Ann Oper Res (2019) 272:243–272
https://doi.org/10.1007/s10479-017-2576-2

ADVANCES IN THEORETICAL AND APPLIED COMBINATORIAL OPTIMIZATION

A skewed general variable neighborhood search algorithm with fixed threshold for the heterogeneous fleet vehicle routing problem

Houda Derbel[1] · Bassem Jarboui[2] · Rim Bhiri[1]

Published online: 14 July 2017

Abstract This article considers the heterogeneous fleet vehicle routing problem, as a variant of a well-known transportation problem: the vehicle routing problem. In order to solve this particular routing problem, a variable neighborhood search with a threshold accepting mechanism is developed and implemented. The performance of the algorithm was compared to other algorithms and tested on datasets from the available literature. Computational results show that our proposed algorithm is competitive and generates new best solutions.

Keywords Metaheuristics · Heterogeneous fleet · Routing · Variable neighborhood search

1 Introduction

The Heterogeneous Fleet Vehicle Routing Problem (HFVRP) is an extension of the Vehicle Routing Problem (VRP). Instead of considering identical vehicles at a central depot, the HFVRP consists of routing a heterogeneous fleet of vehicles with different capacities and costs to supply customers. The work of Baldacci et al. (2008) gives a literature review of the HFVRP and its variants. It reports different solution approches including heuristics and metaheuristics and their performances. Additionally, the authors draw attention to integer programming formulation for the HFVRP while discussing different lower bounds. Recently, a survey on HFVRP was provided by Koç et al. (2016).

✉ Houda Derbel
derbelhouda@yahoo.fr

Bassem Jarboui
bassem_jarboui@yahoo.fr

Rim Bhiri
rimbhiris@gmail.com

1 MODILS, FSEGS, Route de l'aéroport km 4, Sfax 3018, Tunisia

2 Emirates College of Technology, Abu Dhabi, United Arab Emirates

Two features are considered to classify different variants of the problem in the literature: the fleet limitation and the type of costs. Most works in the literature tackled five variants. The different variants are named by using two acronyms: HVRP (Heterogeneous VRP) for problems with limited number of vehicles for each type and FSM (Fleet Size and Mix) for variants with unlimited ones followed by : F for problems with fixed costs and V for those with variable costs. The five variants considered in the literature are as follows:

- HVRPFV: limited fleet with fixed and variable costs
- HVRPV: limited fleet with variable costs but without fixed costs
- FSMFV: unlimited fleet with fixed and variable costs
- FSMF: unlimited fleet with only fixed costs
- FSMV: unlimited fleet with only variable costs

To our knowledge, the first work which dealt with variable neighborhood search (VNS) to solve the HFVRP was provided in Imran et al. (2009). The VNS was enhanced by different local search methods including the sweep algorithm (see Gillett and Miller 1974) and the 2-opt (see Lin 1965) together with the Dijkstra's algorithm (1959) inorder to obtain the initial solution. Two VNS variants which are different in the order of use of the diversification and Dijkstra's algorithm were developed. The authors make use of existing data for the implementation. They proposed some modification for large data instances to better suit the HFVRP particularities. The performance of the VNS algorithm was shown in other domains such as scheduling problems (see Rahmani and Ramezanian 2016).

The HFVRP is classified as $\mathcal{NP}$-hard problem because it is reduced to a classical VRP when the provided fleet is homogenous. In this paper, we propose a skewed generalized variable neighborhood search (SGVNS) metaheuristic for the HFVRP due to its computational complexity. The algorithm is based on the exploration of different neighborhoods introducing local search procedures. We will deal with the variants with limited fleet discussed above. The algorithm is tested on instances from the literature and the results are compared with other existing methods.

The remainder of this paper is organized as follows: Sect. 2 describes some works related to the HFVRP and its main variants. A formal definition of the problem is presented in Sect. 3 while Sect. 4 gives a brief review to the VNS and provides an outline of the proposed metaheuristic. Section 5 contains the results obtained and a comparison with those reported in the literature and the final conclusions are presented in Sect. 6.

2 Literature review

A HFVRP survey touching upon the five variants aforementioned together with the approaches to solutions can be found in Baldacci et al. (2008). The FSM was initially proposed by Golden et al. (1984) to optimize the fleet composition whereas the HFVRP was introduced by Taillard (1999) to determine the optimal set of routes with a fixed fleet. In Golden et al. (1984), the authors proposed a mathematical formulation for the FSMF and efficiently compute some lower bounds. They also developed two heuristic algorithms to solve the FSM. The first one is based on the saving algorithm (see Clarke and Wright 1964) and the second is a two-phase giant-tour based approach. The giant-tour scheme is used by Teodorovic et al. (1995) to solve a stochastic HFVRP. Mathematical programming based methods have been developed by Yaman (2006), Choi and Tcha (2007). In Yaman (2006), the author described six different formulations based on flow variables and Miller–Tucker–Zemlin (MTZ) inequalities to model subtour elimination. In Choi and Tcha (2007), lower bounds for all variants of FSM are obtained using a column generation algorithm enhanced by a set covering formulation. A hybrid algorithm composed by an Iterated Local Search (ILS) based heuristic and a Set Partitioning (SP) formulation was proposed in Subramanian et al. (2012) to solve FSM variants. SP is also combined with tabu search algorithm (TSA) in Lee et al. (2008) to solve HFVRP with variable and fixed costs. More recently, a TSA which strikes a balance between intensification and diversification while using the main concepts of TS was applied in Brandao (2011). In Lee et al. (2008), a slightly improved solution quality is provided.

The HFVRP was first introduced by Taillard (1999) and later studied by Tarantilis et al. (2003, 2004), Li et al. (2007) and Brandao (2011). In Taillard (1999), the authors used a heuristic column generation method to solve medium and large size problem instances. The works of Tarantilis et al. (2003) and Tarantilis et al. (2004) developed two algorithms belonging to the stochastic search methods namely, a listed based threshold accepting (LBTA) and a backtracking adaptive threshold accepting (BATA). The numerical results show that BATA improves solutions in comparison with LBTA and taillard's heuristic. A deterministic tabu search algorithm was proposed by Brandao (2009) to solve the FSMVRP. The author also adapted this algorithm for the HFVRP Brandao (2011). They have shown that solving the HFVRP is much more difficult than solving the FSM. A deterministic variant of the simulated annealing metaheuristic: a a record-to-record travel algorithm (HRTR) was considered in Li et al. (2007). HRTR generated six new best-known solutions in comparison with LBTA and BATA algorithms. Baldacci and Mingozzi (2009) presented an exact algorithm for the HVRPFD based on the set partitioning formulation; they used three types of bounding procedures based on the LP-relaxation and the Lagrangean relaxation. Computaional results have shown that the exact algorithm gives better solutions when it is tested on taillard's instances (Taillard 1999). Notwithstanding that, such algorithms are not only time-consuming but they are not appropriate for solving larger instances. Iterated local search (ILS) was promising in dealing with HFVRP. The first ILS approach using a variable neighborhood descent (VND) with random neighborhood ordering (RVND) in the local search was developed by Penna et al. (2013). Later, Subramanian et al. (2012) proposed a hybrid ILS with a SP formulation (ILS-RVND-SP). These algorithms have been evaluated on the set of instances of Taillard (1999). The latter algorithm improves the result of one instance and is equal to the best known solution (BKS) for the HVRPFD and the FSMFD. Recently, an ILS-based algorithm was designed to solve a real variant of HFVRP where performing multiple trips and being unable to serve particular customers (docking constraints) are allowed (Coelho et al. (2016)). In addition, both HVRPV and HVRPFV problems are tackled in Liu (2013). The author developed a hybrid population heuristic which yielded competitive results with those existing in the literature such as Prins (2009).

In practice, different situations could represent both FSM and HFVRP. The FSM is suitable for strategic decisions when the size and the composition of the vehicle fleet is not yet decided whereas the HFVRP is more adapted for operational decisions when deciding the vehicles needed among existing ones in the fleet. A case study involving a heterogeneous vehicle fleet in the French fourniture industry is presented in Prins (2002). Several real applications can be found in Li et al. (2007). Examples include FedEx Ground and newspaper delivery because the need for different types of vehicles. Tarantilis and Kiranoudis (2007) developed a flexible adaptive memory-based algorithm to solve two case studies from diary and construction company.

