Restricted Dynamic Heterogeneous Fleet Vehicle Routing Problem with Time Windows[*]

Jesica de Armas, Belén Melián-Batista, and José A. Moreno-Pérez

Dpto. de Estadística, I.O. y Computación
Escuela Técnica Superior de Ingeniería Informática
Universidad de La Laguna
38271 La Laguna, Spain
{jdearmas,mbmelian,jamoreno}@ull.es

Abstract. This paper tackles a Restricted Dynamic Heterogeneous Fleet Vehicle Routing Problem with Time Windows as a real-world application of a courier service company in the Canary Islands, Spain. In this particular application of the Vehicle Routing Problem with Time Windows (VRPTW), customer requests can be either known at the beginning of the planning horizon or dynamically revealed over the day. Moreover, a heterogeneous fleet of vehicles has to be routed in real time. In addition, some other constraints required by the company, such as the allowance of extra hours for the vehicles, as well as, the use of several objective functions, are taken into account. This paper proposes a metaheuristic procedure to solve this particular problem. It has already been installed in the fleet management system of the company. The computational experiments indicate that the proposed method is both feasible to solve this real-world problem and competitive with the literature.

Keywords: Dynamic vehicle routing, Time windows, Heterogeneous fleet, Metaheuristics.

1 Introduction

Contrary to the classical static vehicle routing problems, real-world applications often include evolution, as introduced by Psaraftis in 1980 [9], which takes into consideration the fact that the problem data might change over the planning horizon. Latest developments in fleet management systems and communication technology have enabled people to quickly access and process real-time data. Therefore, dynamic vehicle routing problems have been lately given more attention. Last decade has been characterized by an increasing interest for dynamic routing problems, with solution methods ranging from mathematical programming to metaheuristics (see Pillac et al, 2013 [8] for a comprehensive review of dynamic vehicle routing problems).

[*] This work has been partially funded by the Spanish Ministry of Economy and Competitiveness (project TIN2012-32608) and the Spanish Ministry of Industry, Tourism and Trade (project TSI-020100-2011-298).

The main goal of the Restricted Dynamic Heterogeneous Fleet VRPTW (RD-HFVRPTW) tackled in this paper is to dynamically route couriers taking into account not only the requests known at the beginning of the planning horizon, but also new service requests that arrive over it. In the particular real-world application posed to the authors by a company in the Canary Islands, all delivery and about the 60% of the pick-up requests are known in advance, whereas the remaining pick-up requests arrive over the planning horizon. Moreover, companies offering courier services often have a heterogeneous fleet of vehicles, which represents an additional difficulty in the resolution of the problem. Using exact methods is not a suitable solution for this kind of problems, since the arrival of a new request has to be followed by a quick re-optimization phase to include it into the solution at hand. Therefore, most dynamic problems rely on the use of metaheuristics.

Solution approaches for Dynamic Vehicle Routing Problems (DVRP) can be divided into two main classes: those applied to dynamic and deterministic routing problems without any stochastic information, and those applied to dynamic and stochastic routing problems, in which additional stochastic information regarding the new requests is known. Given the fact that in the real-world application tackled in this paper, the information is dynamically given by the company fleet management system, we will focus on the first class of dynamic problems. In this case, solution methods can be based on either periodic or continuous re-optimization. Periodic optimization approaches firstly generate an initial solution consisting of a set of routes that contain all the static customers. Then, a re-optimization method periodically solves a static routing problem, either when new requests arrive or at fixed time slots [1]. On the other hand, continuous re-optimization approaches carry out the optimization over the day by keeping high quality solutions in an adaptive memory. In this case, vehicles do not know the next customer to be visited until they finish the service of a request.

