A Hybrid Evolutionary Approach for a Vehicle Routing Problem with Double Time Windows for the Depot and Multiple Use of Vehicles

Sonia Hajri-Gabouj[(1, 2)]

Saber Darmoul[(1, 2)]

[(1)]INSAT, Institut National des Sciences Appliquées et de Technologie

Centre urbain nord, BP 676

1080 TUNIS.

[(2)]LARA, Ecole Nationale d'Ingénieurs de Tunis

BP 37, le Belvédère

1002 TUNIS

Sonia.Gabouj@insat.rnu.tn

Abstract: This paper deals with a new extension of the basic vehicle routing problem including depot double time windows and multiple use of vehicles (VRPDM). The VRPDM is a combination between a variant of the vehicle routing problem with time windows (VRPTW) and the Vehicle Routing Problem with Multiple use of vehicles (VRPM). It consists in designing and assigning multiple routes to vehicles within a given planning period in order to service a set of customers. The depot has double time windows, one for loading the vehicles and the other for their returning back. The aim is to minimize the number of required vehicles. To solve this combinatorial optimisation problem, an evolutionary algorithm incorporating new combinations of crossover and mutation operators is developed. Computational testing on 70 problem instances shows the effectiveness of the suggested approach.

Keywords: Vehicle routing problem, double time windows, multiple use of vehicles, optimization, evolutionary algorithm.

1. Introduction

The motivation of this work arises from a distribution real case problem facing the fuel supplying companies. We are interested in designing and scheduling vehicle routes within a given planning period to cover known customer demands. Customers do not have any time window. However the depot has double time windows, one for loading the vehicles and the other for their returning back. Servicing customers even after the depot closes for loading is allowed. Moreover, making use of a vehicle involves a considerable fixed cost. As customer locations are near the depot and the vehicle capacity is relatively small, customers are quickly serviced. Consequently, a vehicle should be assigned several routes in order to use fewer vehicles.

The problem is considered as a Vehicle Routing Problem with Double time windows for the depot and Multiple use of vehicles (VRPDM). It is a new extension of the basic vehicle routing problem (VRP) [10], which consists of a combination between the Vehicle Routing Problem with Multiple use of vehicles (VRPM) and a new variant of the Vehicle Routing Problem with Time Windows (VRPTW).

The basic formulation of VRPs was introduced by Dantzig and Ramser in 1959 [11]. The model considers a set of customers with known demands and locations that must be served from one depot. A set of homogeneous vehicles with equal capacity is available to service the customers. The aim is to design a set of routes servicing all customers in order to minimize a total cost. All routes must start and end at the depot and the vehicle capacity must not be exceeded [1]. In VRPTW, every customer can not be serviced before its time window opens and after its time window closes [27]. The depot has only one time window that is called the scheduling horizon and within which vehicles may leave and return back to the depot. The VRPM is another variant of the VRP where the vehicles may perform several routes as long as the total duration of the routes for each vehicle does not exceed a pre-defined limit T_{max}.

To solve routing problems, exact as well as heuristic approaches have been designed [8]. Laporte et al. [21] listed 500 main bibliographic references. The VRPTW has been widely studied in the literature. Kallehauge et al. [18], Desrochers et al. [12], Kohl et al. [19] and Kolen et al. [20] provided exact approaches. However, finding an optimal solution to the VRPTW is NP-hard and becomes NP-complete if the fleet size is fixed [26]. Accordingly, heuristic methods are more frequently applied to solve large size practical problems [24, 30]. For example, the route construction heuristics of Solomon [27], the local search of Savelsbergh [26], the meta-strategy simulated annealing and tabu search of Osman [23], the ant colony of Gambardella et al. [14], the Tabu Search of Cordeau et al. [9] and the genetic and evolutionary algorithms of Berger et al. [3], Bräysy et al. [6], Homberger et al [17], Thangiah [32] and Potvin et al. [25]. In [7], it is shown that the evolutionary approach proposed in [17] is currently the most efficient for solving the VRPTW. On the other hand, the VRPM has received little attention in the literature despite its importance. In fact, only Taillard et al. [29], Brandão et al. [4, 5] and Zhao et al. [33] addressed the VRPM and solved it by tabu search heuristics. Fagerholt [13] and Suprayogi et al. [28] also designed solution approaches to solve ship routing problems considered as vehicle routing problems with multiple use of vehicles and a heterogeneous fleet. However, the major drawback of these two approaches is that they are not well suited to solve large unconstrained problems.

To the best of our knowledge, the VRPs with depot double time windows (VRPD), the VRPM with time windows and the VRPDM are new variants of VRPs that have not yet been addressed in the literature. In this paper, we have developed a hybrid evolutionary approach to solve the VRPDM, incorporating new combinations of crossover and mutation operators.

The remainder of this paper is organized as follows. Section 2 describes the real case we address and gives a mathematical formulation of the VRPDM. Section 3 presents the hybrid evolutionary approach. Section 4 reports computational results and analysis for 70 problem instances involving different customer sizes and volume ranges.

The VRPDM consists in designing vehicle routes to satisfy a set of known customer demands of a fuel product. Vehicles can perform multiple routes leaving the depot at time e_0, at the earliest and returning to the depot at time l_{0r}, at the latest. However, the depot is open to load vehicles only during the time interval $[e_0 ; l_{0\ell}]$ where $l_{0\ell} < l_{0r}$. Though, vehicles still have the possibility to service customers after $l_{0\ell}$. Accordingly, the depot is considered to have double time windows. To model the VRPDM as a mathematical program, let us define :

m: number of required vehicles.
n: number of required routes.
N: number of customers.
K: maximum number of routes.
C_i: customer i.
t_i^k : arrival time at customer i using route k.
t_{ij}: travelling time from C_i to C_j.
d_0^k : starting time of route k on the depot.
s_i: service time at customer i.
s_0^k : loading time of route k at the depot C_0.
V_k: volume carried on route k.
ϑ_i : volume requested by C_i.
C_{max}: vehicle capacity.
T_{max}: driver maximum working day hours.
$[e_0; l_{0\ell}]$: time window for loading vehicles in the depot.
$[e_0; l_{0r}]$: time window for the returning back of vehicles to the depot ($l_{0\ell} < l_{0r}$).

We also define the following 0-1 decision variables:

$$x_{ijk} = \begin{cases} 1 & \text{if route k travels directly from } C_i \text{ to } C_j \\ 0 & \text{otherwise} \end{cases}$$

$$b_k = \begin{cases} 1 & \text{if route k exists} \\ 0 & \text{otherwise} \end{cases}$$

$$\delta_{kk'} = \begin{cases} 1 & \text{if route k followed by route k' are both assigned to the same vehicle} \\ 0 & \text{otherwise} \end{cases}$$

The objective is to minimize the number of required vehicles while satisfying all customer demands :

$$\text{minimize } m \quad (1)$$

under constraints (2) to (14)

The constraint (2) guarantees that each customer is visited exactly once:

$$\sum_{\substack{j=0 \\ j\neq i}}^{N} \sum_{k=1}^{K} x_{ijk} = 1 \qquad \forall i \in [1;N] \tag{2}$$

The constraint (3) ensures that each route leaves C_i after visiting him:

$$\sum_{\substack{i=0 \\ i\neq h}}^{N} x_{ihk} = \sum_{\substack{j=0 \\ j\neq h}}^{N} x_{hjk} \qquad \forall h \in [1;N], \forall k \in [1;K] \tag{3}$$

The constraint (4) specifies that there are exactly n routes going out of the depot:

$$\sum_{j=1}^{N} \sum_{k=1}^{K} x_{ijk} = n \qquad \text{for} \quad i = 0 \tag{4}$$

The constraint (5) guarantees the existence of routes

$$x_{ijk} \leq b_k \leq \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} x_{ijk} \qquad \forall k \in [1;K], \forall i, j \in [1;N], i \neq j \tag{5}$$

Constraints (6) and (7) limit respectively the duration and the total volume of each route:

$$\sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} (t_{ij} + s_0^k + s_i) x_{ijk} \leq T_{max} \qquad \forall k \in [1;K] \tag{6}$$

$$\sum_{i=0}^{N} \sum_{\substack{j=1 \\ j\neq i}}^{N} \vartheta_j x_{ijk} \leq C_{max} \qquad \forall k \in [1;K] \tag{7}$$

