GVNS for a Real-World Rich Vehicle Routing Problem with Time Windows

Jesica de Armas · Belén Melián-Batista · José A. Moreno-Pérez · Julio Brito

Received: date / Accepted: date

Abstract Rich Vehicle Routing Problems are Vehicle Routing Problems (VRPs) that deal with additional constraints, which aim to better take into account the particularities of real-world applications. They combine multiple attributes, which constitute a complement to the traditional models. This work proposes an adaptive solution method based on metaheuristics for solving a Rich VRP, referred to as Fixed Heterogeneous Fleet Vehicle Routing Problem with Time Windows. This software has been embedded into the fleet management system of a company in the Canary Islands. The attributes considered by the company are: a fixed heterogeneous fleet of vehicles, soft and multiple time windows, customers priorities and vehicle-customer constraints. Furthermore, the company requires the consideration of several objective functions that include travelled distance and time/distance balance. Exact algorithms are not applicable when solving real-life large VRP instances. This work presents a General Variable Neighbourhood Search metaheuristic, which obtains high quality solutions. The computational experiments of this work are presented in four sections, which comprise the parameter setting, the analysis of the effect of the considered attributes, the comparative with the literature for the standard VRP with Time Windows, and the study of the solutions provided by the algorithm when compared with the solutions implemented by the company. Finally, it is worth mentioning that the

Jesica de Armas
Universidad de La Laguna Dpto. de Estadística, I.O. y Computación, 38271 La Laguna, Spain
E-mail: jdearmas@ull.es

Belén Melián-Batista
Universidad de La Laguna, Dpto. de Estadística, I.O. y Computación, 38271 La Laguna, Spain
E-mail: mbmelian@ull.es

José A. Moreno-Pérez
Universidad de La Laguna, Dpto. de Estadística, I.O. y Computación, 38271 La Laguna, Spain
E-mail: jamoreno@ull.es

Julio Brito
Universidad de La Laguna, Dpto. de Estadística, I.O. y Computación, 38271 La Laguna, Spain
E-mail: jbrito@ull.es

algorithm has been integrated into a fleet management system and several tests with real companies have been conducted.

Keywords Rich VRPTW · Fixed Heterogeneous Fleet · General Variable Neighbourhood Search · Metaheuristics

1 Introduction

Many practical applications related to logistics in intelligent freight transportation systems lead to vehicle routing problems with varying degrees of difficulty regarding the problem constraints. The basic Vehicle Routing Problem (VRP) is composed of a set of customers requiring a specified volume of goods to be delivered. A fleet of homogeneous vehicles dispatched from a single depot is used to deliver the goods, returning to the same depot once the routes have been completed. The constraints associated to the problem are that vehicles can carry a maximum capacity and each customer has to be visited once by a single vehicle. The VRP has been the subject of intensive research since the 1960s. A wide range of exact methods, heuristics and metaheuristics has been proposed in the specialized literature. We refer the interested reader to the following surveys by Schmid et al. (2013); Vidal et al. (2013); Eksioglu et al. (2009); Potvin (2009); Laporte (2009); Gendreau et al. (2008); Baldacci et al. (2007), and books by Toth and Vigo (2002) and Golden et al. (2008).

The aim of this work is to solve a real-world VRP that has been posed to the authors by a company in the Canary Islands, Spain. The resulting software has been embedded into a fleet management system. The requirements provided by the company lead to the consideration of several constraints, which have to be integrated into the standard VRP. In the literature, there is a tremendous number of research papers related to VRP with additional constraints, which range from the need of time windows to regulations related to long-distance transportation. With the purpose of collecting all these possible constraints, Vidal et al. (2013) have given the notion of *attributes* of VRPs. Attributes refer to additional constraints that aim to better take into account the specificities of real-world applications. These attributes complement the traditional VRP formulations and lead to a variety of *Multi-Attribute Vehicle Routing Problems* (*MAVRPs*). These MAVRPs are supported by a well developed literature that includes a wide range of heuristics and metaheuristics (Glover, 1986). Their analysis is limited to single objective optimization problems. Furthermore, some MAVRPs combine multiple attributes together, obtaining the so-called *Rich VRPs* (*RVRP*) (Schmid et al., 2013; Tarantilis et al., 2009). The problem tackled in this work corresponds to this last class of RVRPs.

Vidal et al. (2013) distinguish three main classes of attributes that appear frequently in the literature. These classes are the assignment of customers and routes to resources, the sequence choices, and the evaluation of fixed sequences. We refer the interested reader to that paper for more details about this classification. The attributes that are taken into consideration in this work are summarized in the following items.

– *Heterogeneous fleet*. In the particular application of VRP tackled in this paper, it is considered a heterogeneous fleet of vehicles. When the number of available vehicles is not limited, the problem is usually referred to as Vehicle Fleet Mix Problem

(VFMP). In the case in which the fleet of vehicles is limited, a different version of the problem, called Heterogeneous Fleet VRP (*HFVRP*), is revealed. This last problem corresponds to the real-world application solved in this work. Precisely, it is available a fixed set of heterogeneous vehicles. Therefore, we refer to the so obtained problem as Fixed HFVRP with Time Windows (*FHFVRPTW*). Exact and heuristic methods have been proposed for solving HFVRPs (Baldacci et al., 2008; Baldacci and Mingozzi, 2009; Taillard, 1999; Li et al., 2007; Paraskevopoulos et al., 2008; Prins, 2009; Brandão, 2011; Penna et al., 2011; Subramanian et al., 2012). Most literature papers assume an unlimited number of available vehicles, so that the objective is generally to obtain a solution that either minimizes the number of vehicles and/or total travel cost. However, the real-world problems arising in companies face several resource constraints such as a fixed fleet of vehicles. Therefore, it might not be possible to obtain a feasible solution for a certain instance. In this case, it is required to obtain a valid solution for the company by adding more vehicles, letting the drivers work after their working shift, postponing customers and maximizing the number of customers served, etc.

- *Time windows*. Additional constraints arise if time windows are associated to the depot and customers, obtaining the Vehicle Routing Problem with Time Windows (*VRPTW*). VRPTW and a vast set of its variants have been widely studied in the literature. For recent reviews, see Bräysy and Gendreau (2005) and Gendreau and Tarantilis (2010). Furthermore, in the practical application reported in this paper, working shifts of vehicles are considered as time windows associated with each vehicle.
- *Soft & Multiple time windows*. In the implementation carried out in this paper, additional time attributes, which are the existence of multiple time windows for customers and multiple time intervals in the working shifts for vehicles, are taken into consideration (Ibaraki et al., 2005, 2008). Note that time windows may differ among customers, and working shifts may differ among vehicles. In any case, the customers have to be visited at maximum once during the day. Moreover, soft time windows and soft working shifts are considered, since some of them can be violated, incurring in additional costs. Particularly, if working shifts of vehicles can be extended, extra hours are allowed for the drivers. This leads to additional salary costs, since the extra time is more expensive. A work related to VRP with soft time windows is Taillard et al. (1997).
- *Customer priority*. In addition to the previous attributes, which are thoroughly analyzed in the paper by Vidal et al. (2013), the company under consideration in this work assigns priorities to some customers. Depending on these priorities, some customers can be postponed until the next day and their service is not required during the current planning horizon. Together with extending the working shifts of the vehicles, postponing customers allow the system to obtain valid solutions for the company. Therefore, in the case in which the fixed fleet of vehicles is not sufficient for serving all customers, allowing extra time and/or postponing customers are possible alternatives if they are permitted by the company.
- *Vehicle-Customer restrictions*. There are also vehicle-customer limitations, which indicate that some customers cannot be served by some vehicles. Therefore, there

will be a set of vehicle-customer constraints that can be due to several reasons such as road restrictions.

In addition to these attributes, *several objective functions* are required by the company to solve the problem at hand. Although the optimality criterion of minimizing the total travelled distance is the most commonly used in the VRP literature, more recent approaches recognize the vehicle routing problem as a multi-objective optimization problem. Jozefowiez et al. (2008) provide an overview of the research into routing problems with several objectives. Important objectives, besides the minimization of the total travelled distance, are the minimization of the number of vehicles in use, the minimization of the total required time, the maximization of the collected profit and some other objectives related to reaching a balance between the routes. In order to establish a balancing objective, the workload for a route has to be defined. It can be expressed, for example, by the number of visited customers, the quantity of delivered goods, the route length or the required time (Borgulya, 2008; Jozefowiez et al., 2008). Among the researchers who have worked on vehicle routing problems considering several objectives and time windows we can mention, for instance, Hong and Park (1999), who consider the minimization of total vehicle travel time and the minimization of total customer waiting time. Rahoual et al. Rahoual et al. (2001) consider objectives related to the minimization of the number of used vehicles and the minimization of the total covered distance. More recently, Calvete et al. Calvete et al. (2007) minimize the total operational cost, the under-utilization of labour and the vehicle capacity. Jozefowiez et al. Jozefowiez et al. (2007) discuss the motivations for applying multi-objective optimization on vehicle routing problems and the potential benefits of doing it. Ghoseiri et al. Ghoseiri and Ghannadpour (2010) present a model and a solution method based on genetic algorithms to solve the multi-objective problem considering as objectives both the total required fleet size and total travelling distance. Melián-Batista et al. Melin-Batista et al. (2014) consider for the first time in the literature the objective of balancing routes regarding time in conjunction with time windows in a multi-objective context.