The following literature references regarding periodic optimization approaches to solve the DVRP with Time Windows (DVRPTW) are worth mentioning. Chen and Xu [1] proposed a dynamic column generation algorithm for solving the DVRPTW based on their notion of decision epochs over the planning horizon, which indicate the moments of the day when the re-optimization process is executed. Some other papers that make also use of time slices and solve static VRPs are due to Montemani et al. [7], Rizzoli et al. [10] and Khouadjia et al. [4]. In these last papers, requests are never urgent and can be postponed since time windows are not handled. On the other hand, the work by Hong [3] does consider time windows and therefore, some request can be urgent. Hong proposes a Large Neighborhood Search algorithm for real-time vehicle routing problem with time windows, in which each time a new request arrives, it is immediately considered to be included in the current solution. In our work, the same consideration is taken into account given the urgency of some requests. The main differences of the problem tackled in this paper and the one proposed by Hong are on one hand, the fact that we consider a heterogeneous fleet of vehicles, and on the other hand, the fact that we consider a restricted version of dynamism.

Furthermore, in our real-world problem customers and vehicles can have more than one time window in the same planning horizon. In addition, customers can be postponed according to their assigned priorities. There also exist constraints that do not allow certain customers to be visited by some of the vehicles due to road restrictions. Extra hours for the vehicles may also be allowed in our problem, which incur in additional costs. Finally, several objective functions are considered and hierarchically evaluated. Hierarchic evaluation means that the functions are considered in a certain order, so that if two selected solutions have equal objective function values for a function, then the next one in the order is considered to break ties. The objective functions considered in this work are total traveled distance, time balance, that is defined as the longest minus the shortest route in time required, and cost, which includes fuel consumption and other salaries.

The main contributions of this paper are the following. This work tackles a variant of a Dynamic VRPTW, which handles all the real-world constraints required by a courier service company. The problem combines constraints, which have not been managed all together in the literature as far as we know. A metaheuristic solution approach is proposed. Furthermore, this optimization tool has been inserted into the fleet management system of the company Computational experiments over instances based on the real ones are carried out in this paper. Moreover, some preliminary experiments performed with the fleet management system are quite promising. The static part of the problem has already been successfully tested with other companies. The current state of this real-world application is the on-line communication between the fleet management system and the courier service company. Therefore, the last phase of the whole system is the combination of the solver proposed in this paper with the rest of the system, which will be performed in future works.

The rest of the paper is organized as follows. Section 2 is devoted to thoroughly describe the real-world problem posed to the authors by the company. Section 3 summarizes the metaheuristic procedure developed to solve the problem at hand. Section 4 reports the computational experiments performed in this work. Finally, the conclusions are given in Section 5.

2 Problem Description

The real-world VRPTW tackled in this paper is defined by means of a network that contains the depot and a set of N customer nodes, C, which represent the requests characterized by their type (static or dynamic), demand, location, arrival time, at_i and time window, $[e_i, l_i]$, which might not be unique. As indicated above customers can have several time windows during the day. The depot has an associated time window, $[e_0, l_0]$, and a set of K heterogeneous vehicles $V = \{v_1, .., v_K\}$ with different capacities $VC = \{vc_1, ..vc_K\}$, driving to a Heterogeneous Fleet VRPTW (HFVRPTW). Moreover, associated with each vehicle, k, there are one or more time windows $[ev_k, lv_k]$ that represent its working shift and that can be different from one vehicle to another.

As mentioned in the introduction, in the particular application of the courier service company considered in this paper, all delivery requests and a percentage of the pick-up requests are known at the beginning of the planning horizon. Therefore, these customers are considered to be static. The remaining pick-up requests, which are known over the day, are considered to be dynamic and an arrival time $at_i \in [e_0, l_0]$ is associated to the dynamic customer i. Thus, a Dynamic HFVRPTW is being taken into consideration. At this point, it is worth mentioning that the company does not allow exchanging delivery packages between vehicles due to operational purposes. If it were permitted, vehicles would have to meet in some intermediate point and devote time to carry out the exchange. Therefore, the static customers (deliveries and pick-ups) that are routed at the beginning of the planning horizon do not change their assigned route. Finally, the dynamic customers have then to be assigned to any of the existing routes while guaranteeing feasibility. The so obtained problem is referred to as Restricted Dynamic HFVRPTW (RDHFVRPTW) in this paper.