Constraints (8) to (13) define the time feasibility:

$$t_j^k \geq d_0^k + s_0^k + t_{0j} - (1 - x_{0jk}).l_{0r} \qquad \forall j \in [1;N]; \forall k \in [1;K] \tag{8}$$

$$t_j^k \geq t_i^k + s_i + t_{ij} - (1 - x_{ijk}).l_{0r}$$
$$\forall i \in [1;N]; \forall j \in [0;N]; i \neq j; \forall k \in [1;K] \tag{9}$$

$$d_0^k \leq l_{0\ell} . \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} x_{ijk} \qquad \forall k \in [1;K] \tag{10}$$

$$t_0^k \leq l_{0r} . \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} x_{ijk} \qquad \forall k \in [1;K] \tag{11}$$

$$e_0 \leq d_0^k \leq l_{0\ell} \qquad \forall k \in [1;K] \tag{12}$$

$$e_0 \leq t_0^k \leq l_{0r} \qquad \forall k \in [1;K] \tag{13}$$

The constraint (14) makes sure that $\delta_{kk'}$ is equal to 1 if route k followed by route k' are both assigned to the same vehicle, otherwise ε is an infinitesimal positive number needed to enforce $\delta_{kk'}$ to 1 if $d_0^{k'} = t_0^k$.

$$\frac{d_0^{k'} - t_0^k}{T_{max}} + \varepsilon \leq \delta_{kk'} \leq 1 + \frac{d_0^{k'} - t_0^k}{T_{max}} \qquad \forall k, k' \in [1;K]; k \neq k' \tag{14}$$

Hence, the numbers of required routes and vehicles are respectively given by :

$$n = \sum_{k=1}^{K} b_k \quad \text{and} \quad m = n - \sum_{k=1}^{K} \sum_{\substack{k'=1 \\ k' \neq k}}^{K} \delta_{kk'} \, . \tag{15}$$

2. A Hybrid Evolutionary Approach for the VRPDM

Evolutionary Algorithms (EAs) are randomised parallel search techniques modelled on natural selection. They are based on the principles of genetic algorithms (GAs) introduced in 1975 by Holland [16]. They move from one generation of solutions to another by evolving new solutions through evaluation, selection, recombination and mutation [15]. EAs have been successfully applied to a wide variety of combinatorial optimization problems.

2.1 Coding

A chromosome is represented as an integer string of length N (number of customers). Each gene in the string is the integer node number pre-assigned to the customer. No specific genes are put in the chromosome, neither to mark the depot nor to show the limits of routes, because such delimiters lead to unvalid offspring resulting from reproduction. Consider for example the set of customer demands of figure 1. One chromosome representation is given by figure 2.

C_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ϑ_i (m^3)	6	6	13	5	6	7	6	6	8	7	6	7	1	2	10	12	13	8	8	8

Figure 1: A Set of Customer Bemands.

6	10	12	19	2	7	1	16	4	3	13	11	15	5	8	9	14	17	18	20

Figure 2: Chromosome Represention.

A solution to the problem is obtained by first decoding the chromosome into routes which are then assigned to and scheduled within vehicle plannings.

2.2 Chromosome Route Configurations

To decode the chromosomes into route configurations, the gene values are sequentially inserted into a route. A new route is detected if the total demand of the route exceeds C_{max} or if the route duration exceeds T_{max}. Each route originates and ends at the depot referred to by the node number 0. The sequence of genes in the route is the order in which these customers will be serviced. S_R denotes the set of routes ordered as they appear while decoding the chromosome into route configurations. It is interesting to note that constraints (2) to (9) are satisfied.

As illustration, we decode the chromosome of figure 2. The resulting route configurations are reported in table 1.

Table 1: Chromosome Route Configurations (C_{max}=29m^3 and T_{max}=10 hours).

	S_R	V_k (m^3)	D_k (Hours)
Route 1	0 – 6 – 10 – 12 – 19 – 0	29	5.5
Route 2	0 – 2 – 7 – 1 – 0	18	3
Route 3	0 – 16 – 4 – 0	17	5
Route 4	0 – 3 – 13 – 11 – 0	20	2
Route 5	0 – 15 – 5 – 8 – 0	22	4.5
Route 6	0 – 9 – 14 – 0	10	3.5
Route 7	0 – 17 – 18 – 20 – 0	29	4.5

2.3 Scheduling Vehicle Routes Algorithm

Once route configurations are available, we need to assign several of them to the same vehicle. To this end, we have developed the Scheduling Vehicule Routes Algorithm (SVRA) which exploits the depot double time windows and satisfies constraints (10) to (14). It is based on two main stages. Let D_k be the duration of route k. In the first stage, a vehicle is assigned a long duration route $\left(D_k \geq l_{0r} - l_{0\ell}\right)$ such that its returning back matches l_{0r}. Then, proceeding backwards, the available time is filled with other routes. This procedure is repeated until there is no more long duration route. In the second stage, a vehicle is assigned the first route from the remaining ones such that its beginning date matches $l_{0\ell}$. Then, proceeding backwards, the available time is filled with other routes. If one of these routes has a longer duration than the last one in the vehicle planning, they are permuted. Formally, the SVRA is described as follows:

Begin

Set m = 0.

While it exists a route k such that $[D_k \geq (l_{0r} - l_{0\ell})]$ do SVRA-1 to SVRA-3

SVRA-1: Set the vehicle time counter $V_c = T_{max}$ and $m = m + 1$.

SVRA-2: Assign the first route k in S_R to the vehicle V_m. Set $d_0^k = l_{0r} - D_k$, $V_c = V_c - D_k$ and $S_R = S_R \setminus k$.

SVRA-3: While it exists a route k such that $[D_k \leq V_c]$ do SVRA-3.1 and SVRA-3.2

SVRA-3.1: Assign the first route k in S_R to the vehicle V_m.

SVRA-3.2: Set $d_0^k = d_0^k - D_k$, $V_c = V_c - D_k$ and $S_R = S_R \setminus k$.

While $S_R \neq \varnothing$ do SVRA-4 to SVRA-6

SVRA-4: Set $V_c = T_{max}$ and $m = m + 1$.

SVRA-5: Assign the first route k in S_R to the vehicle V_m. Set $d_0^k = l_{0\ell}$, $V_c = V_c - D_k$, $S_R = S_R \setminus k$ and $d_0^{first} = l_{0\ell}$.

SVRA-6: While it exists a route k' such that $[D_{k'} \leq V_c]$ do SVRA-6.1

SVRA-6.1: Assign the first route k' in S_R to the vehicle V_m. Set $S_R = S_R \setminus k'$ and $V_c = V_c - D_{k'}$. If $D_{k'} > D_k$ then do SVRA-6.2 else do SVRA-6.3.

SVRA-6.2 : Set $d_0^{k'} = l_{0\ell}$ and $d_0^k = d_0^{first} - D_k$. Set $d_0^{first} = d_0^k$ and k = k'.

SVRA-6.3 : Set $d_0^{k'} = d_0^{first} - D_{k'}$. Set $d_0^{first} = d_0^{k'}$.

End

To illustrate this, we apply the SVRA on the route configurations of table 1. The resulting vehicle routes are reported in table 2.

Table 2: SVRA Solution for the Route Configurations of Table 1 (e_0 = 7.00 am, $l_{0\ell}$ = 2.00 pm and l_{0r} = 6.00 pm)

Vehicles	Routes	D_k (Hours)	d_0^k (am)	t_0^k (am)
1	1	5.5	12.30	18.00
	2	3	9.30	12.30
2	3	5	13.00	18.00
	4	2	11.00	13.00
3	5	4.5	13.30	18.00
	6	3.5	10.00	13.30
4	7	4.5	7.00	11.30

2.4 Evaluation

Every chromosome is evaluated according to its fitness function which is the sum of the number of vehicles and the total travel time.

2.5 Reproduction

During the reproduction phase, parent solutions selected from the current population via the Roulette Wheel Selection [15] undergo crossover and mutation to produce new offspring. As crossover and mutation operators are directly responsible for high EA performances they have to be well suited for the problem.

2.5.1 Crossover

The crossover process combines genes of selected parent chromosomes in order to potentially create offspring with better fitness. We make use of the following two crossover operators.