In this work, given the fact that the fleet of vehicles is fixed, the company might require either minimizing the total distance or balancing time or distance if the use of all the available vehicles is mandatory. Therefore, the company has to indicate which principal objective function will be required. If having idle available vehicles is not allowed, then the time/distance balance objective function will be selected as principal one. Time balance is measured as the difference between the longest and shortest routes regarding time. Distance balance is also measured as the difference between the longest and shortest routes regarding distance. Otherwise, minimizing the total distance will be the principal objective function. Furthermore, a set of other objective functions are considered together with the principal one, as explained below; particularly, minimizing the number of vehicles, extra hours, postponed customers and cost. All these functions will be used following a different lexicographic ordering of them, depending on the particular goal in each case.

Finally, *infeasible solutions* are taken into consideration. Particularly, infeasibilities due to the use of more vehicles than available, the extension of the time windows

of the customers and the working shifts of the vehicles, or the postponing of customers are tackled.

Due to the difficulty for solving VRPs to optimality, heuristics and metaheuristics constitute an increasingly active research area in the literature. In our work, a General Variable Neighbourhood Search algorithm (VNS) (Hansen et al., 2010b) is proposed. The main differences between the problem tackled in this paper and the ones proposed in the literature are that we consider a fixed heterogeneous fleet of vehicles, and several real constraints/attributes. Furthermore, as far as we know, this is the first work in the literature that uses all the previously explained attributes together.

A tremendous amount of work in the field of vehicle routing problem using VNS has been published. Bräysy (2003) gives the internal design of the Variable Neighbourhood Descent (VND) and Reduced Variable Neighbourhood Search (RVNS) algorithms in detail, analyzes the VRPTW problem, and indicates the VND algorithm as one of the most effective ways to solve VRPTW problems. Polacek et al. (2004) design a VNS to solve the multidepot vehicle routing problem with time windows MDVRPTW. Kytöjoki et al. (2007) design a guided VNS algorithm to handle the 32 existing large-scale VRP problems and compare it with a tabu search algorithm (TS). The results showed that the VNS algorithm was more efficient than the TS algorithm in computational time. Goel and Gruhn (2008) introduce a RVNS to solve the general VRP problem including time windows, vehicle constraints, path constraints, travel departure time constraints, capacity constraints, order models compatibility constraints, multisupplier point of the orders, and transport and service position constraints. Hemmelmayr et al. (2009) propose a VNS algorithm for periodical VRP problem. Fleszar et al. (2008) adopt a VNS algorithm to solve the open-loop VRP problem and test 16 benchmark problems. In summary, several literature papers have proved the effectiveness of developing VNS algorithms to solve a wide variety of VRPs.

It is worth mentioning that the solution method proposed in this work, implemented as a metaheuristic, has already been integrated into the optimization tool of the fleet management system used by some companies. The fleet of a company which wants to use this optimization tool must have the necessary devices to communicate with the management system, and then the system will be able to use the optimization tool and provide an optimized route plan. The optimization tool has been implemented using $C\#$ and the current solution method has been implemented using C++. Therefore, in order to integrate the metaheuristic method in the optimization tool, a DLL has been developed. The data-interchange format used to communicate the optimization tool with the DLL has been JSON, that is a text-based open standard designed for human-readable data interchange. Furthermore, it is worth noting that through the system interface, the company can activate or deactivate the consideration of the different attributes.

The main contributions of this paper relies upon the fact that the VRPTW including several real-world constraints required by some companies has been tackled. Particularly, a fixed heterogeneous fleet of vehicles is considered. Moreover, since the fleet is fixed, there might be customers which cannot be served during the planning horizon and the so obtained infeasibility has to be managed. Two alternative solutions are given in this work; allowing the drivers work after their working shift or

maximizing the number of customers served postponing the remainder. Furthermore, multiple objective functions are taken into consideration. The problem combines constraints which have not been managed all together in the literature as far as we know. A metaheuristic solution approach based on VNS is proposed. Moreover, this optimization tool has been inserted into the fleet management system used by some the companies. Computational experiments on instances based on the real ones and standard benchmark instances, have been carried out in this paper. Since the main goal of our work has been to embed the developed software into a commercial fleet management tool, it is desirable to develop an algorithm that not only performs well over the standard VRPTW instances, but also over constrained real-world problems. Therefore, this paper is not aimed at overcoming the best standard VRPTW results, but rather at proposing an algorithm that works effectively when a fixed heterogeneous fleet is used to solve real-world instances. Finally, it is worth mentioning that the experiments performed with the fleet management system are quite promising.

The rest of the paper is organized as follows. Section 2 is devoted to describe the real-world Fixed Heterogeneous Fleet Vehicle Routing Problem with Time Windows (FHFVRPTW) tackled in this work. Section 3 thoroughly describes the General Variable Neighbourhood Search (GVNS) algorithm developed to solve the problem at hand. Section 4 summarizes the computational results carried out over both instances from the literature and real-world data. Finally, the conclusions and future work are reported in Section 5.

2 Real-world Fixed Heterogeneous Fleet VRPTW

The real-world Fixed Heterogeneous Fleet VRPTW (FHFVRPTW) tackled in this paper is defined by means of a network $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes, and $\mathcal{A}$ is the set of arcs. It contains the depot, D, and a set of n customer nodes, $\mathcal{C}$, which represent the requests characterized by their demand, location and time windows, which might not be unique. Customer i can have several time windows during the day, $h \in H_i$, although it is served at maximum once during the day. The depot has an associated time window, that is unique, and a set of m heterogeneous vehicles with different capacities $Q = \{q\_v_1, .., q\_v_m\}$. Moreover, associated with each vehicle, k, there is a working shift composed of several time intervals, $h' \in H_k$, during the planning horizon that can be different from one vehicle to another. With the purpose of formally describing the FHFVRPTW, the following mixed linear integer program is presented.

Model Parameters

- $\mathcal{C} = \{1, .., n\}$: set of n customers.
- $\mathcal{V} = \{0, n+1\} \cup \mathcal{C}$; set of nodes, where 0 and $n+1$ are two copies of the depot.
- s_i: service time of customer $i \in \mathcal{C}$.
- q_i: demand of customer $i \in \mathcal{C}$.
- t_{ij}: travelling time between customers $i \in \mathcal{V}$ and $j \in \mathcal{V}$, with arc $(i, j) \in \mathcal{A}$.
- d_{ij}: travelling distance between customers $i \in \mathcal{V}$ and $j \in \mathcal{V}$.
- $\mathcal{K} = \{1, .., m\}$: set of m vehicles.
- $Q = \{q\_v_1, .., q\_v_m\}$: capacities of the vehicles.

- $[a_{ih}, b_{ih}]$, with $h \in H_i = \{1, .., n_i\}$: multiple time windows of customer $i \in \mathcal{C}$.
- $[ad_0, bd_0]$ and $[ad_{n+1}, bd_{n+1}]$ represent the time window corresponding to the depot.
- $[a_{kh'}, b_{kh'}]$, with $h' \in H_k = \{1, .., n_k\}$: multiple time intervals for the working shift of vehicle k.
- M: a large amount.

Decision Variables

- $x_{ijkh'}$ equal to 1 if and only if vehicle k traverses arc $(i, j) \in \mathcal{A}$ in its $\{h'\}^{th}$ working shift.
- $y_{ikhh'}$ equal to 1, if and only if customer i is visited by vehicle k in the h^{th} time window of the customer and the $\{h'\}^{th}$ working shift of the vehicle.
- u_i: extra time with respect to the start of service at customer i.
- $v_{ikh'}$: extra time with respect to the start of service at depot ($i \in \{n+1\}$) of vehicle k in its working shift h'.
- $e_{ikh'}$: time at which vehicle k starts the service at customer i at working shift h'.

