J Heuristics (2008) 14: 425–455
DOI 10.1007/s10732-007-9045-z

A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows

D.C. Paraskevopoulos · P.P. Repoussis · C.D. Tarantilis · G. Ioannou · G.P. Prastacos

Received: 7 April 2006 / Revised: 26 October 2006 / Accepted: 12 January 2007 /
Published online: 23 October 2007

Abstract This paper presents a solution methodology for the heterogeneous fleet vehicle routing problem with time windows. The objective is to minimize the total distribution costs, or similarly to determine the optimal fleet size and mix that minimizes both the total distance travelled by vehicles and the fixed vehicle costs, such that all problem's constraints are satisfied. The problem is solved using a two-phase solution framework based upon a hybridized Tabu Search, within a new Reactive Variable Neighborhood Search metaheuristic algorithm. Computational experiments on benchmark data sets yield high quality solutions, illustrating the effectiveness of the approach and its applicability to realistic routing problems.

Keywords Variable neighborhood search · Logistics · Vehicle scheduling

1 Introduction

Variable Neighborhood Search (VNS) is a recently proposed metaheuristic for solving combinatorial and global optimization problems. Its basic idea is the systematic change of neighborhoods within local search (Hansen and Mladenović 2001). VNS-based metaheuristics have been successfully applied to a variety of problems and real-life applications, e.g., the p-median, the traveling salesman, and other problems (García-López et al. 2002). Recently, Gendreau and Potvin (2005) mention Bent and van Hentenryck (2004), Cordone and Calvo (2001), Rouseau et al. (2002), Polacek et al. (2004) and Bräysy (2001, 2003) as the most recent, closely related to

This work is supported by the General Secretariat for Research and Technology of the Hellenic Ministry of Development under contract GSRT NM-67.

D.C. Paraskevopoulos · P.P. Repoussis · C.D. Tarantilis (✉) · G. Ioannou · G.P. Prastacos
Management Science Laboratory, Department of Management Science & Technology, Athens University of Economics & Business, 28 Hydras st., 11362, Athens, Greece
e-mail: tarantil@aueb.gr

VNS, solution methodologies developed to address successfully vehicle routing problems. Motivated by this trend, this paper presents a Reactive Variable Neighborhood Tabu Search (ReVNTS) hybrid metaheuristic approach for solving the Heterogeneous Fleet Vehicle Routing Problem with Time Windows (HFVRPTW).

The Vehicle Routing Problem (VRP) is a focal problem of distribution management within the area of service operations and supply chain management. The VRP intensifies from a practical perspective when the vehicle fleet is heterogeneous (HFVRP), i.e., vehicles differ in their equipment, capacity, age or cost (Taillard 1999). This problem is also abbreviated in literature as VRPHE or HVRP, and mainly arises when the dispatcher wishes to revise the fleet composition to better suit customer needs (Tarantilis et al. 2003). Thus, vehicles of different capacity allow the scheduler to maximize utilization by deploying the appropriate vehicle in areas with a respective concentration of customers. Furthermore, using a heterogeneous fleet, it is also possible to service customers requiring small vehicles because of accessibility restrictions (Semet 1995).

The HFVRP involves the design of a set of minimum cost routes, each originating and terminating at the depot, using a heterogeneous fleet of vehicles with various capacities, fixed and variable costs, to service a set of customers with known demands. Each customer must be serviced only once by exactly one vehicle, while the total demand of a route must not exceed the capacity of the vehicle type assigned to it. The routing or distribution cost of a vehicle derive from the sum of its fixed cost and a variable cost proportional to the distance traveled. Given these specifications, three major variants of HFVRP appear in literature. The first one is known as the Fleet Size and Mix VRP (FSMVRP) (Golden et al. 1984), the second as the Heterogeneous Fixed Fleet VRP (HFFVRP) (Tarantilis et al. 2003) and the third as the Heterogeneous Fleet VRP with variable unit running costs (HVRPRC) (Salhi et al. 1992). Although, the above notation is often used in literature to describe these different VRPs, we propose some standardized suffixes in order to "construct" these abbreviations. In particular, suffixes (TW), (HF), (RC), (F) and (U) will indicate the time windows, the heterogeneous nature of the fleet, the consideration of variable unit running costs and the type of the heterogeneous fleet as fixed or unlimited, respectively. Thus, for instance, the abbreviation of the Fleet Size and Mix VRP with variable unit running costs is constructed in the following manner as HF-U-VRP-RC, which is a special case of HFVRP.

Considering the HFUVRP, there are various types of vehicles of unlimited availability. The objective is to determine the fleet of vehicles such that the sum of fixed and variable costs is minimized, or similarly to determine the optimal fleet composition with minimum overall distribution costs (Golden et al. 1984). On the other hand, the HFFVRP generalizes HFUVRP by limiting the number of available vehicles of each type that can be deployed (Taillard 1999). From an operational viewpoint, in case of HFUVRP the fixed cost of a vehicle represents leasing or hiring costs, while in the case of HFFVRP the fixed cost of a vehicle can be viewed as the set-up cost on a daily basis; in cases where a company has its own vehicle fleet the fixed cost is usually related to the depreciation or the maintenance costs over a period (Desrochers and Verhoog 1991). Finally, in case of HFVRP with variable unit running costs the type of heterogeneous fleet can be either fixed or unlimited, however, the traveling costs depend on the vehicle type (Choi and Tcha 2005).

In this paper, we propose a solution methodology for the HFVRPTW that can address both the Fleet Size and Mix VRP with Time Windows and the Heterogeneous Fixed Fleet VRP with Time Windows variants of the problem, denoted as HFUVRPTW and HFFVRPTW respectively, with uniform unit running costs for all vehicle types. Recently, the solution methods proposed for homogeneous routing problems, i.e., the capacitated VRP and the VRP with Time Windows (VRPTW), have substantially progressed (Tarantilis and Kiranoudis 2002; Tarantilis 2005). However, the HFVRP, and in particular the time windows variants, has attracted much less attention. To this end, research related to the HFVRPTW is limited, while most of the research approaches tackle only the HFUVRPTW. Note that the HFFVRPTW, according to our survey, has not been addressed before.

Golden et al. (1984) were among the first to address the HFUVRP and proposed a set of benchmark data sets. In particular, five adaptations of Clarke and Wright (1964) savings algorithm were suggested and some two-step "route first- cluster second" methods, called "Giant Tour Algorithms", were developed. First, a solution is initialized by generating a single tour that visits all customers, like a traveling salesman tour, and then this "giant tour" is partitioned into sub-tours, each satisfying the problem constraints. Other heuristic methods proposed for the HFUVRP are those of Geysens et al. (1984, 1986), Desrochers and Verhoog (1991) and Salhi and Rand (1993). More specifically, Geysens et al. (1984, 1986) present two heuristic approaches. The first incorporates vehicle capacity constraints to the objective function by using penalty multipliers and the second is a two-stage algorithm. Finally, Desrochers and Verhoog (1991) developed matching based savings heuristics based on successive route fusions, while Salhi and Rand (1993) developed a multi-phase approach, employing perturbation procedures to improve vehicle use and fleet utilization.

In addition to heuristic approaches, several more sophisticated metaheuristic and other methods have been proposed in the literature for the HFUVRP. Semet and Taillard (1993), Rochat and Semet (1994) and Osman and Salhi (1996) proposed Tabu Search approaches. More recently, Gendreau et al. (1999) developed a tabu search based metaheuristic, which consisted of the following components: a generalized insertion heuristic called GENIUS, several strategies to diversify the search and Adaptive Memory Programming. On the other hand, Renaud and Boctor (2002) proposed a two-phase sweep based algorithm. In the first phase, the proposed algorithm generates several routes served by one or two vehicles. In the second phase, the selection of routes and vehicles to be used is made by solving a special structured set-partitioning problem. Finally, Brandäo and Mercer (1997) proposed an enhanced tabu search heuristic for a real life HFUVRP application where various practical constraints were considered.

For the HFFVRPRC, Taillard (1999), Tarantilis et al. (2003, 2004) and Li et al. (2006) offered the latest algorithmic developments. Taillard (1999) presented a robust heuristic column generation method, based on Adaptive Memory Programming, which was used initially to solve a series of homogeneous VRPs for each vehicle type without any limitations. The tours of the homogeneous VRP solutions were then combined to produce HFVRP solutions, which were subsequently improved. On the other hand, Tarantilis et al. (2003, 2004) proposed backtracking and list based threshold accepting metaheuristics, called BATA and LBTA respectively. More recently, Li

et al. (2006) proposed a two-level record-to-record travel algorithm which is a deterministic variant of Simulated Annealing. Finally, Tarantilis and Kiranoudis (2005) proposed an adaptive memory-based algorithm for two real life HFFVRPRC case studies from the dairy and construction sectors.

All the aforementioned approaches solved the HFVRP without considering time windows constraints. Liu and Shen (1999a, 1999b) proposed the first construction and improvement heuristics for the HFUVRPTW. In particular, a number of parallel insertion based savings heuristics capable of generating feasible solutions of good quality were developed, utilizing the generic construction insertion scheme of Solomon (1987). Furthermore, in order to take into account possible savings in vehicle acquisition costs, the savings criteria of Golden et al. (1984) were modified. Liu and Shen (1999a) also introduced a new set of benchmark problem instances for the HFUVRPTW. In a manner similar to Golden et al. (1984), the characteristics of the heterogeneous fleet were produced, utilizing the well known 100 customer data sets of Solomon (1987) as proposed for the VRPTW.

Recently, Dullaert et al. (2002) proposed sequential construction heuristics extending Solomon (1987) and Golden et al. (1984) customer selection and vehicle insertion criteria. Furthermore, Dullaert et al. (2002) provides an analysis on the heterogeneous vehicle fleet characteristics, proposed by Liu and Shen (1999a), with respect to real life applications. Finally, Dell-Amico et al. (2006) proposed a multi start parallel regret construction heuristic enhanced with several cost metric functions and next customer selection criteria, embedded within a Ruin-and-Recreate metaheuristic framework. In particular, several vehicle reassignment procedures (combination and splitting) were proposed within the Ruin-and-Recreate phases, driven by the heterogeneous nature of the vehicles.

The main contribution of this paper is the development of an efficient and robust methodology for solving the HFVRPTW, considering both HFUVRPTW and HFFVRPTW variants of the problem with uniform unit running costs. The proposed method consists of a two-phase multi-start metaheuristic solution framework, called ReVNTS. In the first phase, several initial solutions are produced using a new semi-parallel construction heuristic followed by a sophisticated ejection chain approach, which improves the vehicles' capacity utilization and reduces the number of vehicles. In the second phase, the solutions are improved using a Variable Neighborhood Tabu Search (VNTS) hybrid metaheuristic algorithm. The proposed implementation uses Tabu Search (TS) internally within a reactive Variable Neighborhood Search (VNS) allowing the efficient and effective exploration of the solution space. Furthermore, a specialized shaking mechanism is introduced, which utilizes the information provided by local optimum solutions delivered during VNS iterations, in order to diversify the search towards more promising regions.