2.5.1.1 Heuristic Crossover

The Heuristic Crossover (HX), successfully used by Tan et al. [31], is concerned with distances between nodes and produces only one offspring from a pair of parents. It could be summarized as follows:

Step-1: Randomly select two parents;

Step-2: Randomly define a cut point at the same location on both parents;

Step-3: Consider the gene situated immediately after the cut point on each parent. If they are different randomly select one, keep it on the corresponding parent and swap it on the other parent.

Step-4: Evolving by means of swapping, transmit the next nearest customer of the two parents to the offspring.

Figure 3 shows two parent chromosomes and their resulting offspring. Let x, y and z be 3 different node numbers assigned to three customers such that x<y<z. In this example, we assume that $d_{xy}<d_{xz}$.

Parent 1	18	20	6	10	12	15	19	1	2	7	11	4	5	3	8	9	16	14	17	13
Parent 2	12	10	1	20	19	9	4	3	17	16	13	15	6	18	14	8	5	7	11	2

(3-a): Illustration of Step 1 and Step 2.

Parent 1	18	20	6	10	12	15	19	1	2	7	11	4	5	3	8	9	16	14	17	13
Parent 2	12	10	6	20	19	9	4	3	17	16	13	15	1	18	14	8	5	7	11	2

(3-b): Illustration of Step 3.

Offspring	6	10	12	9	4	1	2	7	11	15	3	5	8	14	16	18	13	17	19	20
V_k (m^3)	29				29					29			29					29		

Figure 3: Heuristic Crossover illustration.

2.5.1.2 Order Crossover

In [31] the HX is combined with the PMX, an absolute position preserving crossover. However, since crossover operators preserving the relative order, like the Order Crossover (OX), generally provide much better results than those preserving the absolute position [22], we combined the OX with the HX. Figure 4 shows two parent chromosomes and their resulting offspring with the Order crossover. Cut points are located on genes 10 and 13 on both parents.

Parent 1	18	20	6	10	12	15	19	1	2	7	11	4	5	3	8	9	16	14	17	13
V_k (m^3)	23			24			26				17			27			28			
Parent 2	12	10	6	20	19	9	4	3	17	16	13	15	1	18	14	8	5	7	11	2
V_k (m^3)	29				21			26		29				28					12	
Offspring 1	20	19	9	3	17	16	13	15	1	7	11	4	5	18	14	8	2	12	10	6
V_k (m^3)	24			26		29				23				29					14	
Offspring 2	6	10	12	19	2	7	11	4	5	16	13	15	1	3	8	9	14	17	18	20
V_k (m^3)	29				29					29				29				29		

Figure 4: Order Crossover Illustration.

2.5.2 Mutation

Mutation consists in randomly modifying genes of one chromosome to further explore the solution space and to ensure the genetic diversity [15]. We make use of the following four mutation operators.

2.5.2.1 Randomly Permute Two Routes Operator

The randomly Permute two Routes operator (PR) is a new developed operator. It consists in swapping two randomly selected routes of the chromosome. In case only one route is involved, the parent chromosome is transmitted to the next generation. This operator plays a key role because it influences the chromosome decoding not only into route configurations but also into scheduled vehicle routes. To illustrate this, let us permute routes 4 and 5 in the set of routes of table 1 as shown in figure 5. Under the assumption that the new mutated chromosome decoding into route configurations is preserved, we obtain table 3. It is interesting to note that in this illustration one vehicle has been saved.

Offspring	6	10	12	19	2	7	1	16	4	15	5	8	3	13	11	9	14	17	18	20
Routes	1				2			3		4			5			6		7		

Figure 5: Randomly Permute two Routes Illustration.

Table 3: SVRA Solution for the Route Configurations of Figure 5.

Vehicles	Routes	D_k (Hours)	d_0^k (am)	t_0^k (am)
1	1	5.5	12.30	18.00
	2	3	9.30	12.30
2	3	5	13.00	18.00
	4	4.5	8.30	13.00
3	7	4.5	13.30	18.00
	6	3.5	10.00	13.30
	5	2	8.00	10.00

3.5.2.2 Exchange Two Route Sequences Operator

The Exchange two Route Sequences operator (ERS) is a new mutation operator. It consists in swapping two sequences of customers between two randomly selected routes of the chromosome. The ERS is based on three steps summarized as follows:

Step-1: Randomly select a chromosome;

Step-2: Randomly select two routes from the chromosome route configurations;

Step-3: Randomly define a cut point on each route;

Step-4: Exchange the customers sequences situated after the cut point on the two routes;

Note that if the parent chromosome involves only one route or at least one of the selected routes contains only one customer then the ERS transmits the parent to the next generation.

To illustrate the ERS, let us consider routes 4 and 6 (Table 1). The cut points are located on the first gene of each route. After exchanging route sequences, the resulting offspring is given by figure 6. Note that in this illustration the ERS helped save one route.

Offspring	6	10	12	19	2	7	1	16	4	3	14	15	5	8	9	13	11	17	18	20
V_k (m^3)	29				18			17		25			27					29		

Figure 6: Exchange two Route Sequences Illustration.

2.5.2.3 Swap Sequence Operator

The Swap Sequence operator (SS) randomly permutes a sequence of customers in the chromosome [31]. First, two cut points are randomly selected on the chromosome. Then, customers located between the two cut points are randomly swapped. Figure 7 shows one parent chromosome and its resulting offspring. Cut points are located on genes 7 and 17. Note that in this illustration the SS helped maximize the load of routes.

Parent	6	10	12	19	2	7	1	16	4	3	13	11	15	5	8	9	14	17	18	20
V_k (m^3)	29				18			17		20			22			10		29		

Offspring	6	10	12	19	2	7	11	4	5	16	13	15	1	3	8	9	14	17	18	20
V_k (m^3)	29				29					29			29					29		

Figure 7: Swap Sequence illustration.

2.5.2.4 Immigration

In order to continuously explore new regions of the search space, we call for the Immigration operator (I) which introduces new customer configurations totally generated on random basis. Although in [2] this operator is shown to have an important role in preventing premature convergence of the population, it has never been used in routing problems. In this paper, we adapted this operator to the VRPDM we adress.

2.6 Hybrid Evolutionary Algorithm

The Hybrid Evolutionary Algorithm (HEA) is based on the following main steps :

HEA-1: Choose a population size N_c which will be held constant through all the generations. Set the maximum number of generations N_g;

HEA-2: Set k=1 and randomly generate the initial population G_k;

Repeat

HEA-3:. For each chromosome of G_k

HEA-3.1: Decode it into route configurations;

HEA-3.2: Use the SVRA to determine the number of vehicles required;

HEA-3.3: Evaluate its fitness function;

HEA-4: Include the top n_{best} chromosomes of G_k in G_{k+1};

HEA-5: Include in G_{k+1} n_{HX} and n_{OX} offspring created respectively by HX and OX.

HEA-6: Include in G_{k+1} n_{PR}, n_{ERS}, n_{SS} and n_I offspring created respectively by PR, ERS, SS and Immigration.

HEA-7: Set k=k+1.

Until N_g generations are reached.

It should be specified that (16) is satisfied through all generations :

$$N_c = n_{best} + n_{HX} + n_{OX} + n_{PR} + n_{ERS} + n_{SS} + n_I . \quad (16)$$

3 Computational Results

3.1 Problem Sets

The HEA was applied to 70 instances including seven real data sets (see Table 4) which involve 100 customers and seven volume ranges $Dv_{min}v_{max}$ (D0105, D0110, D0115, D0120, D0125, D0129 and D2029). The label $Dv_{min}v_{max}$ means that all volume demands are within $[v_{min}, v_{max}]$. From these real data sets, we randomly generated 63 other instances by variying the number of customers N. Table 5 presents the random instances related customers. Each problem instance is labeled $CNDv_{min}v_{max}$ meaning that the number of customers is equal to N and that the volume range is $Dv_{min}v_{max}$.

Table 4: Customer Coordinates and Requests per Volume Range.