Objective Function

If the total travelled distance is considered as principal objective, the objective function (1) is established.

$$\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} d_{ij} \sum_{k \in \mathcal{K}} \sum_{h' \in H_k} x_{ijkh'} +$$

$$+M \sum_{i \in \mathcal{V}} u_i + M \sum_{k \in \mathcal{K}} \sum_{h' \in H_k} v_{n+1kh'} +$$

$$+ \sum_{i \in \mathcal{C}} M(1 - \sum_{k \in K} \sum_{h \in H_i} \sum_{h' \in H_k} y_{ikhh'}) \quad (1)$$

Constraints

Constraints (2) guarantee that the capacity of the vehicles is not exceeded in each of the time intervals of their working shifts. Note that the vehicles can perform several routes starting from the depot in their working shifts; one in each time interval.

$$\sum_{i \in \mathcal{V}} \sum_{h \in H_i} q_i y_{ikhh'} \leq q\_v_k, \forall k \in \mathcal{K}, h' \in H_k \quad (2)$$

Constraints (3) and (4) ensure that every customer is visited once at maximum and that the depot is used by every vehicle k in each of the time intervals of its working shift. Note that customers can be postponed, which is penalized in the third term of the objective function.

$$\sum_{k \in \mathcal{K}} \sum_{h \in H_i} \sum_{h' \in H_k} y_{ikhh'} \leq 1, \forall i \in \mathcal{C} \quad (3)$$

$$\sum_{k \in \mathcal{K}} \sum_{h \in H_i} \sum_{h' \in H_k} y_{ikhh'} \leq \sum_{k \in \mathcal{K}} n_k, \forall i \in \{0, n+1\} \quad (4)$$

Flow conservation is guaranteed by Constraints (5) and (6).

$$\sum_{i \in \mathcal{V}} x_{ijkh'} = \sum_{h \in H_j} y_{jkhh'}, \forall j \in \mathcal{V} \setminus \{0\}, k \in \mathcal{K}, h' \in H_k \tag{5}$$

$$\sum_{j \in \mathcal{V}} x_{ijkh'} = \sum_{h \in H_i} y_{ikhh'}, \forall i \in \mathcal{V} \setminus \{n+1\}, k \in \mathcal{K}, h' \in H_k \tag{6}$$

The set of constraints (7) ensures feasibility with respect to time.

$$e_{ikh'} + s_i + t_{ij} \leq e_{jkh'} + M(1 - x_{ijkh'}),$$

$$\forall i, j \in \mathcal{V}, k \in \mathcal{K}, h' \in H_k \tag{7}$$

Constraints (8), (9) and (10) enforce the time windows of customers and working shifts of vehicles, whose violations are penalized in the objective function.

$$\sum_{h \in H_i} \sum_{h' \in H_k} a_{ih} y_{ikhh'} \leq \sum_{h' \in H_k} e_{ikh'} \leq$$

$$\sum_{h \in H_i} \sum_{h' \in H_k} b_{ih} y_{ikhh'} + u_i, \forall i \in \mathcal{C}, k \in \mathcal{K} \tag{8}$$

$$ad_i \leq e_{ikh'} \leq bd_i + u_i, \forall k \in \mathcal{K}, h' \in H_k, i \in \{0, n+1\} \tag{9}$$

$$a_{kh'} \sum_{h \in H_i} y_{ikhh'} \leq e_{ikh'} \leq b_{kh'} \sum_{h \in H_i} y_{ikhh'} + v_{ikh'},$$

$$\forall k \in \mathcal{K}, h' \in H_k, i \in \{0, n+1\} \tag{10}$$

Finally, the following constraints, labelled as (11) to (15), impose conditions on the variables.

$$x_{ijkh'} \in \{0, 1\}, \forall i, j \in \mathcal{V}, k \in \mathcal{K}, h' \in H_k \tag{11}$$

$$y_{ikhh'} \in \{0, 1\}, \forall i \in \mathcal{V}, k \in \mathcal{K}, h \in H_i, h' \in H_k \tag{12}$$

$$e_{ikh'} \geq 0, \forall i \in \mathcal{V}, k \in \mathcal{K}, h' \in H_k \tag{13}$$

$$u_i \geq 0, \forall i \in \mathcal{V} \tag{14}$$

$$v_{ikh'} \geq 0, \forall i \in \{0, n+1\}, k \in \mathcal{K}, h' \in H_k \tag{15}$$

3 General Variable Neighborhood Search for the FHFVRPTW

Exact algorithmic methodologies are not applicable when solving real-life large vehicle routing problem instances. Therefore, our interest is focused on metaheuristic methodologies that are capable of producing applicable high quality solutions within reasonable computing times. With the purpose of obtaining high quality solutions for the real-world problem at hand, this work proposes an algorithm based on General Variable Neighbourhood Search (GVNS) (Hansen et al., 2010b). Variable Neighbourhood Search (VNS) is a metaheuristic for solving combinatorial and global optimization problems based on a simple principle; systematic changes of neighbourhoods within the search. Many extensions have been made, mainly to be able to solve large problem instances (Melian, 2006; Hoeller et al., 2008; Moreno-Vega and Melian, 2008; Hansen et al., 2008, 2010a).

Let $\mathcal{N}_k$ ($k = 1, \ldots, k_{max}$) be a finite set of neighbourhood structures, and $\mathcal{N}_k(s)$ the set of solutions in the k^{th} neighbourhood of a solution s. Usually, a series of nested neighbourhoods is obtained from a single neighbourhood by taking $\mathcal{N}_1(s) = \mathcal{N}(s)$ and $\mathcal{N}_{k+1}(s) = \mathcal{N}(\mathcal{N}_k(s))$, for every solution s. This means that a move to the k-th neighbourhood is performed by repeating k times a move into the original neighbourhood. A solution $s' \in S$ is a *local minimum* with respect to $\mathcal{N}_k$ if there is no solution $s \in \mathcal{N}_k(s') \subseteq S$ better than s' (i.e., such that $f(s) < f(s')$ where f is the objective function of the problem). In the implementation performed in this work, the neighbourhoods selected for the shaking process of the VNS are not nested, and different kinds of movements are implemented following the ideas described by Repoussis et al. (2006). The proposed sequence of movements ($k_{max} = 6$) is defined as follows: $GENI$, $Or - opt$, $CROSS$, $2 - opt$, $relocate$ and $swapInter$. This sequential selection is applied based on cardinality, which implies moving from relatively poor to richer neighbourhood structures. The $GENI$ operator (Gendreau et al., 1992) chooses a customer from a route and inserts it into other route between the two closest customers to the previous one. The $Or - opt$ operator (Or, 1976) relocates a chain of consecutive customers of a route. The $CROSS$ operator (Taillard et al., 1997) selects a subsequence of customers from a route, other subsequence of customers from other route, and interchanges both subsequences. The $2 - opt$ operator (Croes, 1958) chooses two customers of a route and inverts the sequence of customer visited between them. The $relocate$ operator (Cassani and Righini, 2004) deletes a customer from a route and inserts it into another route. The $swapInter$ operator selects a customer from a route, other customer from other route, and swaps them.

Additionally, let $\mathcal{N}_l$, ($l = 1, \ldots, l_{max}$) be the finite set of neighbourhood structures that will be used in the local search conducted by a Variable Neighbourhood Descent (VND). The Variable Neighbourhood Descent (VND) method is obtained if the change of neighbourhoods is performed in a deterministic way. Its steps are presented in Algorithm 1. The sequence of movements considered in this work ($l_{max} = 3$) is the following: $relocate$, $swapIntra$ and $swapInter$.

In order to solve the FHFVRPTW, we propose the GVNS metaheuristic, whose pseudocode is shown in Algorithm 2. Once defined the neighbourhood structures $\mathcal{N}_k$ and $\mathcal{N}_l$ in line 1, the best solution is initialized at the empty set and the stopping

Algorithm 1: Variable Neighborhood Descent (VND)

```
   // Function VND(s, l_max).
1  while (improvement is obtained) do
2      Set l ← 1;
3      while (l <= l_max) do
4          s' ← argmin_{y∈N_l(s)} f(s);
5          NeighbourhoodChange(s,s',l); //Change neighbourhood
```

condition is chosen in lines 2 and 3, respectively. Then, for each iteration, an initial solution is generated in line 5. With this purpose, an ordering of the available vehicles is obtained according to which the vehicles are selected to create the routes. This ordering is given taking into account the capacity of each vehicle, in such a way that vehicles with larger capacity are selected before. If there are multiple vehicles with the same capacity, then they will be sorted according to the number of consecutive hours that the vehicle is available, so that vehicles having larger working shifts are selected before. Once having the order of selection of vehicles, the routes are created one after the other. To create a route, a vehicle and a seed customer, which will be selected among the two customers that are the farthest from the depot, have to be chosen. Each customer is then attempted to be inserted, but if it is not compatible with the vehicle due to restrictions, the next vehicle in the sorted list is chosen. After inserting the seed customer, the proposed procedure follows the Solomon algorithm (Solomon, 1987), establishing the route locations where to insert each unplanned customer and selecting the best customer to be inserted. When no more customers can be inserted into the current route, a new one is created.

The process of creating a new initial solution takes into account several aspects. In the first place, it is worth mentioning the fact that some customers cannot be assigned to certain vehicles due to restrictions. Moreover, if the implemented Solomon heuristic requires more vehicles than available, fictitious vehicles are generated. These vehicles are used to create the other necessary routes and their working shifts are set at the least restrictive values of all initial vehicles. Fictitious vehicles are also included when the working shifts of the remaining vehicles are too restrictive to serve the customers or when the customers are not compatible with the remaining vehicles. Before introducing fictitious vehicles in the case in which compatible vehicles are still available, it is checked if it is allowed to expand their working shifts obtaining extra working hours. If this is permitted, some customers can then be assigned to the current expanded route.