The reminder of this paper is organized as follows. Section 1 defines HFVRPTW and introduces the necessary notation for both variants of the problem addressed herein. Section 2 introduces the solution methodology and its main components. The semi-parallel construction heuristic is presented first, followed by, the route elimination procedure; then, the Reactive VNTS hybrid metaheuristic is described, along with an analysis of the specialized shaking mechanism. Computational experiments assessing the proof of concept and the value of the suggested approach, along with

a comparative performance analysis, is presented in Sect. 3. The paper concludes in Sect. 4 offering pointers for further research.

2 Problem description and notation

The HFVRPTW can be defined on a complete graph $G = (V, A)$, where $V = \{0, 1, \ldots, n+1\}$ is the node set, $A = \{(i, j) : 0 \leq i, j \leq n, i \neq j\}$ is the arc set and the depot is represented by the two nodes 0 and $n+1$. There is also a set of different vehicle types K. A vehicle of type $k \in K$ has a carrying capacity Q_k, while the number of vehicles of type k is z_k. Each vehicle originates and terminates at the depot while services a subset of customers. Thus, feasible vehicle routes must correspond to paths in G and should start from node 0 and end at node $n+1$. Each customer i ($i = 1, 2, \ldots, n$), where $n = |V| - 2$, has a fixed nonnegative demand q_i and poses a time window interval $[e_i, l_i]$ that represents the earliest and latest time during the planning horizon that service of customer i can take place.

The arrival time of a vehicle must be within the associated time window, and the vehicle must remain at the customer's location for s_i time instants (service time). If a vehicle arrives earlier than e_i at customer i, it is allowed to wait until time e_i. Furthermore, a time window is also associated with nodes 0 and $n+1$, i.e., $[e_0, l_0] = [e_{n+1}, l_{n+1}] = [E, L]$, where E and L represent the earliest possible departure from the depot and the latest possible return at the depot, respectively. Finally, the demands and service times for these two nodes are equal to zero ($q_0 = q_{n+1} = s_0 = s_{n+1} = 0$).

There is a nonnegative travel cost c_{ij}^k, a travel time t_{ij} and a distance h_{ij} associated with the path from customer i to customer j, that is the arc (i, j) of set A. Note that in case of uniform unit running costs for all vehicle types, the travel costs are the same for all vehicles ($c_{ij}^k = c_{ij}^{\hat{k}}, \forall k, \hat{k} \in K$). Furthermore, there is a fixed cost w_k relevant to the activation of a vehicle $k \in K$. As mentioned earlier, in case of HFUVRPTW, w_k usually represents the hiring cost of a vehicle k, while in HFFVRPTW the latter can be viewed as the set-up cost.

Similarly to the multi-commodity network flow formulation of VRPTW (Cordeau et al. 2002), the HFVRPTW requires two groups of variables. The first group is the flow variables x_{ij}^k that model the sequence in which vehicles visit customers, and is defined as follows; 1 if customer i precedes j visited by vehicle k, 0 otherwise. The second group of variables, denoted as a_{ik}, specifies the arrival time or the start time of service at customer i when serviced by vehicle type k.

A feasible HFVRPTW solution exists only if all customers are served by exactly one vehicle of the available fleet composition, and furthermore, each vehicle route satisfies time windows and vehicle capacity constraints. Initially, time window constraints state that $e_i \sum_{j \in \Delta^+(i)} x_{ij}^k \leq a_{ik} \leq l_i \sum_{j \in \Delta^+(i)} x_{ij}^k$ for all $k \in K$ and $i \in N$, $x_{ij}^k (a_{ik} + s_i + t_{ij} - a_{jk}) \leq 0$ for all $k \in K$ and $(i, j) \in A$, and $E \leq a_{ik} \leq L$ for all $k \in K$ and $i \in \{0, n+1\}$, where $\Delta^+(i)$ denotes the set of nodes j such that arc $(i, j) \in A$ and $N = V \setminus \{0, n+1\}$ represents the set of customers. On the other hand, capacity constraints state that for each vehicle route the accumulated service up to any customer must not exceed vehicles' capacity, such that $\sum_{i \in N} q_i \sum_{j \in \Delta^+(i)}^{n} x_{ij}^k \leq Q_k$ for all $k \in K$. Finally, fleet composition restrictions state that $\sum_{j \in N} x_{0j}^k \leq z_k$ for

all $k \in K$. Recall, that the number of available vehicle types z_k is either set to ∞ for all $k \in K$ in case of HFUVRPTW or it is limited to an upper bound in case of HFFVRPTW.

Given the above conditions, the objective of HFVRPTW is to minimize the total distribution costs, or similarly to determine the optimal fleet size and mix that minimizes both the total travelling cost by the vehicles and the fixed vehicle costs, such that all problem's constraints are satisfied.

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k x_{ij}^k + \sum_{k \in K} w_k \sum_{j \in N} x_{0j}^k. \tag{1}$$

The objective function 1 models the trade-off between variable routing and fixed vehicle costs. The first term represents the total travelling cost of the routes followed by all vehicles, while the second term reflects the total set-up and/or acquisition costs emerge from the use of particular vehicle types. It is worth mentioning that the travel cost can either represent the arc distances among customers (total distance travelled) or represent the travel time, i.e., the distance travelled plus the waiting time, as suggested by Liu and Shen (1999b). More specifically, the total travelling cost can be seen as the sum of all the times spent along the route (travelling time, service time and possible waiting time). During all computational experiments, presented in subsequent section, the total travelling cost, as described above, is minimized, excluding the total service time from the cost computations that is constant for all feasible solutions.

3 Solution methodology

A two-phase approach is proposed to tackle the HFVRPTW. In the first phase several initial solutions are produced using a semi-parallel construction heuristic for different combinations of parameter settings. Next, a route elimination procedure is applied in order to reduce the number of vehicle routes and improve the vehicle's utilization using an enhanced ejection-chain heuristic, and finally, a subset of high quality solutions is selected for further improvement. In the second phase, the objective is to minimize the total distribution cost using a Reactive Variable Neighborhood Tabu Search (ReVNTS) hybrid metaheuristic method. Within ReVNTS a specialized mechanism is also incorporated which exploits information gathered during the search in a reactive fashion.

Variable Neighborhood Search was originally proposed by Hansen and Mladenović (2001), and is based upon a basic principle: systematic change of neighborhood during the search. Although, VNS treats a single solution at each iteration, the trajectory followed during the search of the solution space is discontinuous. Indeed, it explores increasingly distant neighborhoods of a current solution, and moves at random from this solution to a new one (shaking). In this way, favorable characteristics of a solution will often be kept and used to obtain promising neighboring solutions (Hansen and Mladenović 2001).

The basic VNS solution framework mainly consists of three phases: shaking, local search and move. At the initialization step, a set of neighborhood structures N_y is

defined, where y denotes the neighborhood index. In the *shaking* phase a solution s' in the yth neighborhood of the current solution s is randomly selected. *Local search* is then applied in order to find the best neighbor s'' of s'. If $f(s'') < f(s)$, s is replaced by s''. The latter is the so called *move* phase. The above scheme is repeated from a new shaking; otherwise, y is incremented and a new shaking is applied using $y + 1$ neighborhood structure. The overall procedure iterates until some termination conditions are met.

Two recent studies, which modify and exploit successfully the basic VNS scheme to tackle the VRPTW, are those of Bräysy (2001, 2003). Bräysy (2001) proposed a Variable Neighborhood Descent (VND) methodology which introduces features of TS within VND. In particular, a short term memory to penalize unsuccessful pairs of edges during the search is used, such that in following repetitions these pairs cannot be considered. As mentioned by Bräysy (2001), such an approach reduces significantly the size of the neighborhood. On the other hand, Bräysy (2003) proposed a reactive VNS using information gathering during the search. In particular, modification of parameter values and changes in the objective functions were performed, if no further improvement could be obtained, in order to escape from local minima solutions.

The above suggestions reinforced our intuition to use explicitly a TS to perform the local search within the basic VNS and develop the appropriate mechanisms to guide the search process. In the literature, two ways of making hybrids of VNS and TS appear; the use of TS within VNS and the use of VNS within TS. In the first case, the local descent of VNS is replaced by TS, while in the second, different neighborhood structures are exploited by TS. The implementation proposed herein, uses TS internally within VNS to perform the local search for given neighborhood structures, while externally VNS controls the neighborhood changes and the shaking mechanism. Using this rationale, it is reasonable to expect a thorough and systematic examination of the solution space by utilizing powerful trajectory local search and exploitation of different neighborhood topologies. A recent compilation of such hybrids can be found in Hansen and Mladenović (2001).

Another major difference considered in our implementation, compared to the basic VNS scheme, is the introduction of a specialized solution reformation mechanism. Similar to the VNS shaking mechanism, the main idea behind solution reformation is to perform a partial solution reconstruction in a reactive fashion during the search. In particular, if during oscillations of shaking and local search phases of VNS no further improvement could be obtained, solution reformation is applied to the current local optimum solution. The main effort is to modify the fleet mix structure, while maintaining to the extent possible the sequence of customers served by vehicles. Such an approach ensures that desirable characteristics of a solution are maintained while the reformed solution may belong to the basins of attraction of different local optima. Therefore, the information gathered during the search (sequence of customers served by particular vehicle types) is exploited, while the search is both diversified towards promising regions and better sampled close to local optimum solutions.

The proposed solution methodology can be briefly described in steps as follows. At Step 1, several initial solutions are produced using a semi-parallel construction heuristic with different combinations of parameter settings. Next, at Step 2, a route elimination procedure is applied to improve the utilization of the vehicles deployed

and reduce, if necessary, the number of vehicle routes. Subsequently, at Step 3, a subset of solutions is selected based on their quality. Finally, at Step 4, the quality of these solutions is further improved using ReVNTS, while the best solution found is return upon the termination of the search process. More specifically, given an initial solution, ReVNTS iterates between shaking and local search until no further improvement can be obtain and the current local optimum solution s is stored. Next, the solution reformation mechanism is applied, and the search restarts from the reformed solution s' until a new local optimum solution s'' is obtained. If $f(s'') \leq f(s)$ then s is replaced by s'' and a new solution reformation is applied, else ReVNTS starts from a new initial solution. The proposed solution methodology is terminated when either all selected initial solutions have been examined or an upper bound limit γ, with respect to the computational time consumption, has been reached.

3.1 Construction heuristic

According to our survey, several construction heuristics have been proposed for the HFUVRPTW, which mainly modify Golden et al. (1984) and Solomon (1987) savings criteria. However, in order to comply with the requirements and the specifications of both HFUVRPTW and HFFVRPTW, we propose a new general and robust semi-parallel construction heuristic method. The term semi-parallel refers to the simultaneous consideration of all available vehicle types at each iteration of construction, while the building of routes is performed in a sequential fashion using several sub-metrics for next-customer selection and in-route insertion.