Other variants of HFVRP with some additional constraints are also adressed in the literature. A multi-level composite heuristic was developed by Salhi and Sari (1997) to solve the multi-depot vehicle fleet mix problem. A cluster-based optimization approach for the multi-depot heterogeneneous fleet vehicle routing problem with time windows was proposed in Dondo and Cerda (2007). More recently, the vehicle loading problem with a heterogeneous fleet was modeled and solved in Liu et al. (2016).

Specifically, the main idea of our algorithm is to allow moves toward unfeasible solutions using an appropriate penality function. In fact, this function uses control parameters in a dynamic fashion to create a compromise between intensification and diversificaton. When the capacity constraints are violated, we move towards feasible regions as long as those parameters are increased (intensification). Those parameters are adequately decreased as soon as the capacity constraints are respected by the current solution in order to visit new solution regions (diversification). In addition, the SGVNS process accepts moving to worse solutions while remaining within the feasible regions. To this end, we introduce a threshold parameter to accept worse solutions while applying shaking and local search. The way in which these ideas are implemented are described in the next sections.

3 Problem description

The HFVRP can be defined as follows: Given a directed graph $G = (V, E)$ where $V = \{0, 1, \ldots, n\}$ is the set of nodes including the depot represented by the vertex 0 and $V' = V \setminus \{0\}$ is the set of n customers. $E = \{(i, j) : i, j \in V\}$ is the set of arcs. Each customer $i \in V'$ has a demand q_i supplied from the depot ($q_0 = 0$) and each arc (i, j) is associated with a distance d_{ij} ($d_{ii} = 0 \forall i \in V$). The fleet is composed by t different types of vehicles. For each type $k \in T = \{1, \ldots, t\}$, n_k vehicles are located at the depot and each vehicle has a capacity Q_k, a fixed cost cost f_k and a variable cost v_k. Every arc (i, j) has a non-negative travelling cost $c_{ij}^k = v_k d_{ij}$. A route (R, k) is defined by the sequence of visited customers begining and ending at the depot ($R = (i_1, i_2, \ldots, i_{|R|})$, $i_1 = i_{|R|} = 0$) using the vehicle of type k. The HFVRP consists in defining a set of routes while minimizing the total cost such that the following constraints are satified:

(i) The total demand of the customers in a route (R, k) does not exceed the vehicle capacity Q_k,
(ii) Each customer is visited exactly by one route,
(iii) The number of routes assigned to a vehicle k does not exceed n_k.

4 The variable neighborhood search algorithm

The variable neighborhood search is firstly introduced by Mladenović and Hansen (1997). The basic idea of the VNS and its variants is the systematic change of the neighborhood when the search is trapped at a local minimum (see Hansen et al. 2010; Mladenović and Hansen 1997). The main step of the classical VNS starts from an intial solution followed by a shaking procedure and a local search. If the solution is improved, one continues with the first neighborhood; otherwise, a second neighborhood is used and the process is repeated until an acceptance criterion is reached. The procedure is a descent, first improvement method with randomization in the shaking phase. In addition to its simplicity, VNS does not need parameters that influence the efficiency of the implementation.

The variable neighborhood descent (VND) is a deterministic variant of the VNS whereas the reduced variable neighborhood search (RVNS) is a stochastic one. More precisely, let X be the set of feasible solutions, $f(x)$ be the value of the objective function to be minimized and the neighborhood structure $N(x),\ x \in X$, be the set of solutions obtained from x by applying some modifications. The VND consists of finding the best neighbor x' of an initial solution x within a neighborhood $N_k(x)$, $k = 1...k_{max}$. If the solution is improved, the algorithm continues the search with the new obtained solution and $k = 1$, otherwise it iterates with N_{k+1}. The last step is referred to *Change-Neighborhood*(x, x', k). Once a local optimum is found, the possibility of finding promising regions from that will arise. To this end, the RVNS considers a set of neighborhoods N_k, $k = 1...k_{max}$, usually taken in a nested way (i.e each neighborhood contains the previous one). Rather than exploring neighborhoods to get the best neighbor as in the VND, the algorithm randomly chooses a point $x' \in N_1(x)$. If $f(x') < f(x)$ then *Change-Neighborhood*(x, x', k) and the procedure is repeated until $k = k_{max}$. The basic VNS consists of three major steps: shaking, local search and changing neighborhood. After an initial solution is found, a solution x' is randomly generated from the first neighborhood $N_1(x)$ during the shaking phase. The local search is then used with x' as an initial solution to obtain a local optimum x''. Finally, the *Change-Neighborhood*(x, x'', k) is applied. Combining the features of VND in the local search phase and RVNS to improve the initial solution leads to the general VNS (GVNS). Interesting applications of the VNS metaheuristic could be found in Melian and Mladenović (2007). A GVNS heuristic was proposed for the multiple travelling salesman problem in (Soylu 2015) where two objectives: minimizing the longest tour length and minimizing the total length of all tours are taken into consideration. The heuristic was applied in the traffic signalization network of Kayseri province in Turkey and gave good results. The work of Armas and Melián-Batista (2015) considers a variant of VRP with multiple objectives and developed a VNS for a dynamic rich VRP with time windows.

4.1 Skewed general VNS

The size of neighborhoods while selecting neighborhood structures remains important to escape the valley containing local optima. Due to the loss of information when considering larger neighborhoods, VNS turns into multistart. To overcome this problem, the skewed variable neighborhood search (SVNS) enhances the exploration of the set X by visiting distant valleys (Mladenović et al. 1997). In this paper, we adress a skewed general variable neighborhood search (SGVNS). We allow visiting a solution worse than the incumbent, if this solution is far from it according to a distance function ρ. The different steps of SGVNS are presented in Algorithm 1.

VNS algorithm and its variants are used efficiently to solve some of location and routing problems. The first implementation of the VNS algorithm to solve the HFVRP was proposed by Imran et al. (2009). The computational results show that the approach is competitive in comparison with the best known results existing in the literature. In the sequel, we describe the different components of our VNS namely the evaluation function and the neighborhood structures followed by a description of the main algorithm.

4.1.1 Evaluation function

In order to have a successful algorithm to solve a problem, an overlap between intensification and diversification is required. The intensification is provided by intensively exploring some regions of the solution space. More than that, the exploration of different regions of the search space diversifies the search. We define an evaluation function in a way to ensure both intensification and diversification. Indeed, in addition to the fixed and varible costs, we added a penality generated with the violation of the capacity constraints for vehicles. Let S be one

Algorithm 1: SGVNS general structure

1 Initialization. Select the set of neighborhood structures N_k, for $k = 1, ..., k_{max}$, that will be used in the shaking phase, and the set of neighborhood structures N'_l for $l = 1, ..., l_{max}$ that will be used in the local search; find an initial solution x and improve it by using RVNS; set $x_{opt} \leftarrow x$, $f_{opt} \leftarrow f(x)$ choose a stopping condition and a parameter value θ;

2 Repeat the following sequence until the stopping condition is met:

1. Set $k \leftarrow 1$;
2. Repeat the following steps until $k = k_{max}$:

(a)(a) Shaking. Generate a point x' at random from the k^{th} neighborhood $N_k(x)$ of x;

(b)(b) Local search by VND:

i. b1) Set $l \leftarrow 1$;

ii. b2) Repeat the following steps until $l = l_{max}$:;

. Exploration of neighborhood. Find the best neighbor x'' of x' in $N'_l(x')$;

. Move or not. If $f(x^{''}) < f_{opt}$ set $x_{opt} \leftarrow x^{''}$ and $l \leftarrow 1$; otherwise set $l \leftarrow l + 1$;

(c)(c) Move or not. If $f(x^{''}) - \theta\rho(x, x^{''}) < f(x)$ then $x \leftarrow x^{''}$ and continue the search with $N_1(k \leftarrow 1)$; otherwise, set $k \leftarrow k + 1$;

candidate solution, $f(S)$ be the sum of costs previously described in the Sect. 3 and $D(R, k)$ the total demand of the customers in a route (R, k), we denote by $F(S)$ the evaluation function which is calculated as follows:

$$F(S) = f(S) + pen(S)$$

where $pen(S)$ is a dynamic penality defined by

$$\alpha \sum_{i=0}^{n_k} \max(0, D(R, k) - Q_k) \tag{1}$$

and α represents a penality of violating the capacity of a vehicle k. As the evaluation function is defined, we allow to visit infeasible solutions that exceed the capacity of a vehicle.
The penality will increase rapidly if the parameter α is fixed to a constant value. Consequently, we propose to create an oscilliation between the feasible and infeasible space by changing α dynamically in the following way: $\alpha = \alpha(1 + \beta)$ if $D(R, k) > Q_k$ otherwise, we set $\beta = \beta(1 - \varepsilon)$. We begin by increasing the parameter β to quickly achieve the feasible solution space. After that, we enlarge the search space by decreasing it when no better solutions could be found . So, the current region as well as distant regions are thoroughly explored. This mechanism is also known as strategic oscillation involved in Glover (1977) and used since then for a variety of Tabu Search procedures.