Before continuing the explanation of the procedure proposed in this work, it is worth mentioning that even though several objective functions are considered, the company has to indicate which is the principal objective in each case. Therefore, the total distance, time balance or distance balance can be considered as principal objective. In the last two cases, if the solution obtained with the procedure in line 5 uses less vehicles than available, then an empty route for each unused vehicle will be included in the solution. The rationale behind this is that the company does not want to have any idle available vehicle.

Algorithm 2: General Variable Neighborhood Search (GVNS)

```
   // Initialization.
1  Select the set of structures N_k, for k = 1, ..., k_max, that will be used in the shaking phase, and
   the set of neighbourhood structures N_l for l = 1, ..., l_max that will be used in the local search.
2  Initialize BestSol ← ∅.
3  Choose a stopping condition.
4  while (the stopping condition is not met (N is not reached)) do
5      Generate an initial solution s.
       // Iterations.
6      while (the stopping condition is not met (M is not reached)) do
7          (1) Set k ← 1;
8          (2) Repeat the following steps until k = k_max:
9              (a) Shaking. Generate a point s' at random from the k^th neighbourhood of s
           (s' ∈ N_k(s)).
10             (b)Local search by VND.
11                 (b1) Set l ← 1;
12                 (b2) Repeat the following steps until l = l_max:
13                     − Exploration of neighbourhood. Find the best neighbour s of s in N_l(s);
14                     − Move or not. If f(s'') < f(s'), set s' ← s'' and l ← 1; otherwise, set
           l ← l + 1;
15             (c) Move or not. If this local optimum is better than the incumbent, move there
           (s ← s''), and continue the search with N_1 (k ← 1); otherwise, set k ← k + 1.
16     Update BestSol.
```

The best solution is initialized to the empty set in line 2. The stopping condition in line 3 consists of a number of iterations that corresponds to a parameter N that is set in the computational experience.

The loop corresponding to lines 6-15 is performed for a number of iterations, M, set by the computational experience. As indicated above, the sequence of neighbourhoods used to carry out the shaking process in GVNS is the following: GENI, Or-Opt, Cross, 2-Opt, Relocate and swapInter. Therefore, line 7 indicates that the first considered neighbourhood is GENI.

The particular implementation performed in this work of these neighbourhoods is summarized in the following items:

– *GENI*

In order to make this movement, the next steps are repeated a certain number of times. Firstly, a source route must be selected among the fictitious ones, but if there is not any fictitious route, a non empty source route is randomly selected. Then, a destination route must be selected among the empty ones, but if there is not any empty route, the destination route is randomly selected among the three ones which have the closest centroid to the source route. This destination route cannot be fictitious and must have more than one customer. Then, a customer that can be deleted from the source route is selected among the three customers closest to the destination route (sum of distances from this customer to all destination route customers), and the two customers from the destination route closest to the previous one are chosen. If some of these customers cannot be found, the process is tried again. Otherwise, the customer from the source route is inserted between

the two customers in the destination route. If the resulting route is infeasible, the movement is undone and the process is tried again. If not, we eliminate the source route from the planning if it is left empty and the main objective in optimization is to minimize the distance or the route is fictitious.

- *Or-opt*
  First of all, in order to perform this movement, it is necessary to check if all routes have only two customers. In this case, this movement cannot be carried out. Otherwise, the next steps are repeated a certain number of times. Initially, a route with more than two customers is randomly selected. Then, two different customers are randomly chosen from this route and a position inside the route is selected to move this sequence of customers. At this point, the movement is done. If the resulting route is infeasible, the movement is undone and the process is tried again.
- *CROSS*
  In order to perform this movement, the next steps are repeated a certain number of times. Firstly, a source route with more than one customer is randomly selected. Secondly, a destination route with more than one customer is randomly selected among the three ones which have the closest centroid to the source route. Then, two customers from the source route and two customers from the destination route are randomly selected and the interchange is done. If any of the routes is not feasible, the movement is undone and the process is tried again.
- *2-opt*
  First of all, in order to perform this movement, it is necessary to check if all routes have only one customer. In this case, this movement cannot be carried out. Otherwise, the next steps are repeated a certain number of times. Initially, a route with more than one customer is randomly selected. Then, two different customers are randomly chosen from this route and the sequence is reversed. If the resulting route is infeasible, the movement is undone and the process is tried again.
- *Relocate*
  In order to perform this movement, the next steps are repeated a certain number of times. Firstly, a source route must be selected among the fictitious ones, but if there is not any fictitious route, a non-empty source route is randomly selected. Then, a destination route must be selected among the empty ones, but if there is not any empty route, the destination route is randomly selected among the three ones which have the closest centroid to the source route. This destination route cannot be fictitious. Later, a customer which can be deleted from the source route is selected among the three customers closest to the destination route (sum of distances from this customer to all destination route customers), and a customer from the destination route after which the previous one can be feasibly introduced is chosen. If some of these customers cannot be found, the process is tried again. Otherwise, the relocation is done, and we eliminate the source route from the planning if it is left empty and the main objective in optimization is to minimize the distance or the route is fictitious.
- *SwapInter*
  In order to perform this movement, the next steps are repeated a certain number of times. Initially, a non-empty first route is selected. Then, a second route must

be selected among the three ones that have the closest centroid to the first route. This second route cannot be empty. Then, a customer that can be deleted from the first route is selected among the three closest customers to the destination route (sum of distances from this customer to all second route customers), and a customer that can be deleted from the second route is selected among the three closest customers to the first route (sum of distances from this customer to all first route customers). If some of these customers cannot be found, the process is tried again. Otherwise, the swap is done and its feasibility is checked. If any of the routes is infeasible, the movement is undone and the process is tried again.

– *SwapIntra*
  First of all, in order to perform this movement, it is necessary to check if all routes have only one customer. In this case, this movement cannot be carried out. Otherwise, the next steps are repeated a certain number of times. Initially, a route with more than one customer is randomly selected. Then, two different customers are randomly chosen from this route and the swap between both customers is done. If the resulting route is infeasible, the movement is undone and the process is tried again.

The processes of shaking, local search and move decision in lines 9, 10 and 15, respectively, are iteratively performed until $k = k_{max}$. In the first place, the shaking step in GVNS generates a solution s' at random from the k^{th} neighbourhood of s ($s' \in \mathcal{N}_k(s)$). Then, a local search based on VND is performed from s' to obtain a solution s''. The VND procedure uses the $\mathcal{N}_l$ neighbourhoods, which in the implementation proposed for solving the FHFVRPTW consists of the next sequence of random movements: Relocate, swapIntra and swapInter.

As indicated above, the user of the system has to indicate which the principal objective function will be among minimizing the total distance, time balance or distance balance. In the last two cases, it is considered the difference between the largest and shortest time/distance made by the used vehicles. Additionally, the developed GVNS takes into consideration additional objective functions that have to be minimized using a hierarchical approach. Hierarchic evaluation means that the objective functions are considered in a certain lexicographic 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. In the case in which the total distance is the principal objective, the lexicographic order is the following.

– Number of fictitious routes.
– Total travelled distance.
– Total number of routes.
– Salary costs incurred for expanding the working shifts of the vehicles (extra hours are more expensive).

The case in which the principal objective function is either time or distance balance, the objective functions order is the following.

– Number of fictitious routes.
– Time/Distance balance.
– Total travelled distance.

– Salary costs incurred for expanding the working shifts of the vehicles (extra hours are more expensive).

Note that in this case, the number of vehicles is fixed as indicated by the company. Therefore, minimizing the number of routes is not an objective to be considered.

The rationale behind considering the number of fictitious routes as the first objective function to be minimized is the fact that they can lead to undesirable infeasible solutions. If the customer services assigned to fictitious routes cannot be relocated in other routes and the company does not allow postponing them to the next day, then an infeasible solution is obtained. This result can also be valid for the company since, in this case, it can rent additional vehicles for a single day. In any case, a feasible solution is always preferred.

The lexicographic order is used to carry out the exploration of the neighbourhoods in the local search (line 11). If there are ties between different solutions with respect to an objective function value, then the next one in the order is used to break them. This approach, considered within VNS, has been referred as Variable Formulation Search in the paper by Pardo et al. (2013).

After one of the N iterations, a solution to the problem, either feasible or infeasible, is obtained. Since several iterations are carried out to finally select the best alternative solution for the company, in line 16, the best solution is updated. Two different goals that substitute the minimization of the number of fictitious routes are taken into account. The obtained lexicographic order corresponding to the travelled distance as principal objective is the following.

– Number of postponed services.
– Number of extra hours.
– Total travelled distance.
– Number of routes.
– Salary costs incurred for expanding the working shifts of the vehicles (extra hours are more expensive).

The obtained lexicographic order corresponding to the time/distance balance as principal objective is the following.

– Number of postponed services.
– Number of extra hours.
– Time/Distance balance.
– Total travelled distance.
– Salary costs incurred for expanding the working shifts of the vehicles (extra hours are more expensive).