A point of primary importance, for the HFVRPTW, is the effective utilization of the available vehicle types since the determination of the appropriate fleet mix is critical and very often hinders high quality solutions. Therefore, pure sequential approaches, which treat one vehicle at each iteration of construction, are not always appropriate, since the choice of the best vehicle type cannot be explicitly considered. On the other hand, although parallel construction heuristics, which iteratively merge routes look attractive, they some times fail to utilize effectively the customers' time windows (recall that time window constraints dictate the sequence in which customers are visited by the vehicles).

Based on these observations, the proposed construction heuristic builds solutions in a semi-parallel fashion. More specifically, at each iteration of the construction mechanism all available vehicle types are considered, and all unassigned customers are candidates for insertion to all vehicles types, not only once but multiple times. Initially, for each vehicle, a customer candidate for insertion is selected from the set of unassigned customers using a greedy function. Next, the selected customer is temporarily assigned to its current best position between two adjacent customers to the associated partial constructed route. This procedure is repeated for each vehicle type explicitly, considering all unassigned customers including those temporarily assigned to other vehicles, until all vehicles filled with customers. At the end of this procedure, a vehicle route is selected and added to the partially constructed solution. The vehicle selection is based on a specialized criterion which measures the average cost per unit transferred. Finally, the customers serviced by the selected vehicle are removed from the set of unassigned customers and the overall procedure is repeated with the selection of a new vehicle, until all customers are serviced. Note that during the building

of vehicle routes, duplication of customers is allowed; however, the fact that only one candidate vehicle is included to the partially constructed solution at each iteration, ensures that every customer is serviced exactly once.

The metric ACUT_k (Average Cost per Unit Transferred) defines the extent to which a particular vehicle type k is efficiently utilized. The main idea is to measure the amount that has to be payed, in order to carry one unit of customer's demand for a particular subset of customers using a vehicle of type k. The latter is expressed as the ratio of the total travelling time plus the fixed cost over the total demand carried by a particular vehicle type. Obviously, the smaller the value of the average cost per unit transferred the more efficient the schedule and the vehicle capacity utilization achieved. As mentioned above, based on the average cost per unit transferred the vehicle that will be included to the partially constructed solution is determined. Therefore, at each iteration, the vehicle type k that minimizes ACUT_k is selected. The metric ACUT_k can be expressed as follows.

$$\mathrm{ACUT}_k = \frac{\sum_{(i,j)\in A} c_{ij}^k x_{ij}^k + w_k}{\sum_{i=1}^n q_i \sum_{j=1}^n x_{ij}^k}. \tag{2}$$

Having defined a criterion to select a candidate vehicle, the focus is given on building the sequence of customers serviced by each vehicle. Following the generic insertion scheme of Solomon (1987), initially a "seed" customer must be determined. In literature, various rationales have been proposed for "seed" customer selection. However, based on experiments, the scheme that best fits the proposed construction heuristic is the selection of a customer i with minimum slack_i ($l_i - \max\{h_{0i}, h_{i0}\}$). In particular, small values of slack_i mean that there is little flexibility in assigning i since the latest time of its time window is close to the time required to travel from depot to i. Thus, it is justified that the more distant customers from the depot with urgent time windows are favored in the early phases of the construction.

On the other hand, various metrics have been proposed in literature, based on time and distance increases of scheduling measures, to evaluate the cost of inserting an unassigned customer into partially built routes. Bräysy and Gendreau (2005) provide a recent compilation of route construction heuristics for the VRPTW. The greedy function of our construction heuristic uses a blend of such metrics modified accordingly to comply with the constraints of HFVRPTW. The basic idea is to take into account short inter-customer distances and effectively utilize the vehicles' available capacity and the start times of service at customer locations. Therefore, our main effort is to determine the subsets and the sequences of customers served by vehicles in addition to the fleet composition, such that the total distribution cost is minimized.

Let i and j be two consecutive customer in a partial constructed route ρ and index u denote an unassigned customer. The greedy function that measures the cost of inserting u between i and j served by vehicle type k is denoted as Φ_{ij}^{uk}. This composite greedy function combines the effects of several sub-metrics through a weighted linear relationship defined as follows:

$$\Phi_{ij}^{uk} = \alpha_1\left(C_{ij,u}^0 + C_{ij,u}^1\right) + \alpha_2 C_{ij,u}^2 + \alpha_3 C_{ij,u}^3 + \alpha_4\left(C_{ij,u,k}^4 + C_{ij,u,k}^5\right), \tag{3}$$

where α_1, α_2, α_3 and α_4 are nonnegative weights such that $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$. The first cost component of Eq. 3 represents the extent to which the vehicle's k arrival

time a_u and the earliest time e_u of customer u are close to one-another. This depends on the insertion position of customer u and provides a measure of the coverage of the selected customer's time window, which results from the insertion of u (Ioannou et al. 2001).

$$C^0_{ij,u} = a_u - e_u. \tag{4}$$

Similarly, $C^1_{ij,u}$ indicates the compatibility of the time window of selected customer u with the specific insertion position into the current route (Ioannou et al. 2001). More specifically, the insertion of u defines the *push forward* (time gap) between the latest service time l_u of customer u and the arrival time a_u.

$$C^1_{ij,u} = l_u - (a_i + s_i + t_{iu}). \tag{5}$$

The second cost component of Eq. 3, $C^2_{ij,u}$, measures the travelling time increase caused by the insertion of u (Solomon 1987):

$$C^2_{ij,u} = t_{iu} + t_{uj} - t_{ij}. \tag{6}$$

Metric $C^3_{ij,u}$ considers the time delay of the vehicle's arrival time at customer j, incurred by the insertion of u; it is expressed as the difference between the vehicle arrival time at customer j, before and after the insertion of u and represents the schedule that has to be pushed forward to accommodate u (Solomon 1987).

$$C^3_{ij,u} = (a_u + s_u + t_{uj}) - (a_i + s_i + t_{ij}). \tag{7}$$

The fourth cost component of Eq. 3 combines two metrics $C^4_{ij,u}$ and $C^5_{ij,u}$. The first one, $C^4_{ij,u}$, maps large loads into small costs such that customers with large demands are assigned to vehicles first. This metric maximizes the utilization of the vehicle's capacity, producing cost efficient routes:

$$C^4_{ij,u,k} = Q_k - \left(\sum_{i \in N \cap u} q_i \sum_{j \in N} x^k_{ij} \right) - q_u. \tag{8}$$

Finally, $C^5_{ij,u,k}$ measures the difference obtained with respect to the average cost per unit transferred by vehicle k prior and after the insertion of customer u.

As already mentioned, the proposed heuristic can address both HFUVRPTW and HFFVRPTW. However, in case of the HFFVRPTW, an additional modification must be considered since the number of available vehicles of each type is limited. In general terms, such a restriction combined with time windows makes difficult to construct feasible solutions for all problem instances. The latter intensifies in situations where the fixed fleet constraint (limited vehicle availability) is very tight. For this reason, during construction a relaxed constraint approach, in terms of fixed fleet, is followed.

Specifically, the proposed construction heuristic considers a combination of limited and unlimited number of vehicles for each type. At the early phases of construction, the fixed fleet of vehicles is used to service as many customers as possible, and if there are any unassigned customers left, additional surplus vehicles are generated

with inherent characteristics to serve them. These additional vehicles are generated according to ACUT_k for each vehicle type. For example, if there exist unassigned customers and the available fleet composition is not sufficient, the method generates one vehicle from each type and assigns to each one of them the maximum possible number of customers. Finally, the vehicle type k with $\min_{k \in K}(\text{ACUT}_k)$ is added to the partial constructed solution. Using this rationale, it is ensured that the additional vehicle mobilized has inherited the best qualifications needed to service these customers. The procedure terminates when all unassigned customers are served. The complete construction heuristic is provided below in the form of pseudocode:

Semi-Parallel Construction Heuristic
Initialize available vehicle type lists $V_k, k = 1, 2, \ldots, k_{\max}$
$S \leftarrow$ InitialSolution( ), $C_s \leftarrow$ CustomerList( ) "Unassigned customers"
Do
  $k \leftarrow 1$
  While $k \leq k_{\max}$ **do**
    seed $\leftarrow$ FindSeedCustomer(C_s)
    If ($V_k \neq 0$) **then** $r_k \leftarrow$ InitializeRoute(k), $r_k \leftarrow$ InsertSeedCustomer(seed)
      Do
        done $\leftarrow$ True
        For all customers u of $C_s - \{\text{seed}\}$ **do**
          For all insertion positions i, j of r_k **do**
          $\Phi_{ij}^{uk} \leftarrow$ GreedyFunction(i, j, u, k), $\phi \leftarrow$ StoreBest(Φ_{ij}^{uk}),
          done $\leftarrow$ False
          Endfor
        Endfor
        $r_k \leftarrow$ InsertAt(ij, ϕ)
      While done = False
      $k \leftarrow k + 1$
    Else
      $k \leftarrow k + 1$
    Endif
  Endwhile
  For all vehicles routes r_k **do**
    $\text{ACUT}_k \leftarrow$ AverCoctPerUnitTransfer(r_k), $x \leftarrow$ StoreBest(ACUT_k)
  Endfor
  $S \leftarrow$ InsertRoute(r_x), $V_x \leftarrow$ Remove(x), $C_s \leftarrow$ RemoveCustomers(r_x)
  If all lists $V_k = 0$ **then** Re-Initialize vehicle type lists $V_k, k = 1, 2, \ldots, k_{\max}$
While $|C_s| > 0$

3.2 Route elimination

The solutions generated by the semi-parallel construction heuristic are subject to route elimination, during which the primary objective is to reduce the number of routes. Obviously, in case of HFFVRPTW the elimination of infeasible vehicles deployed during construction is essential, in order to proceed to the improvement phase.

On the other hand, for the HFUVRPTW there is no evident need that a route elimination procedure must be applied, assuming that the construction heuristic produces good quality solutions with the appropriate blending of vehicle types. However, due to time windows and depending on the structure of the problem, it is possible at late stages of construction the remaining set of unassigned customers to require the deployment of "excessive" not effectively utilized vehicles. For this reason, when the HFUVRPTW is considered, the main effort is to improve the vehicles' utilization.

As proposed earlier, the metric ACUT_k measures the average cost per unit transfer and provides an indication with respect to the vehicles' utilization. Generally, there are two alternatives to improve this: the first is to remove some customers and deploy a vehicle with smaller capacity and therefore lower fixed cost; the second is to eliminate completely the particular route. In case of HFUVRPTW both these alternatives are considered. From the implementation viewpoint, initially the vehicle routes with percentage deviation greater than λ, with respect to the overall average $\overline{\text{ACUT}_k}$ of the solution, are identified. Subsequently, the route elimination procedure is applied to all these vehicle routes sequentially, starting from those with the smallest capacity. The main effort is either to completely eliminate these vehicle routes or to improve their utilization by deploying vehicles with smaller fixed cost. In particular, all customers served by these vehicles are candidates to be serviced by other vehicle routes, until a vehicle with smaller fixed cost can be mobilized to service the remaining subset of customers.