C_i	X_i	Y_i	D0105	D0110	D0115	D0120	D0125	D0129	D202	C_i	X_i	Y_i	D010	D011	D0115	D012	D012	D012	D202
C_0	25	60	0	0	0	0	0	0	0	C_{51}	20	6	2.31	3.02	14.21	18.04	2.95	3.49	21.96
C_1	10	65	1.79	1.67	1.15	10.28	19.46	12.71	22.4	C_{52}	16	7	3.96	1.2	3.8	1.45	23.27	26.58	21.96
C_2	16	70	4.99	9.58	7.3	3.33	4.5	3.26	22.14	C_{53}	17	6	3.93	9.66	9.66	5.55	5.98	23.16	26.58
C_3	18	60	4.2	2.15	1.88	2.66	5.24	15.88	21.66	C_{54}	17	6	4.33	9.93	14.58	8.59	22.68	27.72	22.03
C_4	7	52	2.79	3.53	6.14	16.6	5.72	14.84	25.83	C_{55}	22	6	2.59	6.27	13.33	9.61	21.61	12.11	22.87
C_5	10	40	3.82	9.88	7.8	10.88	11.07	7.38	26.64	C_{56}	20	7	1.15	6.06	3.67	14.76	13.05	9.45	21.8
C_6	18	50	1.2	8.6	3.94	14.53	21.53	4.91	23.63	C_{57}	25	8	4.45	5.5	4.39	12.66	3.51	26.75	26.28
C_7	2	37	1.37	2.69	14.04	4.69	1.03	10.32	24.98	C_{58}	26	7	2.86	9.52	14.52	6.09	1.59	23.64	25.53
C_8	15	18	2.29	8.23	7.17	6	23.33	15.74	20.86	C_{59}	28	7	2.36	6.05	4.55	1.69	18.59	5.02	27.19
C_9	3	24	4.81	1.58	10.45	8.37	4	26.69	28.42	C_{60}	37	8	3.56	6.24	10.01	18.17	1.02	8.75	27.16
C_{10}	10	32	2.68	8.85	3.23	14.77	22.04	16.88	20.37	C_{61}	42	8	3.57	9.93	13.36	18.95	12.77	4.07	22.51
C_{11}	8	36	1.22	9.79	10.73	3.22	11.71	9.05	20.16	C_{62}	50	8	4.3	5.89	1.47	5.41	14.68	11.25	24.07
C_{12}	5	43	1.55	2.88	4.93	10.15	18.37	14.23	22.15	C_{63}	70	8	2.85	2.4	7.79	14.45	21.54	5.66	20.46
C_{13}	0	50	1.68	4.73	3.88	2.38	13.43	25	24.91	C_{64}	80	8	4.31	5.89	3.77	3.82	9.41	13.27	22.19
C_{14}	9	3	4.15	5.36	11.59	14.12	22.97	3.88	20.97	C_{65}	70	7	4.88	6.93	1.79	9.93	15.01	3.9	24.26
C_{15}	8	0	3.15	6.16	12.45	13.06	21.77	11.2	26.11	C_{66}	68	6	1.83	6.52	11.57	10.61	7.84	21.55	22.2
C_{16}	17	10	1.94	5.5	7.08	2.71	12.12	17.62	23.31	C_{67}	52	6	2.42	5.05	5.01	12.75	2.73	26.86	26.28
C_{17}	19	18	3.54	2.19	12.56	4.88	24.9	7.07	20.32	C_{68}	50	6	1.74	9.38	1.55	12.66	11.62	26.98	26.93
C_{18}	27	13	3.77	2.25	4.74	13.19	11.48	7.84	26.02	C_{69}	29	7	4.87	4.8	3.54	10.04	7.69	13.53	28.49
C_{19}	10	11	3.35	7.52	2.17	10.5	18.76	14.31	24.35	C_{70}	31	7	1.97	7.84	14.75	8.82	12.76	6.62	24.33
C_{20}	31	13	2.18	2.47	8.16	6.41	12.4	13.47	28.83	C_{71}	59	7	1.64	2.43	8.06	1.36	12.66	13.33	24.61
C_{21}	37	15	2.72	6.02	7.79	11.12	3.86	19.65	24.47	C_{72}	43	8	1.24	5.34	6.82	9.5	10.88	22.81	26.82
C_{22}	42	9	1.34	8.72	1.37	8.62	9.5	4.67	22.32	C_{73}	58	7	4.37	7.13	8.4	15	24.57	15.65	26.76
C_{23}	50	0	2.41	7.41	7.09	19.16	9.65	21.01	24.54	C_{74}	22	5	3.94	1.01	11.01	19.27	3.95	21.49	23.39
C_{24}	48	20	4.84	6.69	2.36	15.52	5.79	15.12	26.79	C_{75}	23	5	4.53	5.31	6.26	18.19	21.79	10.48	28.5
C_{25}	36	16	2.25	6.55	2.27	1.83	7.51	13.47	22.51	C_{76}	25	4	1.89	6.41	9.67	3.32	20.16	13.28	21.38
C_{26}	26	10	2.05	2.82	5.12	7.94	20.11	16.66	25.12	C_{77}	12	0	2.71	3.89	5.03	17.84	2.87	27.8	25.43
C_{27}	12	55	2.31	9.12	1.12	1.25	18.84	8.15	20.13	C_{78}	21	4	1.05	8.65	5.08	11.85	1.72	10.31	22.95
C_{28}	7	38	4.83	2.22	4.4	1.35	16.58	7.27	23.12	C_{79}	19	3	2.75	6.39	5.81	15.97	16.83	10.98	20.54
C_{29}	21	30	2.65	5.46	14.32	10.83	20.54	8.75	27.41	C_{80}	17	3	4.21	9.06	7.12	12.28	7.74	12.88	23.64
C_{30}	1	62	4.46	8.34	10.49	3.89	6.33	3.6	25.12	C_{81}	18	4	3.1	7.52	11.26	12.33	13.23	1.51	20.28
C_{31}	3	65	2.7	6.19	1.62	10.95	12.86	17.31	24.93	C_{82}	0	3	3.79	1.37	12.01	13.73	23.74	8.12	23.5
C_{32}	0	67	2.74	9.42	3.21	4.99	5.34	24.66	21.48	C_{83}	15	5	1.99	5.93	14.07	11.34	16.32	22.02	28.23
C_{33}	9	71	1.04	6.14	13.93	6.87	2.6	3.98	28.44	C_{84}	13	5	4.67	7.1	3.27	17.16	12.51	4.32	26.35
C_{34}	9	77	3.82	6	9.15	12.52	4.92	8.72	22.33	C_{85}	17	5	2.2	4.91	1.98	2.71	9.01	13.34	27.28
C_{35}	12	83	1.27	4.97	9.54	9.4	7.9	15.36	25.74	C_{86}	8	4	1.38	4.27	5.72	18.06	16.33	5.57	23.67
C_{36}	9	87	4.91	3.6	14.15	6.26	20.35	2.11	28.01	C_{87}	0	5	3.64	2.88	1.82	16.08	8.08	5.12	24.28
C_{37}	0	95	1.03	8.56	10.08	16.18	8.76	19.38	25.22	C_{88}	14	4	3.16	7.77	14.39	16.55	7.1	3.52	21.87
C_{38}	0	100	3.05	7.31	9.04	19.11	6.94	1.96	28.18	C_{89}	0	1	4.48	4.69	14.49	7.54	6.57	20.06	22.85
C_{39}	10	105	1.7	2.12	7.94	12.2	18.6	7.91	22.67	C_{90}	23	1	3.79	5.12	2.41	6.58	7.9	25.08	26.68
C_{40}	17	120	4.66	9.13	7.75	13.1	10.41	25.02	28.1	C_{91}	24	4	4.81	9.46	7.31	17.37	6.86	5	24.4
C_{41}	20	73	4.07	2.52	6.58	9.86	15.3	24.68	23.54	C_{92}	27	0	2.66	6.12	10.24	9.5	16.09	5.06	28.38
C_{42}	22	68	3.12	1.09	14.21	14.08	5.99	20.29	27.5	C_{93}	22	2	1.55	9.82	2	17.04	24.97	6.6	23.65
C_{43}	25	66	4.09	5.04	3.62	4.66	2.66	22.66	23.35	C_{94}	30	1	2.9	6.36	14.38	7.38	9.71	21.36	25.63
C_{44}	15	90	1.55	7.94	1.02	15.83	9.71	7.66	27.7	C_{95}	26	1	2.27	2.19	11.1	3.4	7.25	8.61	28.11
C_{45}	17	85	3.22	7.06	10.73	17.45	13.99	11.72	20.86	C_{96}	20	2	4.06	9.22	5.23	11.98	4.99	24.25	25.92
C_{46}	20	70	2.15	6.01	14.78	6.52	14.84	28.3	24.57	C_{97}	12	1	4.55	7.56	2.79	10.93	11.59	14.24	28.72
C_{47}	11	76	1.98	1.5	3.98	4.33	11.15	21.61	22.85	C_{98}	13	2	2.8	4.43	13.02	7.52	8.16	19.14	21.53
C_{48}	13	70	3.32	5.86	5.34	16.78	21.7	7.1	25.63	C_{99}	10	1	1.03	5.06	4.06	19.41	15.53	22.02	22.98
C_{49}	14	64	2.92	7.03	14.62	9.35	14.12	22.1	22.47	C_{10}	40	0	3.42	2.41	13.04	15.2	18.94	22.24	26.38
C_{50}	14	62	1.94	2.47	4.73	8.82	5.23	17.15	26.07										

Distances and volumes are given respectively in Km and m^3. X_i, Y_i : Cartesian coordinates of customer i.