Therefore, in order to update the best solution in line 16, it is given higher priority to those solutions that include all the services and adjust to the working shifts of the drivers. These two aspects are not considered while running the GVNS, since the extra hours remain unchanged, on one hand, and determining the services belonging to fictitious routes that can be postponed requires high computational costs, on the other hand.

To summarize the description of the proposed algorithm, it is noteworthy that in order to carry out the optimization process, the system requires that the user specifies the desirable principal objective function, if extra hours are permitted and what service priorities allow postponing services.

4 Computational Experiments

This section is devoted to thoroughly describe the computational experiments carried out in this work to assess the quality of the solutions provided by the algorithm developed to solve the real-world FHFVRPTW. Our algorithm has been coded in C++ and runs on a machine with Intel(R) Core(TM) i5-2320 CPU, 3 GHz, 6 GB of RAM. The platform used has been Ubuntu 12.04.

First of all, the parameters of the algorithm are adjusted. Secondly, five different experiments are executed to analyze the effect of adding to the standard VRPTW, the real-world constraints suggested by the company. Furthermore, a comparative analysis with the best results from the literature corresponding to the standard VRPTW Solomon benchmark instances [1] is performed. Finally, results corresponding to real instances are also analyzed.

4.1 Parameter setting

This section reports the experiments carried out to set the parameters that appear in the GVNS implemented in this work. Table 1 summarizes the parameters involved into the different phases of the GVNS that have been adjusted and the values that have been tested.

As indicated in Section 3, the Solomon heuristic has been modified to take into consideration the additional constraints required by the company. In the particular implementation that is proposed in this work, Solomon heuristic provides a different solution for each execution because we have used a parameter in order to select the seed for each route. It allows selecting a seed among the α farthest remaining customers from the depot. After doing the corresponding executions, this parameter has been set to $\alpha = 2$ at the view of the results and the statistical test.

The movement operators used by the GVNS algorithm also need some parameters, which have been set. These parameters are used by movement operators that involve two routes. Regarding the first parameter, when it is necessary to select the second route to make the movement, this is selected among the $\beta = 3$ closest routes to the previous one. The second parameter is used to select a customer to be deleted from a route. This customer is selected among the $\gamma = 3$ closest feasible ones to the route where it will be inserted. The third parameter is used to select a customer from a route after which a previous chosen one could be inserted. This customer is selected among the $\lambda = 3$ closest feasible ones to the previous chosen customer.

Moreover, the number of iterations, M, that the shaking, local search and move decision processes are carried out in each iteration of the GVNS has been fixed to

[1] $http://web.cba.neu.edu/\ msolomon/problems.htm$

$M = 20$. It means that, at least, 20 points are selected from each neighbourhood in the shaking phase.

Finally, as mentioned above, the general algorithm consists of repeating the Solomon and GVNS algorithms for N iterations. The number of repetitions has been fixed to $N = 10$ due to the good relation between the solution quality and the computational time necessary to obtain it.

The parameter setting for the algorithms has been done using the Friedman test (Daniel, 1990), which provides the best configuration of parameters and detects differences or equalities among configurations. The test instances used to do this parameter setting have been chosen among the Solomon instances with 100 customers. Two instances of each category have been randomly selected: $C105$, $C107$, $R104$, $R109$, $RC102$, $RC106$, $C202$, $C203$, $R208$, $R210$, $RC203$ and $RC204$. This nonparametric statistical test has also been useful to know the best combination of operators to be use in the local search phase of the GVNS algorithm. Note that some parameters combinations are not compatible because the time needed to obtain a solution is too high for the company, whose main requirement is to obtain a solution as fast as possible. For example, combination of $M = 40$ and $N = 20$.

Parameter	Value
α	$\in \{\mathbf{2}, 3, 4, 5\}$
β	$\in \{2, \mathbf{3}, 4, 5\}$
γ	$\in \{2, \mathbf{3}, 4, 5\}$
λ	$\in \{2, \mathbf{3}, 4, 5\}$
M	$\in \{5, 10, \mathbf{20}, 30, 40\}$
N	$\in \{1, 5, \mathbf{10}, 15, 20\}$

Table 1: Parameter values used in the GVNS algorithm

4.2 Constraints Effects

Since the problem tackled in this paper takes into account several constraints and the instances in the literature are not prepared to consider all of them together, a set of experiments has been carried out using four test problem instances based on the real data provided by a company in the Canary Islands. These instances, which consist of 100 customers, have been used to show the different behaviours obtained by the algorithm depending on the constraints that are taken into consideration. They are available in https://sites.google.com/site/gciports/vrptw/hfvrptw.

The first experiment reported in this section corresponds to the selection of the principal objective function to be minimized: Total_Distance, Time_Balance or Distance_Balance. On one hand, if the objective function Total_Distance is chosen, the total travelled distance will be optimized, but the balanced time obtained will be worse than the one obtained using Time_Balance as objective function, and the balanced distance will be worse than the one obtained using Distance_Balance as objective function. On the other hand, if either the objective function Time_Balance or

Distance_Balance is chosen, all vehicles will be used, such as required by the company. In case of Time_Balance, the balancing of time among vehicles will be optimized, and in case of Distance_Balance, the balancing of travelled distance will be optimized. For the next experiments, the Total_Distance objective function is used, since it is usually the most common and demanded one. Table 2 summarizes the results obtained from the analysis of the principal objective function. The first two columns indicate the instance under consideration ($N1$ to $N4$), number of postponed services, extra time required by the vehicles, time balance, distance balance, cost, total distance and number of routes in the obtained solutions. Time is expressed in seconds and distance in meters. Last three columns report the data obtained when Total_Distance, Time_Balance and Distance_Balance are considered as principal objective functions, respectively. As shown in Table 2, in general, when an objective function is selected as principal one, the best value of this objective is achieved in each case. Moreover, the best cost is always obtained using the Total_Distance objective function, since the total time spent doing less routes is usually lower.

		Total_Distance	Time_Balance	Distance_Balance
N1	Postponed services	0	0	0
	Extra time	0	0	0
	Time balance	33602	**6779**	23485
	Distance balance	231363	203177	**1261**
	Cost	**5834.08**	7490.67	7407.17
	Distance	$\mathbf{1.0323e+06}$	$2.11398e+06$	$2.16066e+06$
	Routes	7	10	10
N2	Postponed services	0	0	0
	Extra time	0	0	0
	Time balance	17837	**484**	21989
	Distance balance	223827	370569	**7738**
	Cost	**5054.83**	6855.08	7231.36
	Distance	**862046**	$1.94522e+06$	$2.17563e+06$
	Routes	6	10	10
N3	Postponed services	0	0	0
	Extra time	0	0	0
	Time balance	17354	**1256**	49800
	Distance balance	243031	256063	**2331**
	Cost	**5474.00**	7406.53	7893.19
	Distance	$\mathbf{1.03364e+06}$	$1.69382e+06$	$2.14595e+06$
	Routes	6	10	10
N4	Postponed services	0	0	0
	Extra time	0	0	0
	Time balance	14887	**377**	26022
	Distance balance	201456	223874	**6770**
	Cost	**5300.86**	6679.25	7242.97
	Distance	**962762**	$1.98046e+06$	$1.89009e+06$
	Routes	7	10	10

Table 2: Effects of the principal objective function

The second experiment summarized in Table 3 corresponds to the use of a heterogeneous fleet regarding capacity. If we use vehicles with a high capacity (i.e. 500), we will need a small number of vehicles, but when the capacity is lower (i.e. 200), we will need more vehicles, and therefore, more routes to serve all customers. However, when there is a mix of vehicles with different capacities (i.e. a half of 500 and a half of 200, which corresponds to the case of *Mixed Capacities 1* in the table), the result

		High Capacity	Low Capacity	Mixed Capacities 1	Mixed Capacities 2
N1	Postponed services	0	0	0	0
	Extra time	0	0	0	0
	Time balance	16541	45605	38572	44478
	Cost	6099.47	8383.39	6322.69	7694.36
	Distance	**989247**	$1.21453e+06$	$1.0509e+06$	$1.09895e+06$
	Routes	7	11	7	10
N2	Postponed services	0	0	0	0
	Extra time	0	0	0	0
	Time balance	15062	23876	21769	30944
	Cost	5044.47	7829.89	5333.61	5745.78
	Distance	**858015**	$1.34123e+06$	860895	934271
	Routes	6	12	6	9
N3	Postponed services	0	0	0	0
	Extra time	0	0	0	0
	Time balance	37435	19955	21534	26135
	Cost	6146.00	8202.14	5924.33	6824.64
	Distance	**867274**	$1.10491e+06$	879834	961474
	Routes	6	11	7	9
N4	Postponed services	0	0	0	0
	Extra time	0	0	0	0
	Time balance	18032	20459	14304	21417
	Cost	5320.81	8800.92	5668.25	6617.17
	Distance	**969013**	$1.34612e+06$	$1.00986e+06$	$1.06245e+06$
	Routes	7	12	7	10

Table 3: Effects of using a heterogeneous fleet of vehicles with different capacities

will involve the vehicles with a higher capacity and there will be a smaller number of routes. If the number of vehicles with high capacity is reduced, i.e. three vehicles of 500 and the rest of vehicles of 200, which corresponds to the case of *Mixed Capacities 2* in the table, the vehicles with higher capacity will be used first and then the rest of vehicles, and the number of routes will be intermediate. In any case, using vehicles with high capacity is associated with a shorter distance travelled.