The objective function associated to the problem has not yet been established. An additional feature required by the company is the presence of multiple objective functions that have to be taken into consideration in the optimization phase. The total traveled distance, time balance, infeasibility and cost are used as it will be explained in the next section. The first objective function measures the total distance traveled by all the vehicles involved in the solution. The time balance function is stated as the time difference between the largest route and the shortest route regarding time. The infeasibility function reports the sum of time infeasibilities at each customer, i.e., the sum of the differences between the arrival time which exceeds the time window of a customer, and the upper limit of this customer's time window. Finally, the cost function indicates the fuel consumption.

In order to measure the dynamism of a given problem instance, Lund et al. [6] defined the *degree of dynamism* of the system as follows:

$$\delta = \frac{|C_D|}{N} \times 100,$$

where $|C_D|$ indicates the number of dynamic customers. Moreover, since the disclosure time of requests is also important, Larsen [5] defined the *reaction time* of customer i, that measures the difference between the arrival time, at_i, and the end of the corresponding time window, l_i. Notice that longer reaction times indicate that there is more flexibility to insert any new request into the existing routes. Therefore, the effective degree of dynamism provided by Larsen is stated as follows:

$$\delta^e_{TW} = \frac{1}{N} \sum_{i \in C} \left(1 - \frac{l_i - at_i}{T} \right),$$

where T is the length of the planning horizon. These measures will be used in the computational experience section to generate the set of problem instances used in this work.

3 Metaheuristic Solution Approach

The solution method proposed in this work to solve the RDHFVRPTW is summarized in Algorithm 1, which will be now thoroughly described. First of all, an initial solution consisting of all the static customers is generated by using the Solomon Heuristic [11]. The obtained solution is then improved running a Variable Neighborhood Search algorithm proposed in [2] to solve the static problem with all the constraints required by the company for which this dynamic problem has also to be solved. This process (lines $3-4$) is iterated for a certain number of iterations and the best reached solution is selected to be implemented by the company. In this step, all the requests known at the beginning of the planning horizon are already inserted in a route. As indicated above, these requests are not allowed to change their route assignments.

Algorithm 1. General Algorithm

```
   // Create solution $S^*$ containing all static customers
1  Initialize solution $S^*$;
2  while (a maximum number of iterations is not reached) do
3      $S \leftarrow$ Run Solomon Heuristic;
4      $S' \leftarrow$ Apply the VNS proposed by De Armas et al. [2] to $S$;
5      if ($S'$ is better than $S^*$) then
6          $S^* \leftarrow S'$;
   // Insert dynamic customers at their arrival times
7  while (a new dynamic customer, $i$, appears) do
8      Try to insert $i$ in the closest feasible existing route, $r$ if it exists;
9      if (route $r$ does not exist) then
10         if (extra hours are allowed) then
11             Insert customer $i$ in a route without any accumulated infeasibility
               taking into account the different objective functions;
12         else
13             if (the priority of customer $i$ allows it) then
14                 Postpone customer $i$ until the next day;
15             else if (there is an alternative permitted customer $j$ that let the
               insertion of $i$) then
16                 Postpone customer $j$;
17             else
18                 Insert customer $i$ in the route that supposes the smallest
                   infeasibility (if it coincides among routes, consider the remaining
                   objective functions);
19 Report infeasibility to the company;
```

Once the selected initial solution is being implemented, new dynamic customers might be revealed over the planning horizon, which have to be inserted in any existing route. Let us suppose that the dynamic customer i arrives at time at_i. As reported in line 8, the algorithm first tries to insert i in the closest possible feasible existing route. For each of these routes, from the last visited customer to the last customer in the route, it is searched the feasible insertion point either with the least distance or time balance increment. The company is interested in either minimizing the total distance and the number of routes or minimizing the time balance if all the available vehicles are used (stopping available vehicles is not desirable). If there were ties, the remaining objective functions are hierarchically evaluated. In this case, after the distance objective or time balance and cost are considered.