The proposed route elimination procedure is based on the generic Ejection Chain (EC) neighborhood structure proposed by Glover (1992) and used by Bräysy (2003) as a stand alone route elimination method enhanced with an intelligent reordering mechanism called IR-insert. The basic idea, as mentioned by Bräysy (2003), when using EC for route elimination is the following. Consider a solution s and remove a customer i from a route r_k, where $r_k \in s$. If non feasible insertion-relocation positions can be found for customer i considering all $r \in s, r \neq r_k$, the ejection chain is applied. An ejection chain implies the following: find a route r_m in which if a customer j is removed from r_m, customer i can be feasibly inserted into r_m. Subsequently, this procedure is repeated considering the removed from r_m customer j. The removals and insertions are repeated until the customer examined for insertion can be inserted into a route without ejecting another customer. Note that the latter procedure treats one customer at each iteration.

As mentioned above, within the EC framework only simple customer insertion is considered. However, in cases were time window constraints are present, the possibilities that the simple insertion scheme fails are considerable. For this reason, Bräysy (2003) introduced a special intelligent reordering procedure (IR-insert), in case the simple insertion fails. The basic idea behind IR-insert is to find an insertion position for a customer, even if this insertion causes time window infeasibilities, and subsequently try to restore back feasibility using particular customer reordering mechanisms.

The IR-insert scheme originally proposed, initially determined the best insertion position for a customer i into a route r_e, in terms of distance and waiting time increments, assuming that the arrival time a_{ik} of vehicle k performing route r_e at customer i is on time. Subsequently, customer i is inserted into r_e and the first customer u with

violated time window, is identified. Finally, the main focus is to restore the feasibility of the route by changing the sequence of customers served by the vehicle. In particular, two alternative customer reordering mechanisms were proposed by Bräysy (2003). The first one performs a forward reordering, in which precedent customers of u are considered for insertion between any pairs of consecutive customers of the same route. The other alternative, is the reordering of customers serviced before u so that the duration of route, or similarly the start time of service at customer u, is minimized. Using both these mechanisms, one may expect to arrive earlier at customer u with the violated time window and thus restore feasibility. Obviously, the above procedure is repeated using both mechanisms until either the route is feasible or no feasible reordering-insertion positions can be found.

The route elimination procedure proposed in this paper, although it adapts the EC framework and utilizes the basic ideas of IR-insert, is implemented according to a completely different perspective. In terms of EC two major differences are introduced. First, a best-accept strategy is applied. The latter choice is made due to the fact that the solution quality, after a series of successful ejection chains, may deteriorate significantly. However, although the best-accept strategy seems better than the first-accept, there is a tradeoff with respect to the additional memory required, in order to store at each iteration all successive chains. Second, at each iteration of EC, the customers already included in a chain are penalized, in order to avoid cycling. To the other end, in terms of IR-insert, instead of trying myopically to restore feasibility on a particular customer, the overall infeasibility of a route is considered. Therefore, during the customer reordering the goal is to reduce the total infeasibility of the route, without necessarily preserving the feasibility of some particular parts of the route. Another modification, is the introduction of an additional mechanism for restoring feasibility, performing a backward reordering. Finally, different criteria for determining the initial insertion positions and some enhancements within forward reordering mechanisms are considered.

The proposed and modified IR-insert can be illustrated in steps as follows. Let r_e be a route considered for elimination and a customer u of r_e be a candidate for insertion into another route r_m. Initially, the insertion position of u into r_m, performed by vehicle k, that causes the least possible infeasibility, or similarly the position with minimum $\text{Inf}_{r_m} = \sum_{i \in r_m} \max\{a_{ik} - l_i, 0\}$, is determined. Next, customer u is inserted into r_m and customer reordering mechanisms are applied as shown below.

Step 1 *Initialization*: Calculate Inf_{r_m} and identify the first "infeasible" customer i with violated time window. If no infeasible customers are found, terminate.

Step 2 *Simple Reordering*: Examine all possible swaps between preceding customers of i and select the pair that reduces the most the lateness of service, or similarly the start time of service, at customer i.

Step 2a Recalculate the total infeasibility of the route Inf_{r_m}. If customer i is infeasible repeat Step 2 until no further improvement can be obtained and proceed to Step 3; otherwise goto Step 1.

Step 3 *Backward Reordering*: Find the insertion position, between preceding pairs of customers, for the infeasible customer i such that the overall infeasibility Inf_{r_m} is reduced. If customer i is removed from his original position goto Step 1; otherwise proceed to Step 4.

Step 4 *Forward Reordering*: For all preceding customers before i, find which customer u if removed from r_m will cause the larger reduction with respect to Inf_{r_m}.

Step 4a Examine all insertion positions between any pairs of consecutive customers of i. Select the one that will improve the most Inf_{r_m} and goto Step 1. If no improvement can be obtained search for relocation places (simple insertion) into other vehicle routes.

Step 5 If the total infeasibility Inf_{r_m} of route r_m is improved goto to Step 1; otherwise terminate.

3.3 Reactive variable neighborhood search

The proposed Reactive Variable Neighborhood Tabu Search (ReVNTS), mainly consists of the following five phases: initialization, shaking, local search, move and reformation. Given an initial solution s, at the initialization phase, a set of neighborhood structures with increasing cardinality ($|N_1| < |N_2| < \cdots < |N_{y_{\max}}|$) is defined and neighborhood index y is initialized to 1. In the shaking phase, a solution s' in the yth neighborhood of the current solution s is randomly selected. Local search phase begins (Tabu Search in particular) and a new solution s'' is generated. If $f(s'') < f(s)$, s is replaced by s'', neighborhood index is re-initialized and a new shaking phase is applied with $y = 1$. Otherwise, y is incremented and a new shaking phase starts using a different neighborhood structure. The oscillations between shaking and local search are repeated until all possible neighborhood structures are examined, i.e., $y = y_{\max}$, and no further improvement can be obtained. At this point, the best solution found so far (s) is stored ($\hat{s} \leftarrow s$), solution reformation is applied, the neighborhood index is re-initialized and VNTS restarts from the reformed solution s. The above described procedure of oscillations between shaking and local search is repeated until no further improvement can be obtained. Finally, the new local optimum solution s, found after applying solution reformation, is compared to $\hat{s}$ (local optimum solution found prior to solution reformation). If $f(\hat{s}) > f(s)$ $\hat{s}$ is replaced by s and the improvement scheme is repeated from a new solution reformation. Otherwise, the overall scheme terminates, the best overall solution found is updated and ReVNTS starts from a new initial solution.

Recall that during the first phase of the proposed solution methodology, only a subset of f solutions, denoted as F, is selected for further improvement using ReVNTS (second phase). Therefore, ReVNTS is terminated when either all selected initial solutions has been examined or an upper bound limit γ, with respect to the computational time consumption, has been reached. Given the above specifications the pseudocode of ReVNTS can be illustrated as follows:

Reactive VNTS
Define a set of neighborhood structures N_y, $y = 1, 2, \ldots, y_{\max}$
For all solutions s of set F **do**
  $s \leftarrow$ InitialSolution(F), $F \leftarrow$ Remove(s)
  While ($|F| \neq 0$) OR (CPU time consumed $\leq \gamma$) **do**
    $y \leftarrow 1$, done $\leftarrow$ False, Reform $\leftarrow$ False
    While ($y \leq y_{\max}$) AND (done = False) **do**
      $s' \leftarrow$ PickAtRandom($N_y(s)$) "Shaking phase"

$s'' \leftarrow$ TabuSearch(s', y) "Local Search phase"
If ($f(s'') < f(s)$) **then** $s \leftarrow s''$, $y \leftarrow 1$ "Move phase"
Else
If ($y < y_{\max}$) **then** $y \leftarrow y + 1$ "Move phase"
Else
If (Reform = False) **then** "Reformation phase"
Reform $\leftarrow$ True, $y \leftarrow 1$, $\hat{s} \leftarrow s$, $s \leftarrow$ Reformation(s)
Else
If ($f(s) < f(\hat{s})$) **then** "Reformation phase"
$y \leftarrow 1$, $\hat{s} \leftarrow s$, $s \leftarrow$ Reformation(s)
Else
done $\leftarrow$ True
Endif
Endif
Endif
Endif
Endwhile
UpdateBestFoundSolution(s)
Endwhile
Endfor

3.3.1 Shaking

The objective of the shaking phase is to perturb the solution so as to provide a good starting point for the local search. The starting point must belong to basin of attraction of a different local optimum than the current solution, although, not "too far" from s, otherwise the algorithm would degenerate into a simple random multi-start (Blum and Roli 2003). In this way, choosing at random a solution s' in the neighborhood of the current local optimum solution s, favorable characteristics and good features will often be maintained and used to obtain promising neighboring solutions (Hansen and Mladenović 2001). The latter also best fits TS, since an effective sampling close to the neighborhood of a local optimum solution allows a more efficient and effective intensification local search.

3.3.2 Tabu search

The local search phase of the proposed ReVNTS is performed by a TS fine tuned for intensification local search. TS explores the solution space by moving, at each iteration, from a solution s to the best (or first improving) solution in the neighborhood $N_y(s)$. Contrary to other descent methods, the current solution may deteriorate from one iteration to another. To avoid cycling, solutions possessing some attributes of recently explored ones are temporarily declared as *tabu*. The latter is achieved using a so-called short term memory. Tabu moves are represented by attributes which are stored in an ordered queue called *tabu list*. The best admissible move is chosen as the highest evaluation move in the neighborhood of the current solution in terms of objective function and tabu restrictions.

The tabu list is imposed to restrict the search from revisiting solutions that were considered previously and to discourage the search process from cycling between subsets of solutions. The neighborhood of the current solution is, therefore, restricted to the solutions that do not belong to the tabu list. At each iteration the best solution of $N_y(s)$, that does not belong to the tabu list, is chosen as the new current solution. This solution is then added to the tabu list and one of the solutions already added in the tabu list is removed. The duration that an attribute remains tabu is called *tabu tenure* and it can vary over different intervals of time. The tabu status can be overridden only if certain conditions are met; the latter is called aspiration criterion and occur, i.e., when a tabu solution is better than the best solution encountered previously during the search. The overall procedure iterates until a termination criterion is met. The pseudocode of the proposed TS is given below.

Tabu Search
Given a solution s and a neighborhood structure y
Initialize tabu list TL_y of $t_{\min}$ size
elite $\leftarrow s$, AspirationCondition(s), counter $\leftarrow 0$, $t_{\text{size}} \leftarrow t_{\min}$
While counter $\leq$ maxIters **do**
  Find $s' \in N_y(s) \mid s$ subject to tabu & aspiration conditions
  AllowedSet(s) $\leftarrow s'$
  $s \leftarrow$ ChooseFirstImproving(AllowedSet(s))
  UpdateTabulist()
  If ($f(s) < f$(elite)) **then**
    elite $\leftarrow s$, AspirationCondition(elite), counter $\leftarrow 0$, $t_{\text{size}} \leftarrow t_{\min}$
  Else
    counter $\leftarrow$ counter $+ 1$
    If ($t_{\text{size}} < t_{\max}$) **then** $t_{\text{size}} \leftarrow t_{\text{size}} + 1$
  Endif
  $s \leftarrow$ VehicleReassignment(s)
Endwhile
$s \leftarrow$ elite

The termination condition used, maxIters, bounds the maximum number of iterations allowed for TS without observing improvement. This information is also used during the search for similar adjustments on the size of the tabu list. The use of a small tabu tenure results in a more effective intensification search since it allows cycling of small periods. On the other hand, a large tabu tenure will allows a diversified search of more distant neighbors, which may result in escaping from a current local optimum solution. For this reason, initially the tabu tenure t_{size} is set equal to a lower value $t_{\min}$, while at each iteration that no improvement is observed it is incremented up to an upper bound $t_{\max}$. In Contrast, when improvement is achieved, tabu tenure is re-initialized to $t_{\min}$.