The third experiment, which is reported in Table 4, corresponds to the use of a heterogeneous fleet with regard to the working shifts. If the vehicles have restricted working shifts (narrow and multiple time intervals within working shifts) and extra hours are permitted, the results will contain extra hours and more routes than the necessary in the case of homogeneous fleet. Nevertheless, if extra hours are not allowed, but the priorities do allow postponing customers for the next day, the routes will not include the postponed customers. In case of having vehicles with short working shifts and vehicles with large working shifts, the routes will be done by the vehicles with larger working shifts. This involves the lowest number of routes and distance. If we compare the first two cases with restricted windows, we observe that postponing customers when it is possible involves lower travelled distance and cost. However, the number of postponed customers is very small in relation to the total number of customers.

The fourth experiment reported in Table 5 corresponds to the use of narrow and multiple time windows for most of the customers. In this case, if priorities allow postponing customers, the number of postponed services will increase, and additionally the total travelled distance is higher.

		Restricted windows Extra time permitted	Restricted windows Postponing permitted	Mixed windows
N1	Postponed services	0	3	0
	Extra time	2039	0	0
	Time balance	21172	14549	27252
	Cost	6278.89	5907.03	5777.33
	Distance	$1.11156e+06$	$1.02771e+06$	$\mathbf{1.02485e+06}$
	Routes	10	8	**7**
N2	Postponed services	0	2	0
	Extra time	1222	0	0
	Time balance	21846	25974	21549
	Cost	5837.17	5418.86	5382.89
	Distance	$1.45060e+06$	$1.17176e+06$	**856805**
	Routes	10	9	**6**
N3	Postponed services	0	2	0
	Extra time	2073	0	0
	Time balance	11917	26434	31533
	Cost	5246.50	5099.69	6008.39
	Distance	$1.01146e+06$	969548	**870132**
	Routes	8	8	**7**
N4	Postponed services	0	1	0
	Extra time	1038	0	0
	Time balance	21758	28301	15867
	Cost	5998	5904.5	5263.69
	Distance	$1.29676e+06$	$1.14511e+06$	**963579**
	Routes	10	9	**7**

Table 4: Effects of using a heterogeneous fleet of vehicles with different working shifts

		Non-restricted customer windows	Restricted customer windows
N1	Postponed services	**0**	5
	Extra time	0	0
	Time balance	33602	33382
	Cost	5834.08	9265.47
	Distance	$\mathbf{1.0323e+06}$	$1.66452e+06$
	Routes	7	10
N2	Postponed services	**0**	3
	Extra time	0	0
	Time balance	17837	35661
	Cost	5054.83	9829.08
	Distance	**862046**	$1.63362e+06$
	Routes	6	10
N3	Postponed services	**0**	5
	Extra time	0	0
	Time balance	17354	26273
	Cost	5474.00	10676.2
	Distance	$\mathbf{1.03364e+06}$	$1.54976e+06$
	Routes	6	10
N4	Postponed services	**0**	3
	Extra time	0	0
	Time balance	14887	25024
	Cost	5300.86	7086.28
	Distance	**962762**	$1.52592e+06$
	Routes	7	10

Table 5: Effects of using customers with restricted time windows

4.3 Comparative with the literature

This section summarizes the comparison over the standard Solomon instances of 100 customers. Note that the algorithm proposed in this work is thought to solve the FH-

FVRPTW with the inclusion of all the real-world constraints explained in previous sections. 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 6 summarizes the comparative, in which average values of number of vehicles (NV) and traveled distance (TD) are reported. The first column shows the Solomon instance category. Note that in the worst case, the deviation is equal to 7.5987%, and the deviations average is 4.49%. Finally, Tables 7 and 8 show the results obtained by the proposed GVNS for all the standard Solomon instances. The best literature results are the ones reported at Solomon's web page, derived from 23 different algorithms, and these results have been updated with the best results extracted from Hong (2012) and Gong et al. (2012). We compared these results with our best results obtained from 15 executions of each instance. Finally, notice that this paper is not aimed at overcoming the best results for the standard Vehicle Routing Problem with Time Windows, but rather at proposing an intelligent algorithm that works effectively when a fixed heterogeneous fleet is used to solve a real-world problem. However, the GVNS algorithm provides solutions for some instances that overcome the best known literature results. These improvements are marked in bold in the TD column in Tables 7 and 8. The fifth column of these tables shows the time (in seconds) needed to obtain the solutions using the GVNS algorithm. A total average of 12 minutes are needed to obtain the result of a Solomon instance, with an average deviation of 3 minutes.

	NV	TD	Best NV	Best TD	Dev.
C1	10.00	838.45	10.00	828.38	1.22
R1	14.08	1263.07	11.91	1203.16	4.98
RC1	13.62	1409.31	12.00	1345.56	4.74
C2	3.12	632.90	3.00	589.85	7.30
R2	5.00	1019.38	3.00	941.87	8.23
RC2	6.00	1151.08	3.62	1111.99	3.52
Avg.					5.00

Table 6: Computational results of the standard static Solomon instances

Although this has been the solution proposed to the real company, other approaches can be developed in order to improve the average deviation for the standard instances which do not include all the real-world constraints. Until the moment, the used VND process has made movements in or between routes choosing routes and customers randomly. However, it is possible to explore the search space more deeply in order to choose the movement of customers which involves the shortest distance. This process would take much more time, and for this reason it would not be suitable for the real company. Nevertheless, a combination of both processes can be carried out trying to keep similar times. Furthermore, we can stop the search with the first best solution found.

Taking into account all these aspects, three more approaches have been developed. Parameters M and N have been adjusted using Friedman test, among the ones that keep similar times to the original process. For the first tested approach, the relocation movement has been applied in its best first solution version, instead of the

	NV	TD	Dev	Time	Best NV	Best TD
C101	10	828.94	0.00	882.99	10	828.94
C102	10	828.94	0.00	671.19	10	828.94
C103	10	841.87	1.64	495.17	10	828.06
C104	10	877.48	6.01	371.72	10	824.78
C105	10	828.94	0.00	576.09	10	828.94
C106	10	828.94	0.00	594.53	10	828.94
C107	10	828.94	0.00	587.57	10	828.94
C108	10	828.94	0.00	737.72	10	828.94
C109	10	853.10	0.00	443.94	10	828.94
average	10.00	838.45	1.16	595.66	10.00	828.38
R101	21	1714.83	1.16	825.56	18	1612.29
R102	19	1536.73	5.98	805.46	16	1473.41
R103	16	1351.30	4.12	480.30	12	1279.37
R104	12	1034.33	5.32	428.76	9	1007.24
R105	15	1446.98	2.62	543.20	14	1377.11
R106	14	1325.66	4.83	647.24	12	1251.98
R107	13	1173.22	5.56	551.50	10	1104.66
R108	10	1001.45	5.84	604.66	9	960.88
R109	14	1231.90	4.05	741.58	12	1179.73
R110	13	1171.45	4.23	814.31	11	1113.10
R111	12	1136.10	4.98	529.36	10	1096.72
R112	10	1032.96	3.47	679.97	10	981.46
average	14.08	1263.07	4.67	637.66	11.91	1203.16
RC101	17	1704.13	3.69	634.24	15	1641.20
RC102	14	1702.39	3.68	944.86	13	1447.14
RC103	13	1502.39	8.18	769.88	11	1261.67
RC104	11	1374.14	6.81	443.73	10	1135.48
RC105	17	1218.49	5.67	657.47	13	1502.48
RC106	14	1592.39	2.39	685.49	12	1406.25
RC107	12	1440.72	3.06	674.63	11	1230.48
RC108	11	1269.29	2.79	916.62	11	1139.82
average	13.62	1409.31	4.53	715.86	12.00	1345.57

Table 7: Computational results of the standard static Solomon instances

original VND. Parameters N and M have been set to 5 and 7, respectively. Results are shown in Table 4.3. The average deviation decrease from 5.00 to 2.55 using this approach, demonstrating that it is possible to improve the behaviour with the standard instances in the literature which do not include all the real-world constraints.

The second tested approach applies the relocation movement in its greedy version and the original VND half and half. Parameters N and M have been set to 2 and 9 respectively. This way, results in Table 9 are obtained, where the average deviation is reduced to 1.49 for the standard instances. These results are better than the ones obtained if the relocation movement, in its greedy version, is applied only at some steps of the GVNS, for example, when $k = 3$ or $k = 6$ in the shaking process. In this case, parameters N and M have been set to 3 and 9 respectively. Table 10 depicts these results, where the average deviation is 3.55.