4.1.2 Neighborhood structures

In this study, we performed neighborhood structures involving inter and intra-route moves. The aim was to enhance the cost of routes. We have considered neighborhood structures based on insertion, exchange and shift operators (see Penna et al. (2013)). An inter-move involves two different route R_1 and R_2 whereas an intra-move is perfomed inside the same route. The different neighborhoods are depicted in the Figs. (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14) below where one route is represented by a vector beginning and ending with 0 which indicates the depot and the other components depict customers and are described as follows:

1. *2-opt* (N_1): A neighbor of a solution is generated by replacing two non adjacent arcs by two new others within the same route.

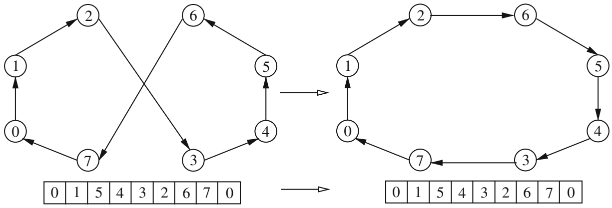

Fig. 1 2-opt intra-route

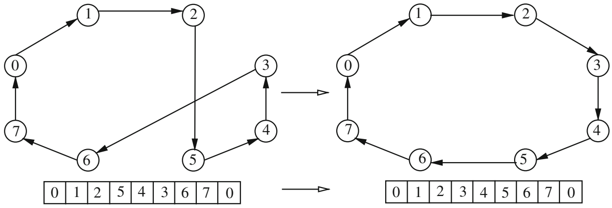

Fig. 2 Swap move of a customer intra-route

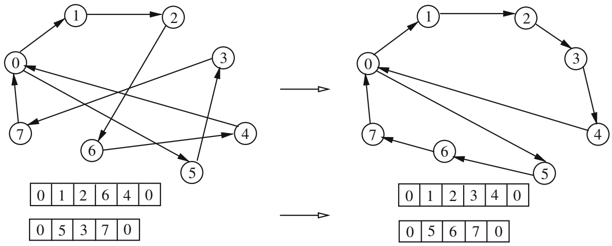

Fig. 3 Swap move of a customer inter-route

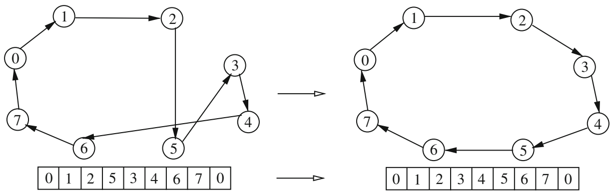

Fig. 4 Shift move of a customer intra-route

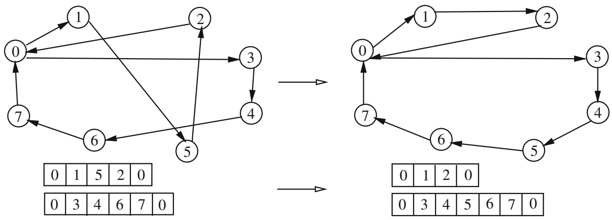

Fig. 5 Shift move of a customer inter-route

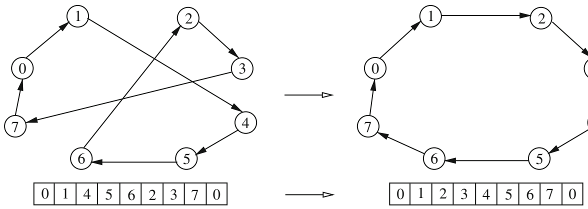

Fig. 6 Extended Or-opt intra-route

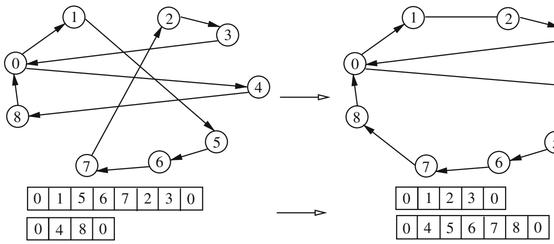

Fig. 7 Extended Or-opt inter-route

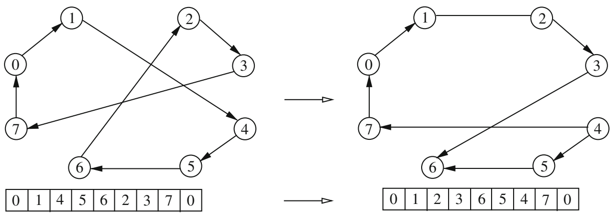

Fig. 8 Inverse Extended Or-opt intra-route

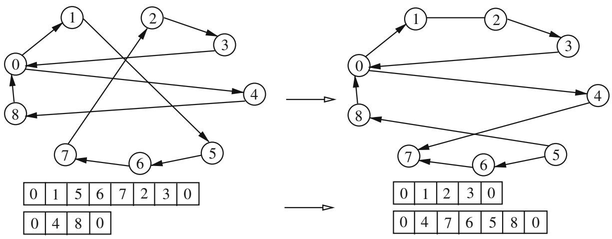

Fig. 9 Inverse Extended Or-opt inter-route

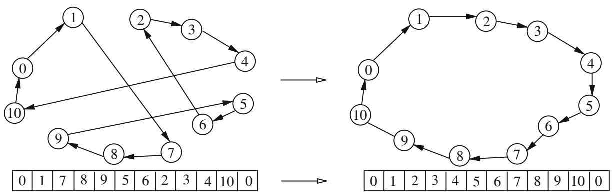

Fig. 10 k-Swap intra-route

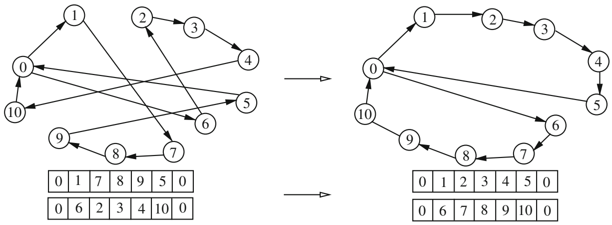

Fig. 11 *k*-Swap inter-route

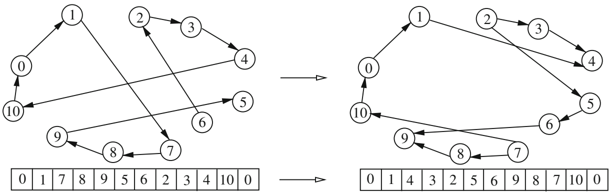

Fig. 12 Reverse *k*-Swap intra-route

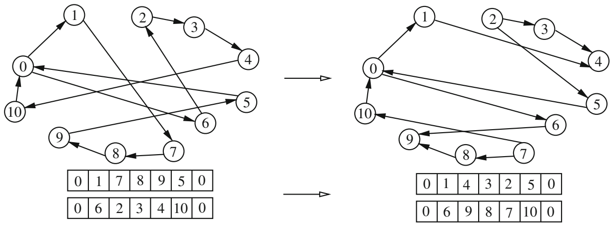

Fig. 13 Reverse *k*-Swap inter-route

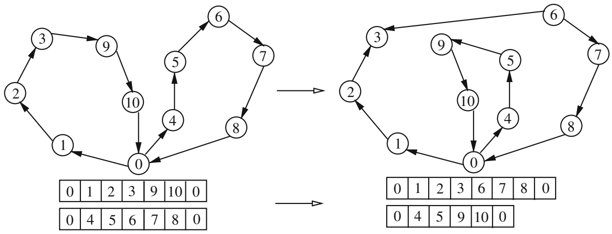

Fig. 14 cross-route move

2. *Swap move of a customer* (N_2): This neighborhood structure corresponds to a permutation of two customers in the same route or in different routes.
3. *Shift move of a customer* (N_3): A neighbor of a solution is generated by switching a customer from its position and inserting it into a new one. This move can be intra-route or inter-route.
4. *Extended Or-opt* (N_4): This neighborhood corresponds to an insertion move but it considers a set of customers rather than only one as with N_1. The neighborhood N_4 consists in removing a set of consecutive customers and inserting it between two other nodes. This move is applied inside the same route or between two different routes. The *k*-shift move (Penna et al. 2013) is a special case of N_4.