As mentioned earlier, for a given neighborhood structure the allowed set of neighbors is built and the first improving admissible neighbor of the selected neighborhood of a current solution is selected. Finally, at the current solution a vehicle reassignment procedure is applied. The main objective of vehicle reassignment is to increase the possibilities of improving vehicle's capacity utilization during the search, through

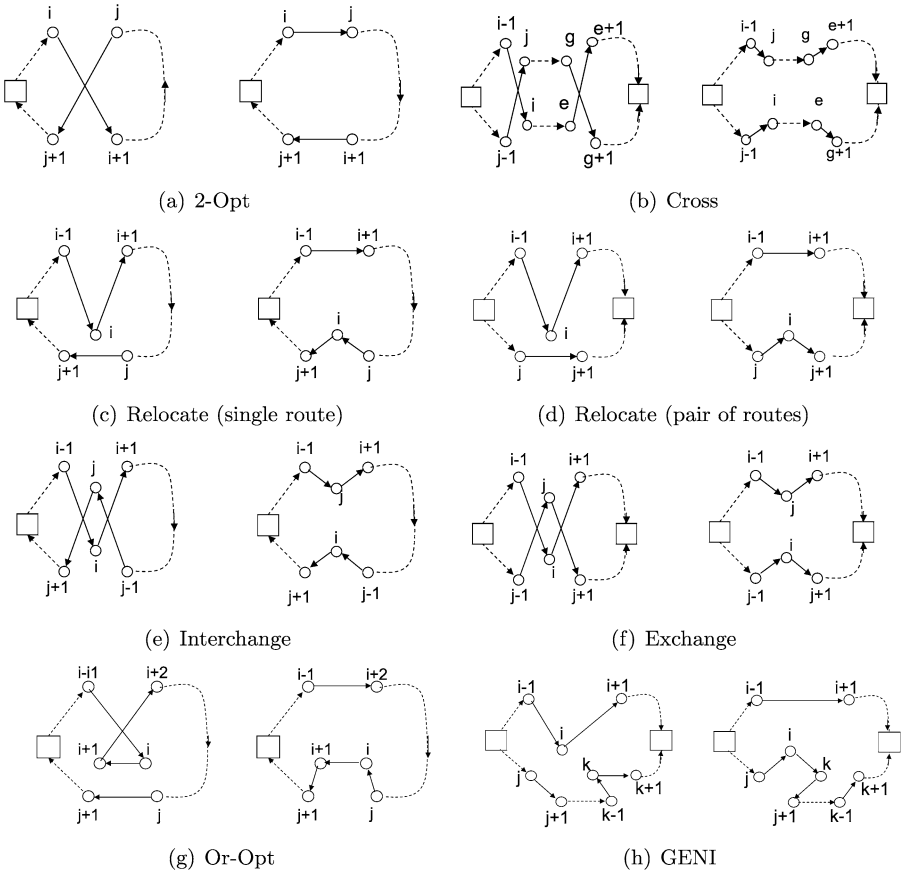
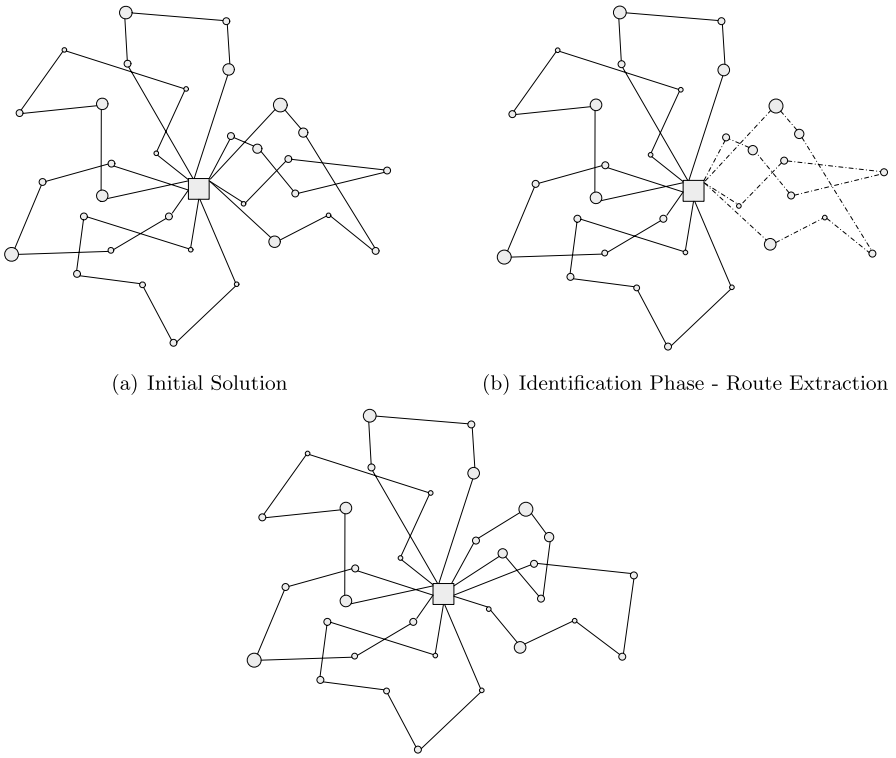


Fig. 1 The set of neighborhood operators

reciprocal exchanges of the vehicle types deployed among routes. Furthermore, during the search it is possible to reassign vehicles with smaller fixed cost, if there are any available, and therefore, reduce the overall cost. Obviously, if this approach fails, one can apply the proposed route elimination procedure. However, based on computational experiments the benefits gained are modest and the computational complexity increases.

3.3.3 Neighborhood structures

The process of changing neighborhoods with increasing cardinality, in case of no improvements, corresponds to a diversification of the search. The effectiveness of this dynamic neighborhood can be explained considering that a solution, locally optimal with respect to a neighborhood, is probably not locally optimal with respect to another neighborhood. This is valid due to the fact that the neighborhood structure determines the topological properties of the search landscape.



(a) Initial Solution

(b) Identification Phase - Route Extraction

(c) Reconstruction phase - Reformatted Solution

Fig. 2 Solution reformation mechanism

In general terms, the definition of rich neighborhoods increases the chances of finding high quality solutions. In particular, the neighborhood structures utilized are defined as a blend of well known local search move operators that transform one solution to another. From the implementation viewpoint, neighborhood change and selection is applied on a sequential fashion based on their cardinality in the following order: GENI, Or-Opt, 2-Opt, Cross, Relocate and Exchange (Kindervater and Savelsbergh 1998). In Fig. 2 all neighborhood structures considered are illustrated. Note that, 2-Opt and Or-Opt are applied only on single routes (intra-route), while Cross and GENI are applied only on pair of routes (inter-route). Finally, in case of Relocate and Exchange both alternatives are considered in a nested fashion, evaluating first intra-route and then inter-route moves.

3.3.4 Solution reformation

The purpose of the solution reformation phase is quite similar to that of shaking. In particular, the objective is to ruin-and-recreate a current local optimum solution in order to diversify the search and escape, in the long term, from the local optimum using the accumulated experience acquired during the search. Assuming that good

quality solutions share some common characteristics, the basic idea of the solution reformation is to maintain some favorable features and reconstruct part of the solution considered. In case the current solution is local optimum with respect to VNTS, it is reasonable to expect that restarting the improvement from the reformed solution, may reach a higher quality region of the solution space. More specifically, the proposed solution reformation consists of two mechanisms. The first identifies which parts (routes in particular) should be maintained and determines the "problematic" features of the solution. The second mechanism reconstructs from scratch the routes that are candidates for reformation.

Let s denote the current solution. The first task is to identify the route r_p with maximum average cost per unit transfer using cost metric Eq. 2. Then, the neighboring routes r_n are defined, $r_n \in N_r(r_p)$. A route r_n is regarded as neighboring to r_p if the customers serviced by r_n, are geographically close to the customers that belong to r_p. In order to measure the extent to which the customers of two particular routes are geographically close, the maximum distance $\max_{(i \in r_n, j \in r_p)} h_{ij}$ among all possible pairs of customers (i, j) must not exceed a predefined limit for closeness z. Finally, all customers of routes r_n, such that $r_n \in N(r_p)$, along with customers of r_p, are extracted from the solution s.

The second task is to reconstruct the ruined part of solution s. For this purpose, the semi-parallel construction heuristic is employed, using all available vehicle types. More specifically, construction parameter α_2 is set equal to 0.4 while the remaining parameters are equal weighted. Although, the solution reformation will most probably deteriorate the quality of the solution in terms of distance traveled, the use of the proposed semi-parallel construction heuristic ensures that the best possible fleet mix is used and the sequence in which customers are serviced by vehicles is attractive in terms of time window utilization.

4 Computational results

4.1 Data sets & parameter settings

For the evaluation of the proposed solution method, various computational experiments were conducted. In particular, computation results reported in this paper were obtained using the benchmark data sets proposed by Liu and Shen (1999b). These benchmarks instances are derived from Solomon (1987) 100 customer instances R, C and RC, originally proposed for the VRPTW. The Cartesian coordinates of customers in problem sets R1 and R2 are randomly generated using a uniform distribution. In contrast, sets C1 and C2 have clustered customers. Finally, sets RC1 and RC2 contain semi-clustered customers, i.e., a combination of clustered and randomly (uniformly) distributed customers. Moreover, R1, C1, RC1 have short scheduling horizon, contrary to R2, C2, RC2 data sets. To define vehicle capacities and fixed costs Liu and Shen (1999b) propose three different subclasses of instances a, b and c, each characterized by different fixed vehicle costs. Table 1 illustrates the fixed costs of the vehicles for each subclass, while A, B, C, D, E, F refer to the vehicle types.

