\gamma = 2$	$\gamma = 3$
R103	$\#v$	13	14	14	14	14	14	13
	f	1292.67	1362.05	1362.05	1292.67	1246.74	1249.25	1292.67
R1011	$\#v$	10	11	11	11	11	11	11
	f	1101.43	1162.15	1162.15	1101.43	1067.09	1090.33	1101.43
R203	$\#v$	3	4	4	3	4	4	3
	f	941.65	1042.60	1041.97	982.71	1042.60	1041.97	982.71
R204	$\#v$	2	3	3	3	3	3	3
	f	831.04	938.45	862.45	862.45	938.45	862.45	862.45
C101	$\#v$	10	11	10	10	11	11	10
	f	828.93	923.66	824.77	828.93	852.94	931.01	824.77
C104	$\#v$	10	11	11	10	10	11	10
	f	824.77	932.65	926.37	828.93	849.66	848.78	828.93
C206	$\#v$	3	4	4	4	4	4	4
	f	591.55	607.93	623.59	630.45	607.93	629.52	630.45
C207	$\#v$	3	4	4	4	4	4	4
	f	588.28	669.87	648.68	626.65	669.87	648.68	607.93
RC104	$\#v$	10	11	11	10	11	11	10
	f	1150.21	1257.58	1242.35	1150.21	1158.42	1235.87	1150.21
RC107	$\#v$	11	12	12	12	12	12	11
	f	1232.26	1329.99	1327.05	1232.26	1230.75	1484.00	1232.26
RC203	$\#v$	3	4	4	4	4	4	4
	f	1093.49	1328.87	1328.87	1305.83	1282.58	1284.90	1226.05
RC204	$\#v$	3	4	4	5	4	4	4
	f	801.39	903.45	903.45	903.45	903.45	903.45	903.45

5 Conclusion

In this paper, we have presented a scheduling model for the vehicle routing problem with time windows in case of unavailability of vehicles due to maintenance activities at a given time and for a defined period after which a subset of customers have been visited. The objective function is to find a new planning/scheduling that minimizes the deviation from the original planning in term of number of vehicles and total travelled distance. Although the problem of planning/scheduling is complicated by a variety of factors, this paper provides an approach based on GVNS to treat this type of problem. Experiments were equally carried out to study how maintenance activity works. Experimental results also showed that the GVNS algorithm consistently produces high performance.

References

[1] George B Dantzig and John H Ramser. The truck dispatching problem. Management science, 6(1):80–91, 1959.

[2] Marius M Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations research, 35(2):254–265, 1987.

[3] Scott Kirkpatrick. Optimization by simulated annealing: Quantitative studies. Journal of statistical physics, 34(5-6):975–986, 1984.

[4] Marius M Solomon. Vrptw benchmark problems. Excerpt from unpublished article, n. pag. http://www. cba. neu. edu/˜ msolomon/.(15 May 2000), 2003.

[5] Jean-Francois Cordeau, Guy Desaulniers, Jacques Desrosiers, Marius M Solomon, and François Soumis. Vrp with time windows. The vehicle routing problem, 9:157–193, 2001.

[6] Gilbert Laporte. Fifty years of vehicle routing. Transportation Science, 43(4):408–416, 2009.

[7] Nenad Mladenović and Pierre Hansen. Variable neighborhood search. Computers & Operations Research, 24(11):1097–1100, 1997.

[8] Eric Prescott-Gagnon, Guy Desaulniers, Michael Drexl, and Louis-Martin Rousseau. European driver rules in vehicle routing with time windows. Transportation Science, 44(4):455–473, 2010.