Table 5: Customers of the Random Instances.

Problems	Customers
C10D0105, C10D0110, C10D0115, C10D0120, C10D0125, C10D0129 C10D2029	$C_{72}; C_{70}; C_{39}; C_{18}; C_{68}; C_{59}; C_{51}; C_{38}; C_{46}; C_{89};$
C20D0105, C20D0110, C20D0115, C20D0120, C20D0125, C20D0129 C20D2029	$C_{48};C_{4};C_{53};C_{50};C_{70};C_{78};C_{88};C_{33};C_{61};C_{21};C_{63};C_{38};C_{75};C_{62};$ $C_{77};C_{34};C_{22};C_{37};C_{31};C_{90};$
C30D0105, C30D0110, C30D0115, C30D0120, C30D0125, C30D0129 C30D2029	$C_{1};C_{61};C_{73};C_{79};C_{55};C_{32};C_{21};C_{83};C_{92};C_{9};C_{50};C_{51};C_{52};C_{20};$ $C_{64};C_{26};C_{62};C_{43};C_{27};C_{57};C_{7};C_{91};C_{14};C_{76};C_{59};C_{39};C_{13};C_{36};$ $C_{75};C_{96};$
C40D0105, C40D0110, C40D0115, C40D0120, C40D0125, C40D0129 C40D2029	$C_{82};C_{7};C_{57};C_{69};C_{39};C_{60};C_{2};C_{26};C_{28};C_{89};C_{41};C_{70};C_{45};C_{65};C_{59};$ $C_{49};C_{86};C_{40};C_{90};C_{14};C_{99};C_{68};C_{10};C_{29};C_{55};C_{30};C_{20};C_{88};C_{78};$ $C_{35};C_{46};C_{52};C_{96};C_{95};C_{18};C_{9};C_{97};C_{23};C_{81};C_{15};$
C50D0105, C50D0110, C50D0115, C50D0120, C50D0125, C50D0129 C50D2029	$C_{76};C_{6};C_{21};C_{45};C_{39};C_{54};C_{25};C_{29};C_{30};C_{44};C_{72};C_{38};C_{51};C_{68};$ $C_{15};C_{33};C_{11};C_{24};C_{59};C_{64};C_{12};C_{31};C_{61};C_{62};C_{58};C_{40};C_{52};C_{27};$ $C_{97};C_{88};C_{32};C_{60};C_{71};C_{84};C_{66};C_{17};C_{67};C_{90};C_{75};C_{98};C_{70};C_{16};$ $C_{37};C_{63};C_{43};C_{22};C_{78};C_{34};C_{89};C_{18}$
C60D0105, C60D0110, C60D0115, C60D0120, C60D0125, C60D0129 C60D2029	$C_{13};C_{96};C_{2};C_{70};C_{34};C_{74};C_{84};C_{17};C_{75};C_{41};C_{61};$C44;$C_{76};$C1; $C_{57};C_{86};C_{51};C_{14};C_{59};C_{77};C_{63};C_{4};C_{30};C_{6};C_{91};C_{68};C_{85};C_{10};$ $C_{18};C_{93};C_{28};C_{58};C_{33};C_{24};C_{69};C_{53};C_{48};C_{35};C_{87};C_{43};C_{82};C_{22};$ $C_{39};C_{88};C_{32};C_{15};C_{12};C_{11};C_{49};C_{97};C_{25};C_{60};C_{99};C_{40};C_{66};$ $C_{45};C_{54};C_{46};C_{89};C_{19};$
C70D0105, C70D0110, C70D0115, C70D0120, C70D0125, C70D0129 C70D2029	$C_{1};C_{2};C_{3};C_{4};C_{5};C_{7};C_{9};C_{10};C_{11};C_{12};C_{16};C_{17};C_{18};C_{20};C_{21};C_{22};$ $C_{23};C_{24};C_{26};C_{27};C_{28};C_{30};C_{31};C_{32};C_{34};C_{35};C_{38};C_{41};C_{42};C_{43};$ $C_{45};C_{46};C_{47};C_{48};C_{49};C_{50};C_{51};C_{53};C_{55};C_{56};C_{57};C_{58};C_{59};C_{60};$ $C_{61};C_{63};C_{64};C_{65};C_{66};C_{67};C_{68};C_{70};C_{71};C_{72};C_{73};C_{74};C_{75};C_{79};$ $C_{80};C_{81};$
C80D0105, C80D0110, C80D0115, C80D0120, C80D0125, C80D0129 C80D2029	$C_{1};C_{2};C_{3};C_{4};C_{5};C_{7};C_{8};C_{9};C_{10};C_{11};C_{12};C_{13};C_{14};C_{15};C_{16};C_{17};$ $C_{18};C_{20};C_{22};C_{23};C_{24};C_{27};C_{28};C_{29};C_{30};C_{31};C_{32};C_{33};C_{34};C_{37};$ $C_{38};C_{39};C_{40};C_{41};C_{42};C_{44};C_{45};C_{46};C_{48};C_{49};C_{50};C_{52};C_{53};C_{54};$ $C_{56};C_{57};C_{58};C_{59};C_{60};C_{61};C_{64};C_{65};C_{67};C_{68};C_{69};C_{70};C_{71};C_{72};$ $C_{73};C_{75};C_{77};C_{78};C_{79};$C80;$C_{81};C_{84};C_{85};C_{86};C_{87};C_{88};C_{89};C_{91};$ $C_{92};C_{94};C_{95};C_{96};C_{97};C_{98};C_{99};C_{100};$
C90D0105, C90D0110, C90D0115, C90D0120, C90D0125, C90D0129 C90D2029	$C_{1};C_{2};C_{3};C_{4};C_{5};C_{6};C_{7};C_{8};C_{9};C_{10};C_{12};C_{13};C_{14};C_{15};C_{16};C_{17};$ $C_{19};C_{20};C_{21};C_{22};C_{23};C_{24};C_{25};C_{26};C_{27};C_{28};C_{29};C_{30};C_{31};C_{32};$ $C_{33};C_{34};C_{35};C_{36};C_{38};C_{39};C_{40};C_{42};C_{43};C_{44};C_{45};C_{47};C_{48};C_{49};$ $C_{50};C_{51};C_{52};C_{53};C_{54};C_{55};C_{56};C_{57};C_{58};C_{59};C_{60};C_{62};C_{64};C_{66};$ $C_{67};C_{68};C_{69};C_{70};C_{71};C_{72};C_{73};C_{74};C_{77};C_{78};C_{79};C_{80};C_{81};C_{82};$ $C_{83};C_{84};C_{85};C_{86};C_{87};C_{88};C_{89};C_{90};C_{91};C_{92};C_{94};C_{95};C_{96};C_{97};$ $C_{98};C_{99};C_{100};$

3.2 Evolutionary Parameters

The HEA was implemented on a Pentium-III PC, 500MHz using the C-Language. The following parameter values were experimentally found to be good and robust for the tested problems:

$$N_g = 20000 \quad \text{and} \quad N_c = 100 \,. \tag{17}$$

$$\begin{cases} n_{best} = n_I = \dfrac{N_c}{20}; n_{HX} = \dfrac{2N_c}{5}; n_{OX} = \dfrac{N_c}{5}; \\ n_{PR} = n_{ERS} = n_{SS} = \dfrac{N_c}{10}; \end{cases} \tag{18}$$

Although earlier research has shown that excessive mutation rates lead to premature convergence [31], often resulting in undesirable local optimal solutions, we noticed that, in our case, small mutation rates do not produce good results.