Generally, the parameter settings highly depend on the characteristics of each problem. Therefore, the use of different parameter settings may significantly improve

Table 1 Liu and Shen (1999a, 1999b) heterogeneous vehicle fleet characteristics for the HFUVRPTW (subclasses *a*, *b* and *c*)

Data set	Vehicle type	Fixed costs			Total capacity
		a	*b*	*c*	
R1	*A*	50	10	5	30
	B	80	16	8	50
	C	140	28	14	80
	D	250	50	25	120
	E	500	100	50	200
C1	*A*	300	60	30	100
	B	800	160	80	200
	C	1350	270	135	300
C2	*A*	1000	200	100	400
	B	1400	280	140	500
	C	2000	400	200	600
	D	2700	540	270	700
R2	*A*	450	90	45	300
	B	700	140	70	400
	C	1200	240	120	600
	D	2500	500	250	1000
RC1	*A*	60	12	6	40
	B	150	30	15	80
	C	300	60	30	150
	D	450	90	45	200
RC2	*A*	150	30	15	100
	B	350	70	35	200
	C	550	110	55	300
	D	800	160	80	400
	E	1100	220	110	500
	F	2500	500	250	1000

the quality of the solutions produced. However, due to the large computational effort needed to optimize the value of each parameter separately, fixed values for all parameters were used. Based on computational experiments, the construction parameters are the most sensitive, affecting significantly the results. Furthermore, no evident correlation was found among these interdependent parameters. For this reason, during the first phase of the proposed solution methodology several parameter settings were tried, within some presignified ranges.

Particularly, construction parameters α_1, α_2, α_3 and α_4 used to control two different trends according to which the initial solutions are constructed. The first three parameters control the sequence in which customers are serviced, and therefore, the time window utilization of the route, while α_4 controls vehicle utilization. However, based on the characteristics of the problems considered, there is a tradeoff between time and capacity utilization. For R1, C1, RC1 data sets, capacity utilization dominates the solution quality, while in R2, C2, RC2 data sets the sequence of customers is of primary importance. The latter is justified by the fact that the average number of customers served by a vehicle is larger. Therefore, when considering R2, C2 and RC2, the main focus is on improving the intra-route sequence of customers, while for R1, C1 and RC1 improvement efforts intensified on the vehicle capacity utilization. For these reasons, the values of α_1 and α_3 ranged between 0.1–0.3, α_2 was always <0.3, while α_4 was within 0.2–0.4 and <0.15, when solving the first and the second group of data sets.

The above defined ranges were used during the initialization phase, via increments of 0.01. Generally, the production of an 1000-solutions population using the semi-parallel construction heuristic takes less than 10 sec. On the other hand, the number of solutions selected for further improvement, or similarly the cardinality $|F|$, is set to be equal to 20. However, as already mentioned, the primary termination condition of ReVNTS is the total CPU time γ. Obviously, the larger the cardinality of F and the CPU time, the larger the possibilities of finding higher quality solutions. However, after a number of ReVNTS iterations, further gains in the solution quality are made at the expense of CPU time. Therefore, based on experiments the value for γ that best fits ReVNTS was set equal to 1200 sec. Another parameter that affects the computational complexity of ReVNTS is the maximum number of iterations MaxIters allowed for the TS when no improvement is observed. Based on experiments, a relatively small value for the MaxIters is selected (MaxIters = 30). Moreover, the minimum tabu list size, $t_{\min}$, was set equal to 10, while the maximum, $t_{\max}$, equal to 30. These values were determined according to the intension of the search to extensively examine a particular neighborhood, or to escape from a local optimum, respectively. The minimum percentage deviation λ used to identify the routes subject for elimination is set equal to 40% and the geographical limit for closeness z used by the solution reformation mechanism is set equal to 40. Finally, the algorithm was developed in Visual C++ and run on a 1.5 GHz Intel PIV personal computer.

4.2 Comparative analysis

Initial computational experiments were conducted for the HFUVRPTW, where the solution cost is derived by the total fixed cost of the vehicle types used (see Table 1) and the total scheduling time, without considering the service time of customers as additional cost. In particular, the proposed solution methodology is compared to Liu and Shen (1999b) and Dell-Amico et al. (2006) methods. As already mentioned in the introductory section, Liu and Shen (1999b) propose construction heuristics invoking also an improvement procedure, referred as **LS**. On the other hand, Dell-Amico et al. (2006) propose a parallel regret construction heuristic, referred as **MPR-TW**, and a multi-start ruin-and-recreate metaheuristic method, referred as **T-RR-TW**. Although,

Table 2 Comparison among construction heuristics proposed for the HFUVRPTW

Data set	LS (No impr.)		MPR-TW		SPH
	TC	% Dev	TC	% Dev	TC
R1a(12)	4562.00	−6.09	4370.27	−1.64	4299.93
R1b(12)	2149.00	−4.13	2028.40	1.72	2063.81
R1c(12)	1788.00	−2.63	1702.22	2.30	1742.26
R1(36)	2833.00	−4.85	2700.30	0.06	2702.00
C1a(9)	8042.00	−1.86	8285.53	−4.94	7895.26
C1b(9)	2626.00	−6.46	2474.91	−0.33	2466.66
C1c(9)	1870.00	−7.74	1668.66	3.86	1735.68
C1(27)	4179.33	−3.64	4143.03	−2.74	4032.53
RC1a(8)	5429.00	−4.76	5245.74	−1.22	5182.43
RC1b(8)	2342.00	2.15	2311.67	3.42	2393.54
RC1c(8)	1929.00	3.53	1914.45	4.26	1999.56
RC1(24)	3233.33	−1.30	3157.29	1.08	3191.84
R2a(11)	3855.00	−8.83	3937.59	−11.17	3542.11
R2b(11)	1915.00	−6.93	1902.16	−6.21	1790.94
R2c(11)	1589.00	−4.51	1609.46	−5.86	1520.41
R2(33)	2453.00	−7.38	2483.07	−8.69	2284.49
C2a(8)	7058.00	−13.14	6854.72	−9.88	6238.56
C2b(8)	2054.00	−5.53	1981.32	−1.80	1946.33
C2c(8)	1373.00	0.41	1283.43	6.90	1378.61
C2(24)	3495.00	−9.64	3373.16	−5.81	3187.83
RC2a(8)	5381.00	−11.23	5276.40	−9.07	4837.76
RC2b(8)	2432.00	−14.33	2351.02	−10.52	2127.21
RC2c(8)	2066.00	−8.15	1979.61	−3.62	1910.38
RC2(24)	3293.00	−11.31	3202.34	−8.24	2958.45
Total Average	3247.78	−6.15	3176.53	−3.82	3059.52

Dullaert et al. (2002) propose construction heuristics for the HFUVRPTW, their results do not appear in the comparison Tables since they did not considered fixed costs.

Table 2 compares the proposed semi-parallel construction heuristic performance to the LS modified heuristics (without improvement) and to the MPR-TW heuristic. In particular, columns of Table 2, represent the total distribution costs (TC), and the av-

erage deviation (%Dev), for each respective method. The average deviation illustrates the percentage difference between distribution costs obtained by our approach (SPH) and distribution costs of LS and MPR-TW. Finally, the last row indicate the average results over all instances, while for each subclass a respective average is reported. On aggregate, the performance of SHP based on the average total distribution cost for all 168 problem instances, dominates LS and MPR-TW with 6.15% and 3.82% reductions on the total average, respectively. In most cases SPH clearly outperforms LS and MPR-TW, producing higher quality solutions with cost reductions up to 14.33% and 11.17% respectively, while the worst case performance of SPH was 3.53% and 6.90% above LS and MPR-TW.

Similar are the figures for the ReVNTS. Table 3 compare the proposed hybrid metaheuristic ReVNTS to the LS modified heuristics with improvement and to the T-RR-TW metaheuristic. On aggregate the total average distribution cost reductions obtained, were 9.99% and 3.49% compared to LS and T-RR-TW, while the maximum and minimum deviations are up to −26.21%, −16.70%, −4.31% and −0.53%, respectively. In terms of total CPU time consumption, the upper limit of ReVNTS running time was set to 1200 sec, while during the first phase SPH consumed on average 30–40 sec and at most 420 sec (worst case) for route elimination. It is also worth mentioning, that cost reductions obtained for all subclasses R2, C2 and RC2 were close to 10%.

In addition, computational experiments were also conducted for the HFFVRPTW. Since we are not aware of benchmarks instances for the HFFVRPTW, we assumed Liu and Shen (1999a) best fleet size and mix, obtained for the corresponding HFUVRPTW, to be the given fixed fleet. Such approaches also followed by Taillard (1999), Tarantilis et al. (2003, 2004) and Li et al. (2006) for the HFFVRP. Thus, using this rationale, the data sets of Liu and Shen (1999b), proposed for the HFUVRPTW, were used as the test bed for our experiments without considering any further modifications. In order to provide the proof of concept of the proposed methodology for the HFFVRPTW, comparisons with Liu and Shen (1999a) modified heuristics LS, are reported; we used a single basis for comparison since Liu and Shen (1999a) are the only authors report the fleet size and mix obtained along with the total distribution cost.

Table 4 illustrates the performance of Liu and Shen (1999a) and the proposed solution methodology for the HFFVRPTW. The results show substantially better total distribution costs indicating the effectiveness of the proposed method in tackling the HFVRPTW either viewed as HFUVRPTW or HFFVRPTW. More specifically, the proposed semi-parallel construction heuristic produced feasible solutions for all instances, while for most of them yielded higher quality solutions. The fact that our heuristic outperforms in most cases LS, although, the latter tackles the HFUVRPTW with the objective to find the best fleet mix, reinforces our intuition to exploit to a larger extent time window information within a semi-parallel construction scheme. The reductions obtained by ReVNTS were up to 12.69%, compared to the LS.

Finally, Tables 5, 6, 7 report the best solutions found by the proposed solution methodology for all data sets considering HFUVRPTW. It is worth mentioning the differences observed among the vehicle types deployed for each subclass. In particular, in subclass a, where the fixed costs are high, small vehicles are the most favorable

Table 3 Comparative analysis among heuristics with improvement LS, metaheuristic T-RR-TW, and the ReVNTS for the HFUVRPTW

Data set	LS (impr.)		T-RR-TW		ReVNTS
	TC	% Dev	TC	% Dev	TC
R1a(12)	4398.00	−6.53	4180.83	−1.27	4128.48
R1b(12)	2054.00	−7.98	1927.57	−1.33	1902.19
R1c(12)	1700.00	−7.45	1615.44	−2.10	1582.18
R1(36)	2717.33	−7.08	2574.61	−1.46	2537.62
C1a(9)	8007.00	−12.09	7229.02	−1.20	7143.16
C1b(9)	2485.00	−5.22	2384.77	−0.97	2361.78
C1c(9)	1705.00	−5.18	1629.70	−0.53	1621.09
C1(27)	4065.67	−9.63	3747.83	−1.06	3708.68
RC1a(8)	5184.00	−4.48	5117.96	−3.15	4961.69
RC1b(8)	2235.00	−4.31	2163.51	−0.97	2142.65
RC1c(8)	1849.00	−4.47	1784.51	−0.82	1769.93
RC1(24)	3089.33	−4.44	3021.99	−2.16	2958.09
R2a(11)	3809.00	−15.26	3568.97	−8.00	3304.57
R2b(11)	1797.00	−19.88	1727.04	−15.21	1498.97
R2c(11)	1513.00	−18.08	1436.22	−12.09	1281.31
R2(33)	2373.00	−17.00	2244.08	−10.64	2028.28
C2a(8)	6717.00	−16.63	6267.75	−8.83	5759.02
C2b(8)	1970.00	−12.31	1897.62	−8.18	1754.07
C2c(8)	1288.00	−4.46	1276.29	−3.51	1232.98
C2(24)	3325.00	−14.05	3147.22	−7.95	2915.36
RC2a(8)	5273.00	−19.67	4752.95	−7.87	4406.28
RC2b(8)	2324.00	−23.04	2156.11	−14.15	1888.83
RC2c(8)	1978.00	−26.21	1828.95	−16.70	1567.22
RC2(24)	3191.67	−21.78	2912.67	−11.14	2620.77
Total Average	3074.00	−9.99	2892.24	−3.49	2794.80

by the algorithm. On the other hand, in subclasses *b* and *c*, where the fixed costs for all vehicle types are significantly lowered, the algorithm deploys larger vehicles, and therefore, reduces the total number of vehicles used.