5. *Inverse Extended Or-opt* (N_5): In this case, we consider a transfer of a set of consecutive customers (bone) from their current position and then we reinsert them in a reverse order starting from the last customer and finishing with the first one.
6. *k-Swap* (N_6): A new solution is obtained by applying a permutation between two bones inside the route or between two different routes. Both bones are reinserted following the same order of visit of customers.
7. *Reverse k-Swap* (N_7): This neighborhood structure considers a swap of two bones as within neighborhood N_6 but with a rearrangement of the visit order. The extracted bone is reinserted in such way that the last customer is the first one.
8. *Swap move of two routes* (N_8): This move consists of exchanging two routes between two vehicles with different capacities.
9. *Cross-route* (N_9): Two arcs, (i, j) and (i', j') belonging to two routes are removed. After that, the routes are reconnected by adding the arcs (i, j') and (i', j). It is reported that the cross is applied between the closest nodes between routes.

4.2 The SGVNS for the HFVRP

This section describes the SGVNS algorithm for the HFVRP and the ensuing steps. To implement our algorithm, we essentially outline three procedures: the shaking phase, the local search phase and the move or not phase. The local search phase corresponds to a VND where neighborhoods N_1,...,N_7 are applied in a sequential way. The VND explores the next neighborhood unless the previous one fails to improve the current solution. In order to escape from the current local optimal solution, we consider larger neighborhoods in the shaking phase. The perturbation is applied using neighborhoods N_3, N_8 and N_9 as follows: we consider three types of shaking; each one according to a probability $Pr(N_i), i \in \{3, 8, 9\}$. The solution space of the third neighborhood is exhaustively explored. Indeed, we apply the insertion move N_3 inter-route, k times for customers in a given route. We denote by $N_3^k, k = 1, \ldots, 5$ this particular case. This neighborhood is aiming at generating a new sequence of customers within a route. The use of the neighborhood N_8 is aiming at changing the assignment of vehicles to routes. In addition, the cross neighborhood N_9 is an attempt to explore other solutions by changing the structure of the routes while exchanging arcs between routes. In the move or not phase, we accept to visit worse solutions without escaping feasible regions according to a threshold parameter τ. The pseudocode of the SVNS for the HFVRP is presented in Algorithm 2.

5 Computational results

Our algorithm was coded in C++ and executed on a core i7 with 3.00 GHz. For each instance, the proposed algorithm is excuted 10 times and the result is rounded up to next higher digit. We test the SGVNS in instances decribed in Sect. 5.1. A comparison with the best known algorithms performed in the literature is reported in Sect. 5.2. The following notations are given to manage the computational results and the comparison with existing ones in the literature. "Inst", denotes the name of the instance, n is the number of customers, BKS denotes the best known solution in the literature, "Best Sol" and "Time" represent, respectively, the objective value of the best solution and the average computational time associated with to the corresponding work. "First time" indicates the first time the best solution was found by the SGVNS. $Gap(\%)$ records the percent deviation of an algorithm between its best value and BKS and is given by the following formula: $\frac{BsetSol-BKS}{BKS} \times 100$. For Tables 4, 5, 6, and 7,

Algorithm 2: SGVNS for the HFVRP

Initialization. Select the set of neighborhood structures N_k, for $k = 1, \ldots, 9$
Find an initial solution S;
set $S_{opt} \leftarrow S$, choose a parameter value α and β;
Repeat the following sequence until the stopping condition is met
$k \leftarrow 1$;
While $k \leq k_{max}$ **do**
$S' \leftarrow$ **Shaking**$(S, N_3^{k\in\{1,\ldots,5\}}, N_8, N_9, Pr(N_i), i \in \{3, 8, 9\})$;
$S'' \leftarrow$ **Seq-VND**$(S', N_{k\in\{1,\ldots,7\}})$;
If $F(S'') < F(S_{opt})$ **Then**
$S_{opt} \leftarrow S''$;
If $F(S'') < F(S)(1 + \tau)$ **Then**
$S \leftarrow S''$;
$k \leftarrow 1$;
Else $k \leftarrow k + 1$;
Update the parameter α;

the last rows: “Average”, “Avg. deviation” and “computer resource” specifie the average cost and time for each set of problems, the percent deviation of the average cost to the average of the BKS costs and the computer resource used for every solution method respectively.

Avg cost and Avg gap(%) denote, respectively the average solution cost of the 10 runs and the gap between the former value and BKS. We observed on preliminary experiments that the following calibration for our algorithm, which we adopt in the following experiments, yield the best results: β is set initially to 0.005, the parameter τ is set to $\frac{1}{4n}$ with n is the number of customers and the different probabilities are set to: $Pr(N_3) = 0.1$, $Pr(N_8) = 0.1$, $Pr(N_9) = 0.8$.

5.1 Benchmark instances

We tested our algorithm on benchmark instances generated by (Taillard 1999; Li et al. 2007; Brandao 2011). There are three types of instances: the first set numbered from 13 to 20 represents instances with customers between 50 et 100 (see Taillard (1999), the second set contains instances named H_i, $i \in \{1, \ldots, 5\}$ with 200–360 customers and are proposed by Li et al. (2007) and the third set identified as $N1$-N_5 was created by Brandao (2011). The characteristics of each set of instances are in Tables 1, 2 and 3 respectively. The last column represents the ratio of total demand and total capacity in percent. For the third set, the authors assume that this ratio is slightly higher than the two others.

5.2 A comparative study

We perform a comparison of our solution approach with the best heuristics available in the literature to the best of our knowledge. For the first set of instances and the case of the HVRPV, we compare our results with those given by a backtracking adaptive threshold accepting algorithm (BATA), heuristic column generation (HCG), a record-to-record travel algorithm (HRTR), a hybrid algorithm composed by an Iterated Local Search (ILS) based heuristic and a Set Partitioning (SP) model (IL-RVND-SP) and a Population heuristic given by Taillard (1999), Tarantilis et al. (2004), Li et al. (2007), Subramanian et al. (2012) and Liu (2013) respectively. The best solutions are recorded in blodface and the solution improved

Table 1 Instances from Taillard (1999)

Problem	n	Type of vehicle																								Ratio (%)
		A				B				C				D				E				F				
		Q_A	f_A	v_A	n_A	Q_B	f_B	v_B	n_B	Q_C	f_C	v_C	n_C	Q_D	f_D	v_D	n_D	Q_E	f_E	v_E	n_E	Q_F	f_F	v_F	n_F	
13	50	20	20	1.0	4	30	35	1.1	2	40	50	1.2	4	70	120	1.7	4	120	225	2.5	2	200	400	3.2	1	95.39
14	50	120	1000	1.0	4	160	1500	1.1	2	300	3500	1.4	1													88.45
15	50	50	100	1.0	4	100	250	1.6	3	160	450	2.0	2													94.76
16	50	40	100	1.0	2	80	200	1.6	4	140	400	2.1	3													94.76
17	75	50	25	1.0	4	120	80	1.2	4	200	150	1.5	2	350	320	1.8	1									95.38
18	75	20	10	1.0	4	50	35	1.3	4	100	100	1.9	2	150	180	2.4	2	250	400	2.9	1	400	800	3.2	1	95.38
19	100	100	500	1.0	4	200	1200	1.4	3	300	2100	1.7	3													76.74
20	100	60	100	1.0	6	140	300	1.7	4	200	500	2.0	3													95.92