3.3 Simulation Analysis

For each problem instance, the HEA was run 20 times (each with a different seed). Each table from 6 to 12 presents the results of one volume range and ten instances varying in customer size. A t-test was designed on the hypothesis that the solutions to obtain have relative mean average deviations of the fitness values $RMAD_F$ and the number of vehicles $RMAD_m$ less than 5%. For each instance, we have also mentioned a lower bound on the number n* of routes needed to service all customers which is the smallest integer not smaller than total demand divided by the capacity of the vehicles.

It is interesting to note that in all cases, the HEA found the same number of vehicles over the 20 runs, thus yielding an $RMAD_m$ of 0% and a maximum $RMAD_F$ of 1.59%. The t-test hypothesis is satisfied which illustrates the robustness of the developed algorithm. Furthermore, as the number of routes increases according to volume ranges and/or customer sizes, the number of vehicles increases slightly but not necessarily, this shows the ability of the HEA to employ available resources effectively. An other observation stemming from tables 6 to 12 is that if the number of customers is few, the HEA quickly finds feasible solutions in a short period of time. As the number of customers gets more important, the computation time increases reasonably.

Table 6: D0105 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D0105	Min	7.59	1	1	263.77	0	4		
	Med	7.59	1	1	263.77	0	49	0	0
n* = 1	Max	7.59	1	1	263.77	1	245		
C20D0105	Min	11.18	2	2	367.19	0	115		
	Med	11.18	2	2	367.19	12	2646	0.08	0
n* = 2	Max	11.27	2	2	370.74	3	593		
C30D0105	Min	13.70	2	4	467.84	49	8315		
	Med	13.72	2	4	468.69	6	891	0.29	0
n* = 4	Max	13.82	2	4	472.8	51	8651		
C40D0105	Min	18.64	3	5	625.69	30	4477		
	Med	18.89	3	5	635.59	66	9476	0.65	0
n* = 5	Max	19.07	3	5	642.69	109	1566		
C50D0105	Min	19.33	3	5	653.18	151	1911		
	Med	19.92	3	6	676.93	52	6573	0.87	0
n* = 5	Max	20.37	3	5	694.72	147	1855		
C60D0105	Min	20.96	3	7	718.39	69	7818		
	Med	21.51	3	7	740.21	57	6903	0.98	0
n* = 7	Max	22.28	3	7	771.12	18	2496		
C70D0105	Min	23.02	3	8	800.98	66	6492		
	Med	23.45	3	8	818.06	199	1987	1.56	0
n* = 8	Max	24.06	3	8	842.29	102	1020		
C80D0105	Min	27.51	4	9	940.44	168	1474		
	Med	28.07	4	9	962.84	151	1300	1.43	0
n* = 9	Max	28.92	4	9	996.68	183	1581		
C90D0105	Min	29.32	4	10	1012.75	223	1755		
	Med	29.64	4	10	1025.69	241	1933	1.59	0
n* = 10	Max	30.55	4	10	1062.2	150	1178		
C100D0105	Min	31.95	4	11	1117.98	279	1988		
	Med	32.7	4	11	1147.89	271	1919	0.96	0
n* = 11	Max	33.14	4	11	1165.73	216	1527		

Table 7: D0110 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D0110	Min	8.54	1	2	301.51	0	6		
	Med	8.54	1	2	301.51	0	15	0	0
n* = 2	Max	8.54	1	2	301.51	0	40		
C20D0110	Min	13.61	2	5	464.25	1	62		
	Med	13.61	2	5	464.25	1	161	0.05	0
n* = 5	Max	13.65	2	5	466.07	0	22		
C30D0110	Min	16.38	2	6	575.35	2	354		
	Med	16.40	2	6	576.18	39	6037	0.11	0
n* = 6	Max	16.43	2	6	577.25	23	3881		
C40D0110	Min	22.82	3	9	792.83	107	13932		
	Med	22.97	3	8	798.75	2	337	0.54	0
n* = 8	Max	23.22	3	8	808.61	3	467		
C50D0110	Min	28.74	4	12	989.55	95	10306		
	Med	29.41	4	12	1016.55	74	8139	0.74	0
n* = 11	Max	29.82	4	12	1032.96	60	6580		
C60D0110	Min	30.95	4	14	1078.11	114	10780		
	Med	31.07	4	14	1082.69	136	12784	0.19	0
n* = 13	Max	31.19	4	14	1087.49	53	5155		
C70D0110	Min	34.21	5	15	1168.57	150	13104		
	Med	34.74	5	15	1189.70	68	5896	0.7	0
n* = 14	Max	35.05	5	15	1201.99	195	16953		
C80D0110	Min	41.31	6	18	1412.25	223	17111		
	Med	41.76	6	18	1430.53	214	16398	0.85	0
n* = 17	Max	42.56	6	18	1462.33	238	18052		
C90D0110	Min	44.6	6	19	1543.86	198	13472		
	Med	44.92	6	19	1557.00	202	13607	1.23	0
n* = 18	Max	46.02	6	19	1600.94	96	6493		
C100D0110	Min	48.66	7	21	1666.29	314	19231		
	Med	48.94	7	21	1701.75	205	12612	0.75	0
n* = 20	Max	48.94	7	21	1677.61	314	19231		

Table 8: D0115 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D0115	Min	10.42	2	4	336.79	0	3		
	Med	10.42	2	4	336.79	1	308	0	0
n* = 4	Max	10.42	2	4	336.79	15	3650		
C20D0115	Min	14.33	2	6	493.25	0	42		
	Med	14.33	2	6	493.25	1	98	0.07	0
n* = 6	Max	14.43	2	6	497.20	4	646		
C30D0115	Min	20.63	3	9	705.25	2	274		
	Med	20.63	3	9	705.25	13	1825	0.24	0
n* = 8	Max	21.16	3	9	726.43	70	10454		
C40D0115	Min	30.50	4	12	1060.07	1	158		
	Med	30.79	4	12	1071.71	101	11595	0.53	0
n* = 12	Max	31.10	4	12	1083.95	5	617		
C50D0115	Min	34.00	5	14	1159.87	106	11015		
	Med	34.37	5	14	1174.86	119	12486	0.7	0
n* = 14	Max	34.86	5	14	1194.40	39	4138		
C60D0115	Min	33.92	5	17	1156.73	184	16456		
	Med	34.42	5	17	1176.92	40	3584	0.83	0
n* = 16	Max	35.02	5	17	1200.86	191	16081		
C70D0115	Min	40.52	6	20	1380.83	149	11449		
	Med	40.98	6	19	1399.03	182	14132	0.7	0
n* = 18	Max	41.41	6	19	1416.59	17	1378		
C80D0115	Min	50.87	7	22	1754.85	277	18479		
	Med	51.26	7	23	1770.45	217	14464	0.66	0
n* = 21	Max	51.95	7	23	1798.07	123	8495		
C90D0115	Min	56.70	8	25	1948.04	203	12181		
	Med	57.36	8	26	1974.27	76	4705	0.79	0
n* = 24	Max	58.35	8	25	2013.88	322	19872		
C100D0115	Min	59.64	8	28	2065.79	309	16216		
	Med	60.11	8	28	2084.25	330	17076	0.37	0
n* = 27	Max	60.30	8	28	2091.81	73	3931		

Table 9: D0120 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D0120	Min	12.16	2	5	406.23	0	2		
	Med	12.16	2	5	406.23	0	11	0	0
n* = 4	Max	12.16	2	5	406.23	0	49		
C20D0120	Min	21.67	3	10	746.75	0	25		
	Med	21.71	3	10	748.47	6	964	0.07	0
n* = 9	Max	21.71	3	10	748.60	12	1887		
C30D0120	Min	24.67	4	11	826.70	3	289		
	Med	24.67	4	11	826.70	62	8028	0.3	0
n* = 10	Max	25.46	4	11	858.23	54	7121		
C40D0120	Min	37.46	5	16	1298.57	3	285		
	Med	37.63	5	16	1305.09	56	6193	0.57	0
n* = 15	Max	38.31	5	16	1332.51	72	7679		
C50D0120	Min	42.39	6	19	1455.65	171	1577		
	Med	42.95	6	20	1478.10	32	2878	0.7	0
n* = 18	Max	43.46	6	19	1498.39	66	5991		
C60D0120	Min	51.45	7	27	1778.01	128	7648		
	Med	52.14	7	27	1805.56	174	1063	0.69	0
n* = 24	Max	52.63	7	27	1825.56	27	1730		
C70D0120	Min	54.32	8	28	1852.89	184	1008		
	Med	54.62	8	28	1864.85	169	7576	0.66	0
n* = 26	Max	55.58	8	28	1903.05	275	1830		
C80D0120	Min	67.78	9	32	2351.34	225	1347		
	Med	68.37	9	33	2374.8	276	1473	0.42	0
n* = 30	Max	68.84	9	32	2393.48	271	1550		
C90D0120	Min	72.24	10	35	2489.71	377	1935		
	Med	72.63	10	35	2505.2	125	5773	0.78	0
n* = 33	Max	73.85	10	35	2554.00	10.	5195		
C100D012	Min	80.9	11	40	2796.07	145	6526		
	Med	81.77	11	40	2830.93	127	5451	0.34	0
n* = 36	Max	82.33	11	39	2853.02	46	2048		