If the previous option is not feasible and no feasible alternative is available for customer i, then there are two options. On one hand, in case that extra hours are allowed for the vehicles (lines $10 - 11$), violating their time window constraints, customer i is tried to be inserted in a route using time window infeasibility as the objective function guiding the search. If there were ties, distance, time balance and cost, are hierarchically taken into consideration. On the other hand, if extra hours are not permitted by the company, then the notion of request priority shows up. If customer i has a low priority, then it can be postponed until the following day (lines $13 - 14$). If it is not possible, but there is a customer j that can be postponed allowing the insertion of i (lines $13 - 14$), then customer j is postponed, while i is inserted. In order to select customer i, the objective functions are hierarchically evaluated in the following order: number of postponed requests, extra hours, distance or time balance and cost.

If all the previous attempts have failed, customer i is inserted in the route with the least increment of the following objective functions (lines $17 - 18$): infeasibility, distance or time balance and cost. Finally, there are two cases in which customer i cannot be inserted in the current solution: vehicle capacities are violated or there is not any vehicle with a working shift long enough to insert the new customer. In these two cases, the corresponding infeasibility is reported to the company (line 19).

4 Discussion and Future Research

This section is devoted to analyze the performance of the algorithm proposed in this work. Despite the fact that experiments within the real system have to be carried out, the goals of the reported experiments are both to corroborate the good behavior of the method and discuss the effect of the input data over the total reached infeasibility. With these goals in mind, it has been created a set of instances based on the real data provided by a company in the Canary Islands, taking into account the features of the courier service company. A total of 20 different instances consisting of 100 customers, from which the 20% are dynamic, have been generated. In order to obtain instances with a wide range of effective dynamism degrees, random, short and large reaction times have been considered. Moreover, the standard Solomon instances are used to compare the results given by our algorithm with the best known solutions form the literature.

The first experiment reported in this section corresponds to the comparison over the standard Solomon instances. Note that the algorithm proposed in this work is thought to solve dynamic problems with the inclusion of all the real-world constraints explained in previous section. Therefore, the method is not supposed to be the most competitive over these instances, particularly due to the fact that real instances have different features. Table 1 summarizes the comparative, in which average values of number of vehicles (NV) and traveled distance (TD) are reported. The first column of the table shows the instance categories corresponding to the Solomon instances. Note that in the worst case, the deviation is about 5%.

Table 1. Computational results of the standard static Solomon instances

	NV	TD	$BestNV$	$BestTD$
$C1$	10.00	835.88	10.00	828.38
$C2$	3.12	621.01	3.00	589.90
$R1$	14.08	1252.41	11.91	1203.16
$R2$	5.00	988.26	3.00	941.87
$RC1$	13.62	1402.50	12.00	1345.56
$RC2$	6.00	1127.62	3.62	1111.99

The second experiment reported in this work is summarized in Figure 1. It shows, for each problem instance, the total infeasibility accumulated at the end of the execution when random, short and large reaction times are considered. It is noticeable that there is a clear difference between short and large reaction times. Let us remind that the reaction time of a customer provides the difference between the end of its time window and its arrival time. Longer reaction times lead to a more insertion flexibility and therefore to less infeasibility. As indicated in Figure 1, this expected behavior is obtained by our solution method.

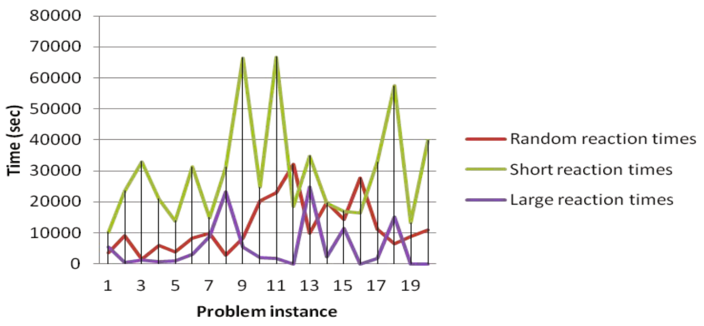

Fig. 1. Total infeasibility for different reaction times

Table 2. Dynamic customers

Dynamic Request	81	82	83	84	85	86	87	88	89	90
Arrival Time	14105	27790	19617	14111	9494	21474	31747	19119	17287	30821
Dynamic Request	91	92	93	94	95	96	97	98	99	100
Arrival Time	21811	21064	14527	15419	19106	20479	7725	25892	12850	15807