Table 2 Instances from Li et al. (2007)

Problem	n	Type of vehicle																		Ratio (%)
		A			B			C			D			E			F			
		Q_A	v_A	n_A	Q_B	v_B	n_B	Q_C	v_C	n_C	Q_D	v_D	n_D	Q_E	v_E	n_E	Q_F	v_F	n_F	
$H1$	200	50	1.0	8	100	1.1	6	200	1.2	4	500	1.7	3	1000	2.5	1				93.02
$H2$	240	50	1.0	10	100	1.1	5	200	1.2	5	500	1.7	4	1000	2.5	1				96.00
$H3$	280	50	1.0	10	100	1.1	5	200	1.2	5	500	1.7	4	1000	2.5	2				93.33
$H4$	320	50	1.0	10	100	1.1	8	200	1.2	5	500	1.7	2	1000	2.5	2	1500	3	1	94.12
$H5$	360	50	1.0	10	100	1.2	8	200	1.5	5	500	1.8	1	1500	2.5	2	2000	3	1	92.31

Table 3 Instances from Brandao (2011)

Problem	n	Type of vehicle																		Ratio (%)
		A			B			C			D			E			F			
		Q_A	v_A	n_A	Q_B	v_B	n_B	Q_C	v_C	n_C	Q_D	v_D	n_D	Q_E	v_E	n_E	Q_F	v_F	n_F	
$N1$	150	50	1	5	100	1.5	4	150	1.9	4	200	2.2	3	250	2.6	2				95.11
$N2$	199	50	1	8	100	1.5	6	150	1.9	5	200	2.2	4	250	2.6	2	350	3.2	1	93.71
$N3$	120	50	1	6	100	1.5	3	150	1.9	3	200	2.2	2							94.83
$N4$	100	50	1	4	120	1.6	4	180	2.1	4	240	2.6	2							96.28
$N5$	134	900	1	5	1500	1.5	3	2000	1.8	2	2500	2.2	1							94.32

with the SGVNS are underlined. The results in Table 4 show that the SGVNS find all the best known solutions except one which is the instance 19. For this instance, we find the same solution as methods developed since 2007 but the solution cost found by Taillard (1999) remains the best. However, the SGVNS improves all the results found by Taillard (1999). Our solution method is as good as or better than BATA, HRTR and ILS-RVNS-SP. When compared with Population heuristic, the SGVNS produces four better solutions and four identical. As for the HVRPFV, we compare our results with those given by ILS-RVND-SP, population heuristic and adaptive memory programming metaheuristic (MAMP) developed by Li et al. (2010). Table 5 shows that our algorithm finds solution values equal to the best known solutions and improves the result for the instance 20.

For instances presented in Tables 3 and 6 shows that the SGVNS is able to find best known solutions or to improve it except for one instance. In particular, the SGVNS outperforms the TSA of Brandao (2011) and it can be observed that our algorithm is competitive with the ILS-RVNS-SP heuristic. In fact, it improves one solution, gives three identical ones and is a slightly worse for the instance $N1$. For larger instances described in Tables 2 and 7 shows the performance of our algorithm to find four new best solutions among five. The new best solutions are introduced in the appendix.

Overall, our algorithm failed to obtain the best known solution of only one instance. Hence the SGVNS proved to be performant.

In addition, it is worth mentioning that the solution costs in light of reported compuation times for previous works are not comparative. For example, in the case of the instance H5 it can be observed that the TSA found the BKS in 13321 s which is equivalent to $37n$ while the ILS-RVND-SP did not after 621.17 s. Therefore, we propose to run our algorithm for three different parameters of time (n, $2n$, $4n$) to be able to assess the efficiency of our results. The average results and percent deviations of our algorithm over different times for the different instances are given in Tables 8, 9, 10 and 11. Our results are summarized in terms of average gap in Fig. 15. In this figure, the instances from Taillard (1999) are labeled from 13 to 20 followed by the letter F for the case of HVRPFV variant. Firstly, as we can see, the improvement of results are meaningful especially for the instances of Li et al. (2007). This can be interpreted that the increase of time can be proved to provide considerable improvements when instances are large.

6 Conclusion

The heterogeneous fleet vehicle routing problem (HFVRP) is a more sophisticated variant of vehicle routing. The problem arises when a set of heterogeneous fleet of vehicles with different capacities and costs are routed to supply customers from a central depot. In this paper, we propose a skewed version of variable neighborhood search (SGVNS) and we have considered the HFVRP variant with limited fleet with fixed and/or variable costs. We explore different neighborhood structures in an exhaustive way in order to provide a balance between intensification and diversification of the solution space. After implementing a SVNS, we provide computational results showing the performance of our algorithm. The SGNS improves five best known solutions for the large instances, one for the small instances and is as good as the best algorithms with a reasonable computation time. Our approach is clearly efficient compared with those cited in this paper. In the future, we propose to study the mixed fleet variant of the HFVRP including additional characteristics presented in practical situations.

Table 4 Results for HVRPV on the instances of Taillard (1999)

Inst.	n	BKS	HCG (Taillard 1999)		BATA (Tarantilis et al. 2004)		HRTR (Li et al. 2007)		ILS-RVND-SP (Subramanian et al. 2012)		Population heuristic (Liu 2013)		Our best: SGVNS		Gap (%)
			Best sol.	Time[a] (s)	Best sol.	Time (s)	Best sol.	Time (s)	Best sol.	Time(s)[b]	Best sol.	Time (s)	Best sol.	Time[b] (s)	
13	50	**1517.84**	1518.05	473	1519.96	843	**1517.84**	358	**1517.84**	1.33	**1517.84**	57.42	**1517.84**	2.98	0.00
14	50	**607.53**	615.64	575	611.39	387	**607.53**	141	**607.53**	1.09	**607.53**	86.98	**607.53**	8.29	0.00
15	50	**1015.29**	1016.86	335	**1015.29**	368	**1015.29**	166	**1015.29**	2.13	1015.83	4.85	**1015.29**	0.97	0.00
16	50	**1144.94**	1154.05	350	1145.52	341	**1144.94**	188	**1144.94**	1.41	1148.57	13.51	**1144.94**	10.74	0.00
17	75	**1061.96**	1071.79	2245	1071.01	363	1061.96	216	1061.96	4.22	1061.96	115.88	**1061.96**	57.16	0.00
18	75	**1823.58**	1870.16	2876	1846.35	971	**1823.58**	366	**1823.58**	4.06	**1823.58**	97.98	**1823.58**	21.98	0.00
19	100	**1117.51**	**1117.51**	5833	1123.83	428	1120.34	404	1120.34	9.12	1120.34	77.21	1120.34	45.55	0.03
20	100	**1534.17**	1559.77	3402	1556.35	1156	**1534.17**	447	1534.17	8.89	**1534.17**	115.49	**1534.17**	64.39	0.00
Average		1227.85	1240.48	2011.12	1236.21	607.12	1228.21	285.75	1228.21	4.03	1228.73	71.16	1228.21	26.51	0.00
Avg. deviation (%)			0.12		0.08		0.00		0.00		0.01		0.00		
Computer Resource			Sun Sparc workstation, 50 MHz		Pentium III, 550 MHz, 128MB RAM		Athlon, 1 GHz, 256MB RAM		Intel Core i7, 2.93 GHz		Intel Pentium 4,3GHz		Intel Core i7, 3GHz		

Bold indicates the best results among all the approaches

[a] Average time of 5 runs

[b] Average time of 10 runs

Table 5 Results for HVRPFV on the instances of Taillard (1999)