Therefore, the best approach to use with the standard instances in the literature is the one that combines the original VND proposed in this paper and the greedy version of the relocation movement, half and half.

	NV	TD	Dev	Time	Best NV	Best TD
C201	3	591.56	0.00	1227.49	3	591.56
C202	3	593.00	0.24	1024.55	3	591.56
C203	4	645.11	8.36	429.08	3	591.17
C204	3	739.08	20.09	485.42	3	590.60
C205	3	609.76	3.42	1102.27	3	588.88
C206	3	627.08	6.15	1231.16	3	588.49
C207	3	621.04	5.27	1083.53	3	588.29
C208	3	636.59	7.58	1824.30	3	588.32
average	3.12	632.90	6.39	1050.97	3.00	589.86
R201	8	1273.71	1.68	1046.62	4	1252.37
R202	5	1104.19	-7.93	673.37	3	1191.70
R203	5	1024.89	8.33	662.63	3	939.54
R204	4	919.30	12.34	592.82	3	801.68
R205	5	1096.91	9.34	1359.37	3	994.42
R206	4	1033.68	12.34	672.00	3	906.14
R207	6	976.61	13.60	721.40	3	843.77
R208	4	836.33	13.10	678.42	2	726.75
R209	5	982.15	7.43	569.57	3	909.16
R210	5	1033.85	9.14	720.96	3	939.34
R211	4	931.62	8.14	827.22	3	855.76
average	5.00	1019.38	8.00	774.94	3.00	941.88
RC201	8	1359.16	-2.94	494.59	5	1399.16
RC202	7	1236.83	-10.53	756.81	3	1367.09
RC203	5	1059.39	0.92	653.92	3	1049.62
RC204	5	937.70	18.00	643.13	3	798.41
RC205	7	1232.68	-2.98	587.91	5	1269.38
RC206	6	1231.54	8.81	758.65	4	1122.99
RC207	6	1140.10	6.93	630.69	3	1061.14
RC208	4	975.22	15.08	1962.08	3	828.14
average	6.00	1151.08	4.16	810.97	3.62	1111.99

Table 8: Computational results of the standard static Solomon instances

	NV	TD	Best NV	Best TD	Dev.
C1	10.00	837.28	10.00	828.38	1.07
R1	13.91	1233.55	11.91	1203.16	2.53
RC1	13.50	1395.71	12.00	1345.56	3.73
C2	3.25	619.32	3.00	589.85	5.00
R2	5.90	975.47	3.00	941.87	3.57
RC2	6.25	1105.87	3.62	111.99	-0.55
Avg.					2.55

Table 9: Computational results of the standard static Solomon instances applying first best solution approach

Finally, it is important to note that the number of vehicles used is not important for the real company referenced in this paper, since it has a fixed fleet of vehicles with a fixed number of drivers. However, if the number of vehicles had to be reduced, a number of routes reduction algorithm would have to be applied before the GVNS algorithm. This process usually involves a high increase of computational time, and for this reason it is not suitable for the real company, whose main requirement is

	NV	TD	Best NV	Best TD	Dev.
C1	10.00	833.62	10.00	828.38	0.63
R1	13.83	1230.08	11.91	1203.16	2.24
RC1	13.62	1395.82	12.00	1345.56	3.74
C2	3.12	611.32	3.00	589.85	3.64
R2	5.27	956.27	3.00	941.87	1.53
RC2	6.12	1080.76	3.62	111.99	-2.81
Avg.					1.49

Table 10: Computational results of the standard static Solomon instances applying best solution and original VND half and half

	NV	TD	Best NV	Best TD	Dev.
C1	10.00	849.92	10.00	828.38	2.60
R1	14.08	1236.74	11.91	1203.16	2.79
RC1	13.63	1404.58	12.00	1345.56	4.39
C2	3.13	629.78	3.00	589.85	6.77
R2	5.64	987.20	3.00	941.87	4.81
RC2	6.13	1111.60	3.62	1111.99	-0.04
Avg.					3.55

Table 11: Computational results of the standard static Solomon instances applying best solution for $k = 3$ and $k = 6$

	NV	TD	Best NV	Best TD	Dev.
C1	10.00	841.37	10.00	828.38	1.57
R1	13.08	1224.68	11.66	1204.93	1.64
RC1	12.87	1405.49	11.50	1368.29	2.72
C2	3.12	621.50	3.00	589.86	5.36
R2	3.36	1079.87	2.73	951.91	13.44
RC2	4.12	1245.58	3.25	1119.35	11.28
Avg.					6.00

Table 12: Computational results of the standard static Solomon instances applying routes reduction algorithm

to obtain a fast solution with a high quality. Nevertheless, the reduction of routes process proposed by Nagata and Bräysy (2009) has been implemented in order to check this option. For this purpose, the parameter setting has been made taking into account that both together, reduction of routes algorithm and the original algorithm proposed in this paper, do not overpass the times of the original approach. Results for the standard static Solomon instances are shown in Table 11. Distance deviations are not good enough due to the short computational time which can be used. The time needed for both the reduction of routes algorithm and the original GVNS algorithm to obtain high quality solutions need to be increased, but, as said before, the real company requires a quick algorithm and has a fixed fleet of vehicles. Therefore, this reduction of routes has not been included in the final algorithm. Note that some best known results are different in the last table. In the related literature there are results

for the standard instances which involve less number of routes although the total distances increase. These results have been used in this last table, because we are applying a process of reduction of routes and it is more suitable.

4.4 Results for real instances

This section is devoted to show results corresponding to real data. In the first place, the solutions provided by the proposed GVNS are compared with the solution implemented by a company for a real instance, which consists of 109 customers. Table 12 summarizes the best solutions obtained by GVNS when considering the total distance objective function, and the best solution implemented by the company. Moreover, the time/distance balance associated to the solutions are also reported. The first row of the table corresponds to the best solution obtained by GVNS if the total distance is considered as objective function. The second row shows the solution obtained by GVNS consisting of 6 routes, which is the number of vehicles of the company solution. The third row shows the company solution.

From these results, we may conclude that the solution implemented by the company is worst than the solutions obtained by GVNS if the total distance objective and the number of routes are taken into consideration. Note that our second obtained solution is better in total distance, time and distance balance than the one that had been implemented by the company, and our first obtained solution provides the lowest total distance although it presents the worst time balance.

	Distance	Time balance	Distance balance	Nr. routes
$GVNS$	852.7	37200	196.5	7
$GVNS$	858.9	12840	125.3	6
$Real\_Sol$	1247.3	14040	228.5	6

Table 13: Best solutions reached for the real instance using the total distance as principal objective function

Finally, since the image corresponding to the solution $N1$ is confusing due to the large number of routes, the solution to an instance based on $N1$ consisting of 50 customers is shown in Figure 2, in which its 5 routes are depicted over the map of the island of Tenerife.

5 Conclusions

This work tackles a real-world rich vehicle routing problem proposed by a company in Spain that combines multiple attributes together, which aim to better consider the specificities of real applications. Particularly, it has been developed an efficient solution method to solve the Fixed Heterogeneous Fleet Vehicle Routing Problem with Time Windows ($FHFVRPTW$). The attributes considered by the company consist of a fixed heterogeneous fleet of vehicles, soft and multiple time windows for customers, soft and multiple time intervals in the working shifts for vehicles, customers

priorities and vehicle-customer constraints, which to the best of our knowledge have not been taken into account all together in the literature. Given the fact that the company might require to either consider or not all these attributes, the fleet management system allows deactivating any of the mentioned attributes. Since exact algorithms are not applicable when solving real-life large vehicle routing problem instances, the focus of this work is put on the use of metaheuristic procedures. Particularly, this paper proposes a General Variable Neighbourhood Search ($GVNS$) metaheuristic, which is able to obtain high quality solutions that are valid for the company. The computational experience carried out in this work, which includes the analysis of the effect of the different attributes taken into consideration, the comparative with the literature for the standard vehicle routing problem with time windows, and the study of the solutions provided by the algorithm when compared with the solutions implemented by the company, corroborate the effectiveness of the developed software. It is worth to mention that the algorithm has been integrated into a fleet management system and several tests with real companies have been conducted. Finally, the next phase of this work is to dynamically route new customers which appear over the planning horizon. It means that, once the initial routes have been specified and vehicles start to make their work, new customers can ask for service. At this point real time communication between vehicles and the fleet management system is required.

Acknowledgements 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).

References

Baldacci, R., Battarra, M., and Vigo, D. (2008). Routing a heterogeneous fleet of vehicles. In *The vehicle routing problem: latest advances and new challenges*, pages 3–27. Springer.