Table 3. Solution example. Insertion of dynamic requests

	$R1$	$R2$	$R3$	$R4$	$R5$	$R6$	$R1$	$R2$	$R3$	$R4$	$R5$	$R6$
Last visited customer	4	5	4	5	4	3	5	6	5	6	5	3
Accumulated infeasibility	0	0	0	0	0	0	0	0	0	0	0	0
	($C97$,$R5$,5,0)						($C85$,$R3$,8,2060)					
Last visited customer	7	8	6	9	7	3	8	9	7	9	8	3
Accumulated infeasibility	0	0	2060	0	0	0	0	0	2060	0	1751	0
	($C99$,$R5$,10,1751)						($C81$,$R4$,16,0)					
Last visited customer	8	9	7	9	8	3	8	9	7	10	8	3
Accumulated infeasibility	0	0	2060	0	1751	0	0	0	2060	0	1751	0
	($C84$,$R7$,17,0)						($C93$,$R4$,13,3103)					
Last visited customer	8	9	7	10	8	3	9	9	7	10	8	3
Accumulated infeasibility	0	0	2060	3103	1751	0	1475	0	2060	3103	1751	0
	($C94$,$R1$,11,1475)						($C100$,$R3$,11,5425)					
Last visited customer	10	11	8	11	9	3	11	12	9	12	10	3
Accumulated infeasibility	1475	0	5425	3103	1751	0	1475	0	5425	3103	1751	0
	($C89$,$R5$,12,1751)						($C95$,$R2$,19,2211)					
Last visited customer	11	12	9	12	10	3	12	13	9	12	10	3
Accumulated infeasibility	1475	2211	5425	3103	1751	0	1475	2211	5425	3103	3592	0
	($C88$,$R5$,13,3592)						($C83$,$R5$,13,1751)					
Last visited customer	13	13	10	12	11	3	14	13	10	13	11	3
Accumulated infeasibility	1475	2211	5425	3103	1751	0	1475	2211	5425	3103	3153	0
	($C96$,$R5$,14,3153)						($C92$,$R1$,20,2945)					
Last visited customer	14	13	11	13	11	3	14	13	11	13	11	3
Accumulated infeasibility	2945	2211	5425	3103	3153	0	3776	2211	5425	3103	3153	0
	($C86$,$R1$,21,3776)						($C91$,$R3$,12,5425)					
Last visited customer	17	16	11	13	14	3	19	17	12	14	14	3
Accumulated infeasibility	3776	2211	5425	3103	3153	0	3776	4814	5425	3103	3153	0
	($C98$,$R2$,20,4814)						($C82$,$R4$,15,8586)					
Last visited customer	21	18	12	15	16	3	22	19	13	15	16	3
Accumulated infeasibility	3776	4814	5425	8586	3153	0	3776	4814	5425	8586	8332	0
	($C90$,$R5$,17,8332)						($C87$,$R5$,17,8612)					

4.1 Example Solution

In this section, a solution example corresponding to a problem instance with short reaction times is provided. The initial static solution is $0-15-73-17-16-40-5-72-51-67-37-47-78-68-25-9-13-27-45-8-39-$