Inst.	n	BKS	MAMP (Li et al. 2010)		ILS-RVND-SP (Subramanian et al. 2012)		Population heuristic (Liu 2013)		Our best: SGVNS		Gap (%)
			Best sol.	Time[a] (s)	Best sol.	Time (s)	Best sol.	Time (s)	Best sol.	Time(s)[a]	
13	50	**3185.09**	3185.09	110	3185.09	1.99	**3185.09**	129.88	**3185.09**	12.54	0.00
14	50	**10107.53**	**10107.53**	34	**10107.53**	1.29	**10107.53**	51.78	**10107.53**	4.03	0.00
15	50	**3065.29**	**3065.29**	46	**3065.29**	1.77	3065.83	87.29	**3065.29**	1.09	0.00
16	50	**3265.41**	**3265.41**	99	**3265.41**	1.67	3268.70	73.85	**3265.41**	23.38	0.00
17	75	**2076.96**	2076.96	148	2076.96	5.95	**2076.96**	128.65	**2076.96**	71.74	0.00
18	75	**3743.58**	**3743.58**	119	**3743.58**	16.47	**3743.58**	115.31	**3743.58**	34.08	0.00
19	100	**10420.34**	**10420.34**	287	**10420.34**	15.80	**10420.34**	238.67	**10420.34**	57.96	0.00
20	100	**4761.26**	4832.17	200	**4761.26**	16.87	4834.17	190.96	4760.**68**	558.66	−0.006
Average		5078.19	5087.05	130.37	5078.18	7.72	5087.77	127.09	5078.11	95.43	0.00
Avg. deviation (%)			0.09		−0.0001		0.09		−0.0008		
Computer Resource			Intel, 2.2GHz		Intel Core i7, 2.93 GHz		Intel Pentium 4,3GHz		Intel Core i7, 3GHz		

Bold indicates the best results among all the approaches
[a] Average time of 10 runs

Table 6 Results for HVRPV on the instances of Brandao (2011)

Inst.	n	BKS	TSA (Brandao 2011)		ILS-RVND-SP (Subramanian et al. 2012)		Our best: SGVNS		Gap (%)
			Best sol.	Time (s)	Best sol.	Time[a] (s)	Best sol.	Time[a] (s)	
N1	150	**2235.87**	2243.76		**2235.87**	51.50	**2235.87**	325.85	0.00
N2	199	**2864.83**	2874.13		**2864.83**	102.77	<u>2856.81</u>	241.03	−0.08
N3	120	**2378.99**	2386.90		**2378.99**	51.71	**2378.99**	133.07	0.00
N4	100	**1839.22**	**1839.22**		**1839.22**	9.64	**1839.22**	223.83	0.00
N5	134	**2047.81**	2062.48		**2047.81**	52.33	**2047.81**	130.91	0.00
Average		2273.34	2281.29		2273.34	57.19	2271.74	210.93	−0.016
Avg. deviation (%)			0.08			0.00		−0.016	
Computer Resource			Compaq Presario, 4GHz, 512MB		Intel Core i7, 2.93 GHz		Intel Core i7, 3GHz		

Bold indicates the best results among all the approaches
[a] Average time of 10 runs

Table 7 Results for HVRPV on the instances of Li et al. (2007)

Inst.	n	BKS	TSA (Brandao 2011)		ILS-RVND-SP (Subramanian et al. 2012)		Our best: SGVNS		Gap (%)
			Best sol.	Time (s)	Best sol.	Time[a] (s)	Best sol.	Time[a] (s)	
H1	200	**12050.08**	**12050.08**	1395	**12050.08**	72.10	**12050.08**	219.87	0.000
H2	240	**10226.17**	**10226.17**	3650	10329.15	176.43	10224.69	393.39	−0.01
H3	280	**16230.21**	**16230.21**	2822	16282.41	259.61	16229.72	346.28	−0.05
H4	320	**17458.65**	**17458.65**	8734	17743.68	384.52	17444.92	561.09	−0.14
H5	360	**23220.72**	**23220.72**	13321	23493.87	621.17	23112.74	607.65	−1.08
Average		15837.17	15837.17	5984.4	15979.84	302.77	15812.43	425.65	−0.25
Avg. deviation (%)			0.00		1.42		−0.25		
Computer Resource			Compaq Presario, 4GHz, 512MB		Intel Core i7, 2.93 GHz		Intel Core i7, 3GHz		

Bold indicates the best results among all the approaches
[a] Average time of 10 runs

Table 8 Average results and Gaps for different parameters of time for the HVRPV: instances of Taillard (1999)

Inst.	BKS	Time$_{<n}$ (s)					Time$_{<2n}$ (s)					Time$_{<4n}$ (s)				
		Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)
13	1517.84	1517.84	0.724	−0.0001	1517.84	0.000	1517.84	0.724	0.000	1517.84	0.000	1517.84	0.724	0.000	1517.84	0.000
14	607.52	607.53	3.13	0.0001	607.53	0.0001	607.53	3.13	0.0001	607.53	0.0001	607.53	3.13	0.0001	607.53	0.000
15	1015.29	1015.29	0.58	0.000	1015.29	0.000	1015.29	0.58	0.000	1015.29	0.000	1015.29	0.58	0.000	1015.29	0.000
16	1144.94	1144.92	6.18	−0.0002	1144.92	−0.0002	1144.92	6.18	−0.0002	1144.92	−0.0002	1144.92	6.18	−0.0002	1144.92	−0.0002
17	1061.95	1061.95	45.55	0.000	1063.32	0.0137	1061.95	45.55	0.000	1062.93	0.0098	1061.95	45.55	0.000	1062.54	0.0059
18	1823.58	1823.58	5.25	0.000	1823.58	0.000	1823.58	5.25	0.000	1823.58	0.000	1823.58	5.25	0.000	1823.58	0.000
19	1117.51	1120.34	2.86	0.029	1122.96	0.0545	1120.34	2.86	0.029	1121.85	0.0434	1120.34	2.86	0.029	1120.34	0.0283
20	1534.17	1534.17	8.79	0.0001	1536.68	0.0252	1534.17	8.79	0.000	1536.68	0.0252	1534.17	8.79	0.000	1535.03	0.0087

Table 9 Average results and Gaps for different parameters of time for the HVRPFV: instances of Taillard (1999)

Inst.	BKS	Time$_{<n}$ (s)					Time$_{<2n}$ (s)					Time$_{<4n}$ (s)				
		Best sol.	First time (s)	Gap (%)	Avg.cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)
13	3185.09	3185.09	1.66	0.000	3185.31	0.0023	3185.09	1.66	0.000	3185,09	0.000	3185.09	1.66	0.000	3185,09	0.000
14	10107.53	10107.53	0.60	0.000	10107.53	0.000	10107.53	0.60	0.000	10107.53	0.0000	10107.53	0.60	0.000	10107.53	0.000
15	3065.29	3065.29	0.64	0.000	3065.29	0.000	3065.29	0.64	0.000	3065.29	0.000	3065.29	0.64	0.000	3065.29	0.000
16	3265.41	3265.41	1.74	0.000	3268.92	0.0351	3265.41	1.74	0.000	3267,50	0.0209	3265.41	1.74	0.000	3265,41	0.000
17	2076.96	2077.97	39.35	0.010	2080.51	0.0356	2076.96	112.86	0.000	2079.79	0.0284	2076.96	112.86	0.000	2077.44	0.0049
18	3743.58	3743.58	9.92	0.000	3745.93	0.0235	3743.58	9.92	0.000	3745.93	0.0235	3743.58	9.92	0.000	3744.86	0.0128
19	10420.34	10420.34	15.01	0.000	10430.67	0.1033	10420.34	15.01	0.000	10425.88	0.0554	10420.34	15.01	0.000	10425.56	0.0522
20	4761.26	4761.26	15.13	0.000	4774.78	0.135	4766.58	91.39	0.053	4772.86	0.116	4760.67	558.66	−0.006	4770.53	0.093

Table 10 Average results and Gaps for different parameters of time for the instances of Li et al. (2007)

Inst.	BKS	Time$_{<n}$ (s)					Time$_{<2n}$ (s)					Time$_{<4n}$ (s)				
		Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg.cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)
$H1$	12050.08	12050.39	26.84	0.0031	12056.32	0.0624	12050.39	26.84	0.0031	12053.84	0.0345	12050.08	447.93	0.0001	12051.56	0.0148
$H2$	10226.17	10288.48	220.49	0.623	10347.32	1.2115	10285.56	439.00	0.5939	10329.50	1.0333	10224.68	906.63	−0.0149	10282.34	0.5617
$H3$	16230.21	16242.88	270.19	0.1267	16355.10	1.2489	16231.92	475.84	0.0171	16317.92	0.8771	16229.71	688.41	−0.005	16304.64	0.7443
$H4$	17458.65	17592.89	287.75	1.34	17790.58	3.3193	17530.96	589.98	0.7231	17733.42	2.7477	17444.92	1260.81	−0.1373	17664.36	2.0571
$H5$	23220.72	23335.90	291.44	1.15	23697.22	4.765	23160.66	658.98	−0.6006	23464.92	2.7477	23112.73	1426.22	−1.0799	23348.17	1.2745