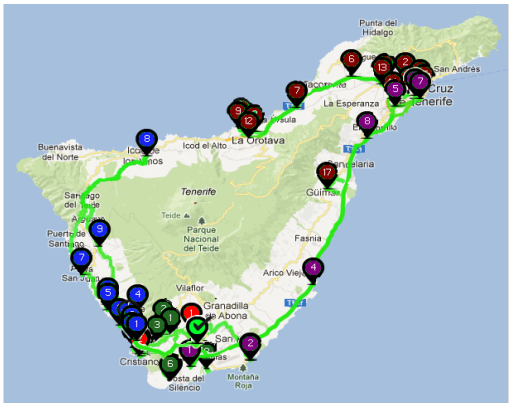

Fig. 1: Some routes corresponding to a real instance

Baldacci, R. and Mingozzi, A. (2009). A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, 120(2):347–380.

Baldacci, R., Toth, P., and Vigo, D. (2007). Recent advances in vehicle routing exact algorithms. *4OR*, 5(4):269–298.

Borgulya, I. (2008). An algorithm for the capacitated vehicle routing problem with route balancing. *Central European Journal of Operations Research*, 16(4):331–344.

Brandão, J. (2011). A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. *Computers & Operations Research*, 38(1):140–151.

Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 15:347–368.

Bräysy, O. and Gendreau, M. (2005). Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation science*, 39(1):119–139.

Calvete, H. I., Gale, C., Oliveros, M.-J., and Sanchez-Valverde, B. (2007). A goal programming approach to vehicle routing problems with soft time windows. *European Journal of Operational Research*, 177(3):1720–1733. 6th International Conference on Multiple Objective Programming and Goal Programming, Hammamet, TUNISIA, APR 04-06, 2004.

Cassani, L. and Righini, G. (2004). Heuristic algorithms for the tsp with rear-loading. In *35th Annual Conference of the Italian Operations Research Society (AIRO XXXV), Lecce, Italy*.

Croes, G. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812.

Daniel, W. W. (1990). *Applied nonparametric statistics*. PWS-Kent Publishing Company, Boston.

Eksioglu, B., Vural, A. V., and Reisman, A. (2009). Survey: The vehicle routing problem: A taxonomic review. *Comput. Ind. Eng.*, 57(4):1472–1483.

Fleszar, K., Osman, I., and Hindi, K. (2008). A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, 195(3):803–809.

Gendreau, M., Hertz, A., and Laporte, G. (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research*, 40(6):1086–1094.

Gendreau, M., Potvin, J.-Y., Bräumlaysy, O., Hasle, G., and Løkketangen, A. (2008). *Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography*. Springer.

Gendreau, M. and Tarantilis, C. D. (2010). *Solving large-scale vehicle routing problems with time windows: The state-of-the-art*. CIRRELT.

Ghoseiri, K. and Ghannadpour, S. F. (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing*, 10(4):1096–1107.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549.

Goel, A. and Gruhn, V. (2008). A general vehicle routing problem. *European Journal of Operational Research*, 191(3):650–660.

Golden, B. L., Raghavan, S., and Wasil, E. A. (2008). *The vehicle routing problem: latest advances and new challenges*, volume 43. Springer.

Gong, Y.-J., Zhang, J., Liu, O., Huang, R.-Z., Chung, H.-H., and Shi, Y. (2012). Optimizing the Vehicle Routing Problem With Time Windows: A Discrete Particle Swarm Optimization Approach. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(2):254–267.

Hansen, P., Mladenovic, N., Brimberg, J., and Moreno Pérez, J. (2010a). *Handbook of Metaheuristics*, chapter Variable Neighborhood Search, pages 61–86. Springer.

Hansen, P., Mladenovic, N., and Moreno-Pérez, J. (2008). Variable neighborhood search. *European Journal of Operational Research*, 191(3):593–595.

Hansen, P., Mladenovic, N., and Moreno Pérez, J. A. (2010b). Variable neighbourhood search: methods and applications. *Annals of Operations Research*, 175(1):367–407.

Hemmelmayr, V., Doerner, K., and Hartl, R. (2009). A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802.

Hoeller, H., Melian, B., and Voss, S. (2008). Applying the pilot method to improve VNS and GRASP metaheuristics for the design of SDH/WDM networks. *European Journal of Operational Research*, 191(3):691–704.

Hong, L. (2012). An improved LNS algorithm for real-time vehicle routing problem with time windows. *Computers & Operations Research*, 39(2):151–163.

Hong, S. and Park, Y. (1999). A heuristic for bi-objective vehicle routing with time window constraints. *International Journal of Production Economics*, 62(3):249–258.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T., and Yagiura, M. (2005). Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation Science*, 39(2):206–232.

Ibaraki, T., Imahori, S., Nonobe, K., Sobue, K., Uno, T., and Yagiura, M. (2008). An iterated local search algorithm for the vehicle routing problem with convex time penalty functions. *Discrete Applied Mathematics*, 156(11):2050–2069.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2007). From single-objective to multi-objective vehicle routing problems: Motivations, case studies and methods. In Golden, B., Raghavan, S., and Wasil, E., editors, *The vehicle routing Problem: Latest Advances and New Challenges*. Springer.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293–309.

Kytöjoki, J., Nuortio, T., Brysy, O., and Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9):2743 – 2757.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416.

Li, F., Golden, B., and Wasil, E. (2007). A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, 34(9):2734–2742.

Melian, B. (2006). Using memory to improve the VNS metaheuristic for the design of SDH/WDM networks. In Almeida, F and Aguilera, MJB and Blum, C and Vega,

JMM and Perez, MP and Roli, A and Sampels, M, editor, *Hybrid Metaheuristics, Proceedings*, volume 4030 of *Lecture Notes in Computer Science*, pages 82–93. 3rd International Workshop on Hybrid Metaheuristics, Gran Canaria, SPAIN, OCT 13-14, 2006.

Melin-Batista, B., De Santiago, A., Angelbello, F., and Alvarez, A. (2014). A bi-objective vehicle routing problem with time windows: A real case in tenerife. *Applied Soft Computing Journal*, 17:140–152.

Moreno-Vega, J. M. and Melian, B. (2008). Introduction to the special issue on variable neighborhood search. *Journal of Heuristics*, 14(5):403–404.

Nagata, Y. and Bräysy, O. (2009). A powerful route minimization heuristic for the vehicle routing problem with time windows. *Operations Research Letters*, 37(5):333–338.

Or, I. (1976). *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking*. Xerox University Microfilms.

Paraskevopoulos, D., Repoussis, P., Tarantilis, C., Ioannou, G., and Prastacos, G. (2008). A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows. *Journal of Heuristics*, 14(5):425–455.

Pardo, E., Mladenovi, N., Pantrigo, J., and Duarte, A. (2013). Variable formulation search for the cutwidth minimization problem. *Applied Soft Computing Journal*, 13(5):2242–2252.

Penna, P. H. V., Subramanian, A., and Ochi, L. S. (2011). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, pages 1–32.

Polacek, M., Hartl, K., Doerner, K., and Reimann, M. (2004). A variable neighborhood search for the multi depot vehicle routing problem with time windows. *Journal of Heuristics*, 10(6):613 – 627.

Potvin, J.-Y. (2009). State-of-the art review - evolutionary algorithms for vehicle routing. *INFORMS Journal on Computing*, 21(4):518–548.

Prins, C. (2009). Two memetic algorithms for heterogeneous fleet vehicle routing problems. *Engineering Applications of Artificial Intelligence*, 22(6):916–928.

Rahoual, M., Kitoun, B., Mabed, M., Bachelet, V., and Benameur, F. (2001). Multi-criteria genetic algorithms for the vehicle routing problem with time windows. In *Metaheuristic International Conference (MIC'2001)*, pages 527–532.

Repoussis, P. P., Paraskevopoulos, D. C., Tarantilis, C. D., and Ioannou, G. (2006). A reactive greedy randomized variable neighborhood tabu search for the vehicle routing problem with time windows. In Almeida, F and Aguilera, MJB and Blum, C and Vega, JMM and Perez, MP and Roli, A and Sampels, M, editor, *Hybrid Metaheuristics, Proceedings*, volume 4030 of *Lecture Notes in Compputer Science*, pages 124–138. 3rd International Workshop on Hybrid Metaheuristics, Gran Canaria, Spain, Oct 13-14, 2006.

Schmid, V., Doerner, K. F., and Laporte, G. (2013). Rich routing problems arising in supply chain management. *European Journal of Operational Research*, 224(3):435–448.

Solomon, M. (1987). Algorithms for the Vehicle-Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2):254–265.

Subramanian, A., Vaz Penna, P. H., Uchoa, E., and Ochi, L. S. (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*.

Taillard, E. (1999). A heuristic column generation method for the heterogeneous fleet vrp. *Rairo-Recherche Operationnelle-Operaions Research*, 33(1):1–14.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31(2):170–186.

Tarantilis, C. D., Zachariadis, E. E., and Kiranoudis, C. T. (2009). A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem. *Intelligent Transportation Systems, IEEE Transactions on*, 10(2):255–271.

Toth, P. and Vigo, D. (2002). The vehicle routing problem. society for industrial and applied mathematics. *SIAM Monographs on Discrete Mathematics and Applications*.

Vidal, T., Crainic, T., Gendreau, M., and Prins, C. (2013). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, In press.