Table 4 Comparisons between Liu and Shen (1999a) and the proposed methodology for the HFFVRPTW

Instance	LS			SPH			ReVNTS	
	FM	TC	% Dev	FM	TC	% Dev	FM	TC
r101	$A^1B^{11}C^{11}D^1$	5061.00	−10.41	$A^1B^{10}C^{11}D^1$	4766.30	−3.98	$B^{10}C^{11}D^1$	4583.99
r102	$A^1B^4C^{14}D^2$	5013.00	−13.40	$A^1B^4C^{13}D^2$	4680.54	−5.88	$B^3C^{14}D^2$	4420.68
r103	B^7C^{15}	4772.00	−13.75	B^6C^{15}	4421.08	−5.39	B^6C^{15}	4195.05
r104	B^9C^{14}	4455.00	−9.58	B^8C^{14}	4413.89	−8.57	B^8C^{14}	4065.52
c101	A^1B^{10}	9272.00	−5.02	A^1B^{10}	9312.34	−5.48	B^{10}	8828.93
c102	A^{19}	8433.00	−18.15	A^{19}	7945.67	−11.32	A^{19}	7137.79
c103	A^{19}	8033.00	−12.45	A^{19}	7790.67	−9.05	A^{19}	7143.88
c104	A^{19}	7384.00	−3.93	A^{19}	7892.91	−11.09	A^{19}	7104.96
rc101	$A^7B^7C^7$	5687.00	−7.71	$A^6B^7C^7$	5569.56	−5.49	$A^4B^7C^7$	5279.92
rc102	$A^5B^6C^8$	5649.00	−9.69	$A^4B^6C^8$	5443.85	−5.71	$A^4B^5C^8$	5149.95
rc103	$A^{11}B^2C^8$	5419.00	−8.33	$A^{11}B^2C^8$	5310.91	−6.17	$A^{10}B^2C^8$	5002.41
rc104	$A^2B^{13}C^3D^1$	5189.00	−3.28	$A^2B^{13}C^3D^1$	5258.81	−4.67	$A^2B^{13}C^3D^1$	5024.25
r201	A^5	4593.00	−21.54	A^5	4070.47	−7.71	A^5	3779.12
r202	A^5	4331.00	−21.01	A^5	3831.74	−7.06	A^5	3578.91
r203	A^4B^1	4220.00	−17.79	A^4B^1	3888.09	−8.53	A^4B^1	3582.51
r204	A^5	3849.00	−22.44	A^5	3395.22	−8.00	A^5	3143.68
c201	A^4B^1	6711.00	−9.29	A^4B^1	7169.22	−16.75	A^4B^1	6140.64
c202	A^1C^3	7720.00	−30.19	A^1C^3	7752.88	−30.74	A^1C^3	5929.82
c203	C^2D^1	7466.00	−2.23	C^2D^1	7654.69	−4.81	C^2D^1	7303.37
c204	A^5	6744.00	−17.88	A^5	6123.23	−7.03	A^5	5721.09
rc201	C^1E^3	5871.00	−6.30	C^1E^3	5795.74	−4.94	C^1E^3	5523.15
rc202	$A^1C^1D^1E^2$	5945.00	−15.84	$A^1C^1D^1E^2$	5700.36	−11.07	$A^1C^1D^1E^2$	5132.08
rc203	$A^1B^1C^5$	5790.00	−28.43	$A^1B^1C^5$	4808.17	−6.65	$A^1B^1C^5$	4508.27
rc204	$A^{14}B^2$	4983.00	−16.76	$A^{14}B^2$	4447.13	−4.21	$A^{14}B^2$	4252.87
Average		5941.25	−12.69		5726.81	−8.62		5272.20

Table 5 Detailed solutions of *a* subclass for the HFUVRPTW

R1			C1			RC1			R2			C2			RC2		
TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix
4539.99	2530.00	$A^1B^{10}C^{12}$	7226.51	5700.00	A^{19}	5253.86	3360.00	$A^6B^8C^6$	3779.12	2250.00	A^5	5820.78	5000.00	A^5	4726.22	3000.00	$A^8B^2C^2$
4375.70	2590.00	$A^1B^2C^{17}$	7137.79	5700.00	A^{19}	5053.48	3390.00	$A^4B^5C^8$	3578.91	2250.00	A^5	5779.59	5000.00	A^5	4518.49	2950.00	$A^9B^3C^1$
4120.63	2550.00	$A^1B^5C^{15}$	7143.88	5700.00	A^{19}	4892.80	3510.00	$A^1B^1C^{11}$	3334.08	2250.00	A^5	5750.58	5000.00	A^5	4327.57	2950.00	A^8B^5
3992.65	2590.00	$B^3C^{15}D^1$	7104.96	5700.00	A^{19}	4783.31	3390.00	$A^4B^1C^{10}$	3143.68	2250.00	A^5	5721.09	5000.00	A^5	4166.73	3100.00	$A^4B^4C^2$
4229.69	2560.00	B^4C^{16}	7171.48	5700.00	A^{19}	5112.91	3360.00	$A^6B^6C^7$	3371.47	2250.00	A^5	5750.53	5000.00	A^5	4645.41	2950.00	$A^9B^3C^1$
4137.96	2560.00	B^4C^{16}	7157.13	5700.00	A^{19}	4997.98	3420.00	$A^2B^6C^8$	3272.79	2250.00	A^5	5757.93	5000.00	A^5	4416.95	3050.00	$A^5B^5C^1$
4061.10	2590.00	$B^3C^{15}D^1$	7135.43	5700.00	A^{19}	4862.67	3420.00	$A^2B^4C^9$	3213.60	2250.00	A^5	5723.91	5000.00	A^5	4338.94	3200.00	$A^1B^4C^3$
3986.07	2680.00	$B^1C^{15}D^2$	7115.71	5700.00	A^{19}	4736.50	3420.00	$A^2B^2C^{10}$	3064.76	2250.00	A^5	5767.78	5000.00	A^5	4109.90	3100.00	$A^4B^4C^2$
4086.72	2630.00	$C^{17}D^1$	7095.55	5700.00	A^{19}				3191.63	2250.00	A^5						
4030.85	2640.00	$B^4C^{13}D^2$							3338.75	2250.00	A^5						
4018.80	2560.00	B^4C^{16}							3061.47	2250.00	A^5						
3961.63	2630.00	$C^{17}D^1$															

Table 6 Detailed solutions of *b* subclass for the HFUVRPTW

R1			C1			RC1			R2			C2			RC2		
TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix
2421.19	552.00	$A^3B^4C^{11}D^3$	2417.52	1440.00	A^8B^6	2469.50	714.00	$A^2B^5C^9$	1965.10	450.00	A^5	1816.14	1080.00	$A^2B^1C^1$	2231.69	650.00	B^3C^4
2219.03	550.00	$A^2B^1C^{13}D^3$	2350.54	1420.00	A^5B^7	2277.79	714.00	$A^2B^4C^8D^1$	1765.09	450.00	A^5	1768.51	1040.00	A^1B^3	2002.62	650.00	B^3C^4
1955.57	616.00	$A^1B^1C^5D^7E^1$	2349.42	1420.00	A^5B^7	2057.55	750.00	C^8D^3	1535.08	450.00	A^5	1744.28	1080.00	$A^2B^1C^1$	1843.72	680.00	$B^2C^2D^2$
1732.26	634.00	$C^3D^9E^1$	2332.94	1380.00	A^7B^6	1914.93	750.00	C^5D^5	1306.72	450.00	A^5	1736.09	1040.00	A^1B^3	1611.28	650.00	$A^1B^1C^5$
2030.83	590.00	$B^1C^8D^7$	2374.01	1420.00	A^5B^7	2337.93	744.00	$A^2B^4C^7D^2$	1575.75	450.00	A^5	1747.68	1000.00	A^5	2195.62	670.00	$A^1B^3C^1D^2$
1924.03	618.00	$C^6D^7E^1$	2381.14	1380.00	A^7B^6	2168.44	720.00	$B^3C^9D^1$	1477.34	450.00	A^5	1756.93	1040.00	A^1B^3	1887.23	660.00	$B^4C^2D^1$
1781.01	634.00	$C^3D^9E^1$	2357.52	1380.00	A^7B^6	2008.39	732.00	$A^1B^1C^7D^3$	1386.84	450.00	A^5	1732.20	1000.00	A^5	1780.72	700.00	C^2D^3
1667.51	644.00	$B^1C^1D^8E^2$	2346.38	1380.00	A^7B^6	1906.69	750.00	$B^1C^6D^4$	1261.09	450.00	A^5	1730.72	1040.00	A^1B^3	1557.74	660.00	C^6
1844.99	628.00	$B^1C^4D^8E^1$	2346.58	1380.00	A^7B^6				1418.51	450.00	A^5						
1792.75	600.00	$B^1C^3D^{10}$							1529.04	450.00	A^5						
1780.03	644.00	$A^1C^3D^7E^2$							1268.14	450.00	A^5						
1677.13	628.00	$C^1D^{10}E^1$															

Table 7 Detailed solutions of *c* subclass for the HFUVRPTW

R1			C1			RC1			R2			C2			RC2		
TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix	TC	FC	Mix
2134.90	293.00	$A^2B^4C^9D^5$	1628.94	800.00	B^{10}	2089.37	396.00	$A^1B^4C^8D^2$	1745.39	270.00	A^8	1269.41	600.00	A^2C^2	1957.60	330.00	$B^4C^2D^1$
1913.37	302.00	$A^3C^8D^7$	1610.96	750.00	A^1B^9	1918.96	366.00	$A^1B^4C^7D^2$	1537.33	250.00	A^4B^1	1252.24	540.00	$A^2B^1C^1$	1699.48	330.00	$A^1B^2C^3D^1$
1633.62	325.00	$A^1C^5D^6E^2$	1611.14	750.00	A^1B^9	1674.50	375.00	$A^2C^8D^3$	1338.42	225.00	A^5	1228.13	540.00	$A^2B^1C^1$	1510.13	325.00	B^3C^4
1382.82	347.00	$B^1C^1D^5E^4$	1610.07	750.00	A^1B^9	1543.55	390.00	$A^2C^4D^6$	1080.66	255.00	A^3C^1	1207.03	520.00	A^1B^3	1256.91	355.00	$C^3D^1E^1$
1729.57	303.00	$B^1C^5D^9$	1628.94	800.00	B^{10}	1972.57	387.00	$A^2B^4C^6D^3$	1350.12	255.00	A^3C^1	1245.51	520.00	A^1B^3	1901.71	350.00	$B^3C^3D^1$
1607.96	311.00	$C^4D^8E^1$	1628.94	800.00	B^{10}	1793.12	372.00	$A^2B^1C^7D^3$	1254.67	225.00	A^5	1229.63	540.00	$A^2B^1C^1$	1598.84	355.00	$C^3D^1E^1$
1455.09	333.00	$A^1C^2D^6E^3$	1628.94	800.00	B^{10}	1635.65	381.00	$A^1B^1C^6D^4$	1186.05	285.00	A^1C^2	1221.16	500.00	A^5	1431.65	355.00	$C^3D^1E^1$
1331.54	353.00	$C^2D^3E^5$	1622.89	730.00	A^3B^8	1531.69	375.00	$B^1C^6D^4$	1022.31	285.00	A^1C^2	1210.72	520.00	A^1B^3	1181.47	355.00	$C^3D^1E^1$
1525.65	325.00	$B^1C^3D^7E^2$	1619.02	750.00	A^1B^9				1233.07	250.00	A^4B^1						
1463.91	342.00	$A^2C^3D^6E^3$							1284.72	250.00	A^4B^1						
1451.92	322.00	$A^3C^3D^7E^2$							1061.70	225.00	A^5						
1355.78	339.00	$C^1D^7E^3$															