Table 11 Average results and Gaps for different parameters of time for the instances of Brandao (2011)

Inst.	BKS	Time$_{<n}$ (s)					Time$_{<2n}$ (s)					Time$_{<4n}$ (s)				
		Best sol	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)	Best sol.	First time (s)	Gap (%)	Avg cost	Avg gap (%)
$N1$	2235.87	2236.12	135.13	0.0025	2247.87	0.12	2236.12	135.13	0.0025	2244.30	0.0843	2235.87	325.85	0.000	2237.66	0.0179
$N2$	2864.83	2863.93	118.18	−0.009	2875.94	0.1111	2856.81	367.14	−0.0802	2871.55	0.0672	2856.81	367.14	−0.0802	2867.40	0.0257
$N3$	2378.99	2379.06	36.85	0.0007	2385.65	0.0666	2379.00	211.67	0.0001	2383.25	0.0426	2378.99	245.41	0.000	2383.11	0.0412
$N4$	1839.22	1839.22	63.87	0.000	1839.22	0.000	1839.22	68.87	0.000	1839.22	0.000	1839.22	223.83	0.000	1839.22	0.000
$N5$	2047.81	2047.81	24.42	0.000	2053.07	0.0514	2047.79	242.28	−0.0002	2052.95	−0.0145	2047.81	242.28	0.000	2047.80	−0.0001

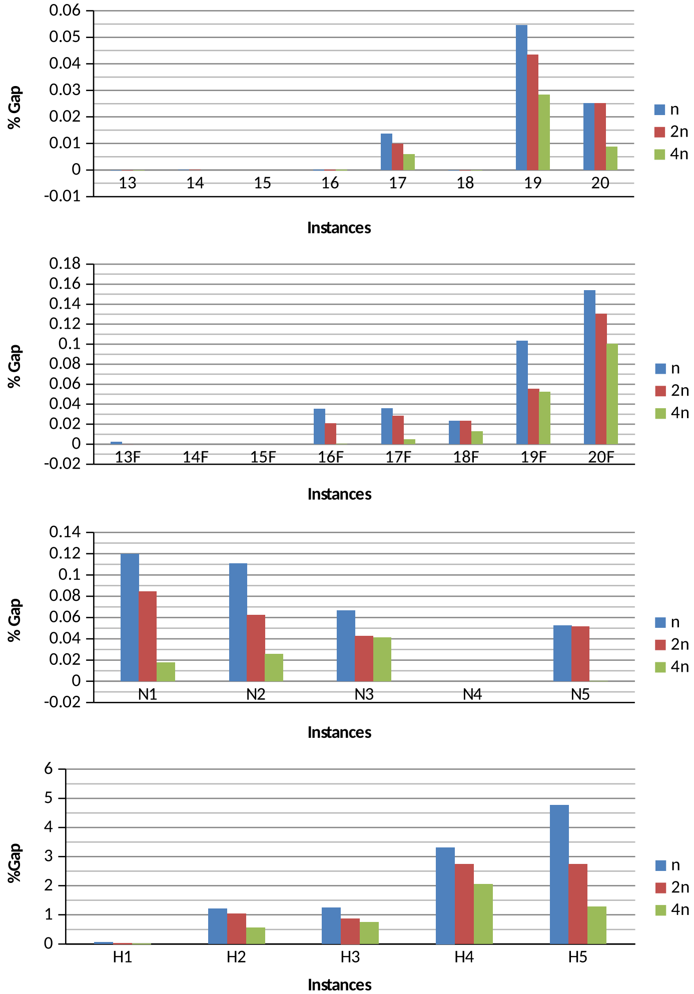

Fig. 15 Results for SGVNS for three parameters of time : n, $2n$ and $4n$

Appendix A

Route number	Sequence of customers	Load	Vehicle type
Instance 20 with fixed and variable cost Solution cost = 4760.68			
1	0-6-84-17-38-14-43-42-0	60	A
2	0-2-57-15-41-67-25-55-0	60	A
3	0-70-63-64-49-36-46-0	60	A
4	0-50-78-34-29-24-54-0	60	A
5	0-89-60-83-45-8-7-0	59	A
6	0-12-80-68-79-3-77-76-28-0	140	B
7	0-62-11-19-47-48-82-18-0	139	B
8	0-52-88-31-10-90-32-30-1-69-27-0	140	B
9	0-33-81-9-35-71-65-66-20-51-0	140	B
10	0-13-95-59-100-91-44-86-16-61-5-0	200	C
11	0-53-58-87-97-92-37-98-85-93-99-96-94-0	200	C
12	0-26-4-39-23-56-75-22-74-72-73-21-40-0	200	C
Instance N2 Solution cost=2856.8125			
1	0-144-57-15-43-42-117-0	46	A
2	0-53-149 -26-0	49	A
3	0-132-69-1-176-0	46	A
4	0-156-112-0	48	A
5	0-154-138-0	45	A
6	0-91-38-140-86-113-17-173-84-60-0	100	B
7	0-94-183-6-147-89-0	97	B
8	0-167-31-190-127-0	95	B
9	0-105-180-40-152-58-0	96	B
10	0-126-63-181-64-49-143-36-46-0	98	B
11	0-195-54-134-163-24-29-121-0	97	B
12	0-18-114-8-174-45-125-199-83-166-0	148	C
13	0-12-109-177-150-80-68-116-184-28-0	149	C
14	0-198-197-56-186-23-75-74-72-21-0	148	C
15	0-137-2-178-115-145-41-22-133-171-73-0	148	C
16	0-153-82-124-47-168-48-7-194-106-0	150	C
17	0-51-103-161-71-135-35-136-65-66-188-20-122-0	195	D
18	0-27-162-10-189-108-90-32-131-160-128-30-70-101-0	197	D
19	0-110-4-155-139-187-39-67-170-25-55-165-130-179-0	199	D
20	0-88-148-62-159-11-175-107-19-123-182-52-146-0	194	D
21	0-13-95-97-92-151-98-37-100-193-85-93-59-99-104-96-0	248	E
22	0-118-5-61-16-141-191-44-119-192-14-142-172-87-0	250	E
23	0-76-196-77-3-158-79-129-169-78-34-164-120-9-81-185-33-157-102-50-111-0	334	F
Instance H2 Solution cost = 10224.6875			
1	0-17-16-15-0	50	A
2	0- 40-1-2-0	50	A
3	0-27-28-29-0	50	A
4	0-5-4-3-0	50	A
5	0-24-23-21-0	50	A
6	0-11-12-13-0	50	A
7	0-108-109-110-111-112-113-0	100	B
8	0-6-7-8-10-0	100	B
9	0-34-32-31-30-0	100	B
10	0-35-37-38-39-0	100	B
11	0-18-19-20-22-0	100	B
12	0-151-150-149-148-188-189-190-191-192-193-153-152-0	200	C