Table 10: D0125 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	RMAD (%)	$RMAD_m$ (%)
C10D0125	Min	11.74	2	5	389.80	0	1		
	Med	11.74	2	5	389.80	0	15	0	0
n* = 4	Max	11.74	2	5	389.80	1	265		
C20D0125	Min	17.27	3	8	570.76	0	105		
	Med	17.27	3	8	570.76	1	151	0.07	0
n* = 7	Max	17.34	3	8	573.54	0	34		
C30D0125	Min	31.83	5	17	1073.40	1	77		
	Med	31.84	5	17	1073.68	4	300	0.12	0
n* = 14	Max	32.23	5	17	1089.05	26	3390		
C40D0125	Min	44.01	6	19	1520.27	51	4508		
	Med	44.78	6	20	1551.18	65	6499	0.51	0
n* = 18	Max	44.95	6	20	1557.92	25	2458		
C50D0125	Min	47.47	7	23	1618.73	125	7703		
	Med	47.77	7	23	1630.93	217	17082	0.57	0
n* = 20	Max	48.37	7	23	1654.89	45	2692		
C60D0125	Min	56.91	8	28	1956.31	86	5448		
	Med	57.36	8	28	1974.38	280	19043	0.59	0
n* = 26	Max	58.09	8	28	2003.68	10	749		
C70D0125	Min	61.39	9	31	2095.65	13	769		
	Med	61.97	9	31	2118.87	331	19018	0.65	0
n* = 28	Max	62.86	9	31	2154.24	16	958		
C80D0125	Min	73.26	10	36	2530.43	83	4615		
	Med	73.79	10	36	2551.41	227	10962	0.5	0
n* = 33	Max	75.02	10	36	2600.77	332	16681		
C90D0125	Min	82.70	12	40	2828.18	413	17846		
	Med	83.26	12	39	2850.33	367	16283	0.49	0
n* = 37	Max	83.77	12	39	2870.71	485	16217		
C100D012	Min	94.31	13	47	3252.53	450	17544		
	Med	94.70	13	46	3268.18	551	19647	0.41	0
n* = 42	Max	95.61	13	47	3304.29	500	18466		

Table 11: D0129 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D0129	Min	11.28	2	5	371.34	0	4		
	Med	11.28	2	5	371.34	15	3068	0	0
n* = 5	Max	11.28	2	5	371.34	105	1685		
C20D0129	Min	20.86	3	9	714.56	0	13		
	Med	20.86	3	9	714.56	1	75	0.09	0
n* = 9	Max	20.97	3	10	718.87	1	193		
C30D0129	Min	32.49	5	16	1099.56	11	1230		
	Med	32.49	5	16	1099.56	82	4749	0.22	0
n* = 15	Max	32.89	5	16	1115.51	119	1071		
C40D0129	Min	48.74	7	21	1669.53	4	329		
	Med	48.82	7	21	1672.89	91	5719	0.52	0
n* = 20	Max	49.59	7	21	1703.6	3	151		
C50D0129	Min	54.24	8	25	1849.80	235	1934		
	Med	54.77	8	25	1870.64	30	1582	0.53	0
n* = 23	Max	55.32	8	25	1892.90	72	6119		
C60D0129	Min	56.3	8	30	1931.99	118	8041		
	Med	56.85	8	31	1954.05	24	1366	0.54	0
n* = 28	Max	57.65	8	30	1985.86	198	1201		
C70D0129	Min	66.58	10	36	2263.35	205	8723		
	Med	67.39	10	36	2295.65	237	9718	0.6	0
n* = 33	Max	67.58	10	36	2303.06	241	1265		
C80D0129	Min	84.99	12	42	2919.42	540	1998		
	Med	85.5	12	42	2940.15	358	1790	0.49	0
n* = 39	Max	86.38	12	42	2975.4	55	2266		
C90D0129	Min	98.97	14	49	3398.75	473	1439		
	Med	99.15	14	49	3406.06	326	1409	0.37	0
n* = 45	Max	99.98	14	49	3439.23	147	4024		
C100D012	Min	104.14	15	54	3565.54	562	1583		
	Med	104.92	15	54	3596.63	509	1843	0.47	0
n* = 49	Max	105.7	15	53	3627.97	459	1198		

Table 12: D2029 HEA Solutions.

Problem		Fitness	m	n	Total Distance	CPU (s)	N_{gen}	$RMAD_F$ (%)	$RMAD_m$ (%)
C10D2029	Min	18.28	3	10	611.36	0	0		
	Med	18.28	3	10	611.36	0	0	0	0
n* = 10	Max	18.28	3	10	611.36	0	0		
C20D2029	Min	34.70	5	20	1188.12	0	0		
	Med	34.70	5	20	1188.12	0	4	0	0
n* = 20	Max	34.70	5	20	1188.12	1	16		
C30D2029	Min	49.46	7	30	1698.52	0	3		
	Med	49.46	7	30	1698.52	83	3163	0	0
n* = 30	Max	49.46	7	30	1698.52	503	17184		
C40D2029	Min	77.13	11	40	2645.02	0	0		
	Med	77.13	11	40	2645.02	87	3993	0	0
n* = 40	Max	77.13	11	40	2645.02	331	9943		
C50D2029	Min	91.65	13	50	3146.10	3	51		
	Med	91.65	13	50	3146.10	229	6881	0	0
n* = 50	Max	91.65	13	50	3146.10	739	16638		
C60D2029	Min	99.88	15	60	3395.34	6	182		
	Med	99.88	15	60	3395.34	216	5567	0	0
n* = 60	Max	99.88	15	60	3395.34	773	18977		
C70D2029	Min	116.76	17	70	3990.44	2	37		
	Med	116.76	17	70	3990.44	191	6338	0	0
n* = 70	Max	116.76	17	70	3990.44	984	15049		
C80D2029	Min	142.89	20	80	4915.72	8	282		
	Med	142.89	20	80	4915.72	278	6863	0	0
n* = 80	Max	142.89	20	80	4915.72	663	17213		
C90D2029	Min	160.39	23	90	5495.56	10	143		
	Med	160.39	23	90	5495.56	201	4699	0	0
n* = 90	Max	160.39	23	90	5495.56	687	15952		
C100D2029	Min	176.79	25	100	6071.72	21	277		
	Med	176.79	25	100	6071.72	344	5549	0	0
n* = 100	Max	176.79	25	100	6071.72	930	13551		

Min : minimum fitness;

Med : medium fitness;

Max : maximum fitness;

CPU : time elapsed in seconds to find the solution;

N_{gen} : number of generations necessary to find the solution;

n* : lower bound on the number of routes;

m : number of required vehicles;

n : number of required routes;

$RMAD_F$: relative Mean Average Deviation of the fitness values over the 20 runs;

$RMAD_m$: relative Mean Average Deviation of the number of vehicles over the 20 runs;

$C_{max}=29m^3$, $T_{max}=10$ hours, $e_0 = 7.00$ am, $l_{0\ell} = 2.00$ pm, $l_{0r} = 6.00$ pm, $s_i = \frac{\alpha \vartheta_i}{C_{max}}$, $s_0^k = \frac{\alpha V_k}{C_{max}}$, $\alpha=0.5$ hours;

The speed of vehicles is assumed to be equal to 40km/h.