5 Conclusions

This paper presented an efficient and effective two-phase Reactive Variable Neighborhood Tabu Search hybrid metaheuristic solution methodology to tackle the HFVRPTW. In the first phase, several initial solutions are produced using a new semi-parallel construction heuristic, followed by an enhanced route elimination procedure for improvement of the overall vehicle utilization. In the second phase, a subset of solutions is selected for improvement using a Reactive Variable Neighborhood Tabu Search hybrid metaheuristic algorithm. The proposed algorithm exploits the systematic neighborhood change and the shaking mechanism offered by the basic VNS scheme, coupled with a TS tuned for intensification local search, and a specialized solution reformation mechanism for diversification and sampling of the solution space driven by the accumulated experience acquired during the search.

The computational experiments conducted on data sets from the literature revealed that the proposed solution methodology outperforms the previously proposed approaches for the HFUVRPTW, with respect to the total distribution costs. The comparison between the proposed semi-parallel construction heuristic and other construction heuristics that have appeared in literature, yielded substantially better results with minimum computational effort. On the other hand, the comparison between heuristics with improvement and metaheuristics with the proposed ReVNTS, indicate that our approach is able to produce consistently high quality solutions that are competitive to the best known results. Especially, the performance on data sets of subclass *a* and all subclasses of R2, C2 and RC2 showed remarkable reduction of the total distribution costs.

Similar are the figures for the HFFVRPTW. According to our survey the HFFVRPTW has never been addressed in literature. All computational experiments used Liu and Shen (1999a) HFUVRPTW data sets, while the reported best fleet size and mix was considered to be the given fixed fleet of vehicles. In particular, the proposed semi-parallel construction heuristic, is able to utilize effectively the given fleet composition producing all times feasible solutions, while achieving even better distribution costs for most problem instances. Similar are the reductions on the total and average distribution costs reached by ReVNTS. This underlines the flexibility of the proposed solution methodology, since it tackles effectively and efficiently both HFUVRPTW and HFFVRPTW. In terms of further research, sophisticated speeding up techniques and additional memory structures within VNS, which will better exploit information gathered during the search, is a worth pursuing research direction.

Acknowledgements The author is indebted to the anonymous reviewers for their valuable comments, suggestions and remarks that helped improve the work presented in this paper.

References

Bent, R., van Hentenryck, P.: A two-stage hybrid local search for the vehicle routing problem with time windows. Transp. Sci. **38**(4), 515–530 (2004)

Blum, C., Roli, A.: Metaheuristics in combinatorial optimisation: overview and conceptual comparison. ACM Comput. Surv. **35**(3), 268–308 (2003)

Brandäo, J., Mercer, A.: A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. Eur. J. Oper. Res. **100**, 180–191 (1997)
Bräysy, O.: Local search and variable neighborhood search algorithms for the vehicle routing problem with time windows. Acta Wasaensia 87, University Wasaenis, Vaasa (2001)
Bräysy, O.: A reactive variable neighborhood search algorithm for the vehicle routing problem with time windows. INFORMS J. Comput. **15**(4), 347–368 (2003)
Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows part I: route construction and local search algorithms. Transp. Sci. **39**(1), 104–118 (2005)
Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M., Soumis, F.: The vehicle routing problem with time windows. In: Toth, P., Vigo, D. (eds.) The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, pp. 157–193. SIAM, Philadelphia (2002)
Cordone, R., Calvo, R.W.: A heuristic for the vehicle routing with time windows. J. Heur. **7**, 107–129 (2001)
Choi, E., Tcha, D.-W.: A column generation approach for the heterogeneous fleet vehicle routing problem. Comput. Oper. Res., doi:10.1016/j.cor.2005.08.002 (2005)
Clarke, G., Wright, J.: Scheduling of vehicles from a central depot to a number of delivery points. Oper. Res. **12**, 568–581 (1964)
Desrochers, M., Verhoog, T.W.: A new heuristic for the fleet size and mix vehicle routing problem. Comput. Oper. Res. **18**(3), 263–274 (1991)
Dell-Amico, M., Monaci, M., Pagani, C., Vigo, D.: Heuristic approaches for the fleet size and mix vehicle routing problem with time windows. Technical Report, Observatorio Logistico, Università di Modena e Reggio Emilia, Italy (2006)
Dullaert, W., Janssens, G.K., Sörensen, K., Vernimmen, B.: New heuristics for the fleet size and mix vehicle routing problem with time windows. J. Oper. Res. Soc. **53**, 1232–1238 (2002)
García-López, F., Melián-Batista, B., Moreno-Pérez, J.A., Moreno-Vega, J.M.: The parallel variable neighborhood search for the p-median problem. J. Heur. **8**, 375–388 (2002)
Gendreau, M., Potvin, J.Y.: Metaheuristics in combinatorial optimization. Ann. Oper. Res. **140**, 189–213 (2005)
Gendreau, M., Laporte, G., Musaraganyi, C., Taillard, E.D.: A tabu search heuristic for the heterogeneous fleet vehicle routing problem. Comput. Oper. Res. **26**, 1153–1173 (1999)
Geysens, F., Golden, B., Assad, A.: A comparison of techniques for solving the fleet size and mix vehicle routing problem. Oper. Res. Spektrum **6**, 207–216 (1984)
Geysens, F., Golden, B., Assad, A.: A new heuristic for determining fleet size composition. Math. Program. Stud. **26**, 233–236 (1986)
Glover, F.: New ejection chain and alternating path methods for travelling salesman problem. In: Balci, O., Sharda, R., Zenios, S. (eds.) Computer Science and Operation Research: New Development in Their Interfaces, pp. 449–509. Pergamon Press, Oxford (1992)
Golden, B., Assad, A., Levy, L., Gheysens, F.: The fleet size and mix vehicle routing problem. Comput. Oper. Res. **11**, 49–66 (1984)
Hansen, P., Mladenović, N.: Variable neighborhood search: principles and applications. Eur. J. Oper. Res. **130**, 449–467 (2001)
Ioannou, G., Kritikos, M., Prastacos, G.: A greedy look-ahead heuristic for the vehicle routing problem with time windows. J. Oper. Res. Soc. **52**, 523–537 (2001)
Kindervater, G.A.P., Savelsbergh, M.W.P.: Vehicle routing: handling edge exchanges. In: Aarts, E., Lenstra, J.K. (eds.) Local Search in Combinatorial Optimization. pp. 337–360. Wiley, UK (1998)
Li, F., Golden, B., Wasil, E.: A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. Comput. Oper. Res., doi:10.1016/j.cor.2005.10.015 (2006)
Liu, F.H., Shen, S.Y.: A method for vehicle routing problem with multiple vehicle types and times windows. Proc. Nat. Sci. Counc. Part A Phys. Sci. Eng. **23**(4), 526–536 (1999a)
Liu, F.H., Shen, S.Y.: The fleet size and mix routing problem with time windows. J. Oper. Res. Soc. **50**(7), 721–732 (1999b)
Osman, I., Salhi, S.: Local search strategies for the vehicle fleet mix problem. In: Rayward-Smith, V.J., Osman, I.H., Reeves, C.R., Smith, G.D. (eds.) Modern Heuristic Search Methods. pp. 131–153. Wiley, New York (1996)
Polacek, M., Hartl, R.F., Doerner, K.: A variable neighborhood search for the multi depot vehicle routing problem with time windows. J. Heur. **10**, 613–627 (2004)
Renaud, J., Boctor, F.F.: A sweep-based algorithm for the fleet size and mix vehicle routing problem. Eur. J. Oper. Res. **140**, 618–628 (2002)

Rochat, Y., Semet, F.: A tabu search approach for delivering pet food and flour in Switzerland. J. Oper. Res. Soc. **45**, 1233–1246 (1994)
Rouseau, L.M., Gendreau, M., Pesant, G.: Using constraint based operators to solve the vehicle routing problem with time windows. J. Heur. **8**, 43–58 (2002)
Salhi, S., Rand, G.: Incorporating vehicle routing into the vehicle fleet composition problem. Eur. J. Oper. Res. **66**, 313–330 (1993)
Salhi, S., Sari, M., Touati, N.: Adaptation of some vehicle fleet mix heuristics. OMEGA **20**, 653–660 (1992)
Semet, F.: A two-phase algorithm for the partial accessibility constrained vehicle routing. Ann. Oper. Res. **61**, 45–65 (1995)
Semet, F., Taillard, E.D.: Solving real-life vehicle routing problems efficiently using tabu search. Ann. Oper. Res. **41**, 469–488 (1993)
Solomon, M.M.: Algorithms for the vehicle routing and scheduling with time windows constraints. Oper. Res. **35**, 254–265 (1987)
Taillard, E.D.: A heuristic column generation method for the heterogeneous fleet VRP. RAIRO **33**, 1–14 (1999)
Tarantilis, C.D.: Solving the vehicle routing problem with adaptive memory programming methodology. Comput. Oper. Res. **32**, 2309–2327 (2005)
Tarantilis, C.D., Kiranoudis, C.T.: BoneRoute: an adaptive memory-based method for effective fleet management. Ann. Oper. Res. **115**, 227–241 (2002)
Tarantilis, C.D., Kiranoudis, C.T.: A flexible adaptive memory-based algorithm for real-life transportation operations: two case studies from dairy and construction sector. Eur. J. Oper. Res., doi:10.1016/j.ejor.2005.03.059 (2005)
Tarantilis, C.D., Kiranoudis, C.T., Vassiliadis, V.S.: A list based threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. J. Oper. Res. Soc. **54**, 65–71 (2003)
Tarantilis, C.D., Kiranoudis, C.T., Vassiliadis, V.S.: A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. Eur. J. Oper. Res. **152**, 148–158 (2004)