53 – 34 – 0 – 11 – 54 – 29 – 70 – 55 – 22 – 42 – 38 – 36 – 79 – 43 – 10 – 57 – 48 – 1 – 75 – 74 – 26 – 18 – 65 – 0 – 32 – 71 – 21 – 76 – 60 – 44 – 80 – 56 – 61 – 0 – 52 – 64 – 23 – 58 – 12 – 7 – 49 – 31 – 20 – 28 – 69 – 33 – 3 – 19 – 62 – 0 – 46 – 14 – 4 – 24 – 77 – 35 – 66 – 6 – 30 – 59 – 41 – 2 – 0 – 63 – 50 – 0, where routes $R1$ to $R6$ are separated by zeros. The dynamic customers go from $C81$ up to $C100$, and their arrival times in seconds are shown in Table 2. Table 3 reports how the dynamic customers are inserted indicating if they incur in any infeasibility. In this table, as an example, vector $(C85, R3, 8, 2060)$ indicates that customer $C85$ is inserted into route $R3$ in position 8 leading to an infeasibility value in time equal to 2060. The dynamic customers are selected according to their arrival times to be inserted into the current solution. Firstly, customer $C97$ is taken into consideration and the best position into every route is calculated. In this case, $C97$ is inserted in position 5 of route $R5$ because it produces the least traveled distance increment. Then, customer $C85$ arrives and has to be served in any of the routes. The best option corresponds to position 8 of route $R3$, reaching a total infeasibility value of 2060. The remainder customers are then selected and inserted as indicated in Table 3. For those readers interested in replicating the experiment, the instance used in this section can be downloaded from https://sites.google.com/site/gciports/vrptw/dynamic-vrptw.

5 Conclusions

This work tackles a variant of a Dynamic VRPTW, which handles all the real-world constraints required by a courier service company. The problem combines constraints, which have not been managed all together in the literature, as far as we know . A metaheuristic solution approach is proposed for solving the problem at hand. It is worth mentioning that this optimization tool has been inserted into the fleet management system owned by the company, who is the technical support of the previous one and the nexus with our research. The computational experience carried out in this work over a set of generated instances based on the real ones, has a twofold goal, corroborating the good behavior of the method and discussing the effect of the input data over the total reached infeasibility. Moreover, preliminary experiments that have been performed with the fleet management system are quite promising. The current state of this real-world application is to set up the on-line communication between the fleet management system and the courier service company, which is being correct. Therefore, the last phase of the whole system is the combination of the solver proposed in this paper with the rest of the system, which constitutes future research.

References

1. Chen, Z.L., Xu, H.: Dynamic column generation for dynamic vehicle routing with time windows. Transportation Science 40(1), 74–88 (2006)
2. De Armas, J., Melián-Batista, B., Moreno-Pérez, J.A., Brito, J.: Real-World Heterogeneous Fleet Vehicle Routing Problem with Soft Time Windows. Under review (2013)

3. Hong, L.: An improved LNS algorithm for real-time vehicle routing problem with time windows. Computers & Operations Research 39(2), 151–163 (2012)
4. Khouadjia, M.R., Sarasola, B., Alba, E., Jourdan, L., Talbi, E.G.: A comparative study between dynamic adapted PSO and VNS for the vehicle routing problem with dynamic requests. Applied Soft Computing 12(4), 1426–1439 (2012)
5. Larsen, A.: The Dynamic Vehicle Routing Problem, PhD Thesis (2001)
6. Lund, K., Madsen, O.B.G., Rygaard, J.M.: Vehicle routing problems with varying degrees of dynamism. Technical Report. IMM Institute of Mathematical Modelling (1996)
7. Montemanni, R., Gambardella, L.M., Rizzoli, A.E., Donati, A.: Ant colony system for a dynamic vehicle routing problem. Journal of Combinatorial Optimization 10(4), 327–343 (2005)
8. Pillac, V., Gendreau, M., Gueret, C., Medaglia, A.L.: A review of dynamic vehicle routing problems. European Journal of Operational Research 225(1), 1–11 (2013)
9. Psaraftis, H.N.: A Dynamic-programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. Transportation Science 14(2), 130–154 (1980)
10. Rizzoli, A.E., Montemanni, R., Lucibello, E., Gambardella, L.M.: Ant colony optimization for real-world vehicle routing problems. Swarm Intelligence 1, 135–151 (2007)
11. Solomon, M.M.: Algorithms for the Vehicle-Routing and Scheduling Problems with Time Window Constraints. Operations Research 35(2), 254–265 (1987)