Besides, to load the vehicles efficiently, the number of routes should be as small as possible. Ideally, it would be equal to n*. In this context, the HEA finds the lower bound on the number of routes for 30 problems over 70. However, the number of routes for 22, 7, 8, 2 and 1 problems has been upper than the lower bound respectively by 1, 2, 3, 4 and 5 routes (the difference is computed with respect to the best number of routes found). This is an acceptable result since having a smaller number of routes is indeed appealing, yet it does not necessarily yield a better solution as it is for example the case of the best solutions found for problems C100D0120 and C40D0110. In fact, in some cases, small routes are more easily inserted into vehicle plannings than big routes for which new vehicles would be needed. Therefore, for each instance, a suitable trade off between short and long duration routes should be found. This is best accomplished by the HEA since on the one hand the evolutionary approach diversifies the solution space by creating different route configurations and on the other hand, the scheduling vehicle routes algorithm groups them within a given vehicle planning.

4. Conclusion

In this paper, a practical new variant of the VRP is adressed. The problem is considered as a Vehicle Routing Problem with Double time windows for the depot and Multiple use of vehicles. We solved the problem by an evolutionary approach hybrided with a scheduling algorithm. The evolutionary approach involves a new combination of crossover operators and makes use of new and adapted mutation operators. This is to smartly difersify the solution space by creating different route configurations. The scheduling vehicle routes algorithm efficiently exploits the depot double time windows and groups several routes within a given vehicle planning.

The computational results show that in all instances considered, the hybrid evolutionary optimisation algorithm yields satisfactory solutions in a reasonable amount of computation time. Its ability to deal with practical size decision problems and to fit customer characteristics changes is also shown.

As further research, we think about extending the suggested approach to more complex problems such as VRPDMs involving customer time windows with various widths and densities, a heterogeneous fleet of vehicles, ...

REFERENCES

1. ARONSON, L.D., **Algorithms for vehicle routing: A survey**, Report 96-21, Faculty of Technical Mathematics and Computer Science, Delft, The Netherlands, 1996.
2. BEAN, J.C., **Genetic algorithms and random keys for sequencing and optimization**, ORSA J. Comput., Vol. 6, pp. 154-160, 1994.
3. BERGER, J., MARTIN, S., BEGIN, R., **A hybrid genetic algorithm for the vehicle routing problem with time windows**, Proceedings of the 12th Bienneal of the Canadian Society for Computational Study of Intelligence, pp. 114-127, Springer-Verlag, Berlin, 1997.
4. BRANDAO, J., MERCER, A., **A tabu search for the multi-trip vehicle routing and scheduling problem**, European Journal of Operational Research, Vol. 100, pp. 180-191, 1997.
5. BRANDAO, J., MERCER, A., **The multi-trip vehicle routing problem**, Journal of the Operational research society, Vol. 49, pp. 799-805, 1998.
6. BRÄYSY, O., BERGER, J., BARKAOUI, M., **A new hybrid evolutionary algorithm for the vehicle routing problem with time windows**, Route 2000 workshop, Skodsborg, Denmark, 2000.
7. BRÄYSY, O., **Genetic Algorithms for the Vehicle Routing Problem with Time Windows**, Arpakannus 1/2001, Special issue on Bioinformatics and Genetic Algorithms, 2001.
8. CORDEAU, J.-F., GENDREAU, M., LAPORTE, G., POTVIN, J.-Y., SEMET, F., **A guide to vehicle routing heuristics**, Journal of the Operational Research Society Vol. 53, pp. 512-522, 2002.
9. CORDEAU, J.F., LAPORTE, G., MERCIER, A., **Unified Tabu search heuristic for vehicle routing problems with time windows**, Publication CRT-2000-03, University of Montreal, Canada, 2000.
10. CRAINIC, T.G., LAPORTE, G., **Fleet management and logistics**, Center for research on transportation, Kluwer Academic Publishers, 1998.
11. DANTZIG, G.B., RAMSER, J.H., **The truck dispatching problem**, Management Science, Vol. 6, pp. 80-91, 1959.

12. DESROCHERS, M., SOLOMON, M., **A new optimization algorithm for vehicle routing problems with time windows**, Operations Research, Vol. 40, 1992.

13. FAGERHOLT, K., **Optimal fleet design in a ship routing problem**, International Transactions in Operational research, Vol 6, pp. 453-464, 1999.

14. GAMBARDELLA, L. M., TAILLARD, E., AGAZZI, G., **MACS-VRPTW : a multiple ant colony system for vehicle routing problems with time windows**, in New Ideas in Optimization, pp. 63-76, McGraw-Hill, London, 1999.

15. GOLDBERG, D.E., **Genetic Algorithms in search**, optimization and machine learning, Addison Wesley, Reading, MA 1989.

16. HOLLAND, J.H., **Adaptation in natural and artificial systems**, University of Michigan Press, Michigan, 1975.

17. HOMBERGER, J., GEHRING, H., **Two evolutionary meta heuristics for the vehicle routing problem with time windows**, INFORMS Journal on Computing, Vol. 37, N° 3, pp. 297-318, 1999.

18. KALLEHAUGE, B., LARSEN, J., MADSEN, O.B.G., **Lagrangean Duality applied on vehicle routing with time windows : Experimental results**, Informatics and Mathematical Modeling, Technical report IMM-TR-2001-9, Technical University of Denmark, Denmark, 2001.

19. KOHL, N., MADSEN, O.B.G., **An optimization algorithm for the vehicle routing problem with time windows based on Lagrangean Relaxation**, Operations Research, Vol. 45, pp. 395-406, 1997.

20. KOLEN, A.W.J., KAN, A.H.G.R., Trienekens, H.W.J.M., **Vehicle routing with time windows**, Operations Research, Vol. 35, pp. 266-274, 1987.

21. LAPORTE, G., OSMAN, I.H, **Routing problems: a bibliography**, Annals of Operations Research, Vol. 61, pp. 227-262, 1995.

22. OLIVER, I.M., SMITH, D.J., HOLLAND, J.R.C., **A study of permutation crossover operators on the Traveling Salesman Problem**, In Proceedings of the Second Int. Conf. On Genetic Algorithms (ICGA'87), Massachusetts Institute of Technology, Cambridge, MA, pp. 224-230, 1987.

23. OSMAN, I.H., **Meta-strategy simulated annealing and tabu search algorithms for the vehicle routing problem**, Annuals of Operations Research, Vol. 41, pp. 421-452, 1993.

24. OSMAN, I.H., LAPORTE, G., **Metaheuristics : A bibliography**, Annals of Operations Research, Vol. 63, pp. 513-523, 1996.

25. POTVIN, J.Y., BENGIO, S., **The vehicle routing problem with time windows. Part II : Genetic search**, ORSA Journal on Computing, Vol. 8, N° 2, pp. 165-172, 1995.

26. SAVELSBERGH, M.W.P., **Local search for routing problems with time windows**, Annuals of Operations Research, 1985.

27. SOLOMON, M., **Algorithms for the vehicle routing and sheduling problems with time window constraints**, Operations Research, Vol. 35, N°2, 1987.

28. SUPRAYOGI, YAMATO, H., ISKENDAR: **Ship Routing Design for the Oily Liquid Waste Collection**, Journal of the Society of Naval Architects of Japan, Vol. 190, pp. 325-335, 2001.

29. TAILLARD, E.D., LAPORTE, G., GENDREAU, M., **Vehicle routing with multiple use of vehicles**, Journal of the Operational research society, Vol. 47, pp. 1065-1070, 1996.

30. TAILLARD, E.D., Gambardella, L.-M., Gendreau, M., Potvin, J.-Y., **Adaptive Memory Programming: A Unified View of Meta-Heuristics**, European Journal of Operational Research Vol. 135, N°. 1, pp. 1-16, 2001.

31. Tan, K.C., Lee, L.H., Zhu, Q.K., Ou, K., **Heurisic methods for vehicle routing problem with time windows**, Artificial Intelligence in Engeneering, Vol. 15, pp. 281-295, 2001.

32. THANGIAH, S.R., **Vehicle routing with time windows using genetic algorithms**, Application handbook of genetic algorithms : New frontiers, Vol. 2, pp. 253-277, CRC Press, Boca Raton 1995.

33. Zhao, Q.-H., Wang, S.-Y., Lai, K-K, Xia, G.-g., **A vehicle routing problem with multiple use of vehicles**, Advanced Modeling and Optimization, Vol. 4, N°. 3, pp. 21-40, 2002.