Route number	Sequence of customers	Load	Vehicle type
13	0-119-120-121-161-201-240-239-238-237-236-196-156-116-36-0	200	C
14	0-9-51-50-49-48-47-46-45-44-43-0	200	C
15	0-118-158-159-160-200-199-198-197-157-117-0	200	C
16	0-126-127-128-129-169-168-167-166-165-164-124-125-0	200	C
17	0-64-104-144-145-185-184-183-182-181-180-179-178-177-176-175-135-136-137-138-139-140-141-142-143-103-63-0	500	D
18	0-66-107-147-187-227-228-229-230-231-232-233-234-235-195-194-154-155-115-114-33-0	500	D
19	0-52-131-130-170-171-172-212-211-210-209-208-207-206-205-204-203-202-162-163-123-122-81-0	500	D
20	0-25-65-105-106-146-186-226-225-224-223-222-221-220-219-218-217-216-215-214-213-173-174-134-133-132-53-0	500	D
21	0-26-67-68-69-70-71-72-73-74-75-76-77-78-79-80-41-42-82-83-84-85-86-87-88-89-90-91-92-93-94-95-96-97-98-99-100-101-102-62-61-60-59-58-57-56-55-54-14-0	1000	E
Instance H3 Solution cost = 16229.7109			
1	0-36-37-38-0	50	A
2	0-11-12-13-0	50	A
3	0-66-94-93-92-64-65-0	100	B
4	0-149-177-205-233-261-260-232-204-176-148-0	100	B
5	0-14-15-16-18-0	100	B
6	0-168-196-224-252-280-253-225-197-169-141-0	100	B
7	0-19-47-46-17-0	100	B
8	0-42-69-97-125-153-181-209-237-265-264-236-208-180-152-124-96-68-40-0	200	C
9	0-32-60-88-116-144-172-200-228-256-257-229-201-173-145-117-89-61-62-0	200	C
10	0-51-80-108-136-164-192-220-248-276-277-249-221-193-165-137-109-81-53-0	200	C
11	0-43-72-100-128-156-184-212-240-268-269-241-213-185-157-129-101-73-45-0	200	C
12	0-49-76-104-132-160-188-216-244-272-273-245-217-189-161-133-105-77-50-0	200	C
13	0-44-71-99-127-155-183-211-239-267-266-238-210-182-154-126-98-70-41-0	500	D
14	0-28-56-112-111-110-138-166-194-222-250-278-279-251-223-195-167-139-140-113-85-29-1-0	500	D
15	0-52-79-107-135-163-191-219-247-275-274-246-218-190-162-134-106-78-21-0	500	D
16	0-74-102-130-158-186-214-242-270-271-243-215-187-159-131-103-75-48-20-0	500	D
17	0-22-23-24-25-26-27-55-54-82-83-84-57-58-86-114-142-170-198-226-254-255-227-199-171-143-115-87-59-31-30-2-3-4-6-7-8-9-10-0	1000	E

Route number	Sequence of customers	Load	Vehicle type
18	0-39-67-95-123-151-179-207-235-263-262-234-206-178-150-122-121-120-119-147-175-203-231-259-258-230-202-174-146-118-90-91-63-35-34-33-5-0	1000	E
Instance H4 Solution cost = 17444.9218			
1	0-25-24-23-0	50	A
2	0-2-1-40-0	50	A
3	0-84-124-204-244-284-285-245-205-125-85-0	100	B
4	0-36-34-33-32-31-29-0	100	B
5	0-108-148-228-268-308-309-269-229-149-109-0	100	B
6	0-18-19-20-22-0	100	B
7	0-16-55-56-57-58-17-0	100	B
8	0-160-200-240-280-320-281-241-201-161-121-0	100	B
9	0-144-184-224-264-304-305-265-225-185-145-0	100	B
10	0-35-37-38-39-0	100	B
11	0-72-112-152-192-232-272-312-313-314-274-273-233-193-153-113-73-0	200	C
12	0-8-48-128-168-208-248-288-289-249-209-169-170-130-129-49-9-0	200	C
13	0-61-101-141-221-261-301-300-260-220-258-257-256-216-217-177-176-136-96-0	200	C
14	0-60-100-140-139-138-137-97-98-99-59-0	200	C
15	0-156-196-236-276-277-278-238-237-197-198-158-157-0	200	C
16	0-74-114-154-155-195-194-234-235-275-315-316-317-318-319-279-239-199-159-0	500	D
17	0-53-93-133-173-174-214-213-253-254-294-293-292-252-251-291-290-250-210-211-212-172-171-131-132-92-52-0	500	D
18	0-15-54-95-94-134-135-175-215-255-295-296-297-298-299-259-219-218-178-179-180-181-182-222-262-302-303-263-223-183-143-142-102-103-104-64-63-62-21-0	1000	E
19	0-26-27-28-68-67-66-65-105-106-107-147-146-186-226-266-306-307-267-227-187-188-189-190-230-270-310-311-271-231-191-151-150-110-111-71-70-69-30-0	1000	E
20	0-3-4-5-6-7-47-46-45-44-43-42-41-80-79-78-77-76-75-115-116-117-118-119-120-81-82-83-123-122-162-202-242-282-283-243-203-163-164-165-166-206-246-286-287-247-207-167-127-126-86-87-88-89-90-91-51-50-10-11-12-13-14-0	1500	F

Route number	Sequence of customers	Load	Vehicle type
Instance H5 Solution cost = 23112.7382			
1	0-204-240-276-312-348-349-313-277-241-205-0	100	B
2	0-213-249-285-321-357-356-320-284-248-212-0	100	B
3	0-209-245-281-317-353-352-316-280-244-208-0	100	B
4	0-49-48-47-46-45-9-0	100	B
5	0-193-229-265-301-337-336-300-264-228-192-0	100	B
6	0-24-60-96-131-132-133-97-61-0	100	B
7	0-85-121-120-119-83-84-0	100	B
8	0-55-92-128-164-200-236-272-308-344-345-309-273-237-201-165-129-93-57-0	200	C
9	0-44-80-116-152-188-224-260-296-332-333-297-261-225-189-153-117-81-82-0	200	C
10	0-37-73-109-145-181-217-253-289-325-360-324-288-252-216-179-144-108-72-0	200	C
11	0-54-89-125-161-197-233-269-305-341-340-304-268-232-196-160-124-88-52-0	200	C
12	0-5-41-77-113-149-185-221-257-293-329-328-292-256-220-184-148-112-76-40-4-0	200	C
13	0-56-91-127-163-199-235-271-307-343-342-306-270-234-198-162-126-90-53-0	500	D
14	0-71-107-143-142-141-140-139-138-137-136-172-173-174-210-246-282-318-354-355-319-283-247-211-175-176-177-178-214-250-286-322-358-359-323-287-251-215-180-146-182-218-254-290-326-327-291-255-219-183-147-111-110-74-75-39-38-0	1500	E
15	0-42-43-79-78-114-150-186-222-258-294-330-331-295-259-223-187-151-115-118-154-190-226-262-298-334-335-299-263-227-191-155-156-157-158-194-230-266-302-338-339-303-267-231-195-159-123-122-86-87-51-50-13-0	1500	E
16	0-3-2-1-36-35-34-33-32-31-30-29-28-27-26-25-62-63-64-65-66-67-68-69-70-106-105-104-103-102-101-100-99-98-134-135-171-207-243-279-315-351-350-314-278-242-206-170-169-168-167-203-239-275-311-347-346-310-274-238-202-166-130-94-95-59-58-23-22-21-20-19-18-17-16-15-14-12-11-10-8-7-6-0	2000	F

References

Armas, J., & Melián-Batista, B. (2015). Variable neighborhood search for a dynamic rich vehicle routing problem with time windows. *Computers and Industrial Engineering*, *85*, 120–131.

Baldacci, R., Battara, M., & Vigo, D. (2008). *The vehicle routing problem: Latest advances and new challenges, chapter Routing a heterogeneous fleet of vehicles, 11–35*. Berlin: Springer.

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

Brandao, J. (2009). A deterministic tabu search algorithm for the Ãr´nCeet size and mix vehicle routing problem. *European Journal of Operational Research*, *195*, 716–728.

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

Choi, E., & Tcha, D.-W. (2007). A column generation approach to the heterogeneous fleet vehicle routing problem. *Computers and Operations Research*, *34*, 2080–2095.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Oprations Research*, *12*, 568–581.

Coelho, V. N., Grasas, A., Ramalhinho, H., Coelho, I. M., Souza, M. J. F., & Cruz, R. C. (2016). An ILS-based algorithm to solve a large-scale real heterogeneous fleet VRP with multi-trips and docking constraints. *European Journal of Operational Research*, *250*, 367–376.

Dondo, R., & Cerda, J. (2007). A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. *European Journal of Operational Research*, *176*, 1478–1507.

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle dispatch problem. *Operations Research*, *22*, 340–344.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Science*, *8*, 156–166.

Golden, B. L., Assad, A. A., Levy, L., & Gheysens, F. G. (1984). The fleet size and mix vehicle routing problem. *Computers and Operation Research*, *11*, 49–66.

Hansen, P., Mladenović, N., & Moreno, Âă P J. (2010). Variable neighbourhood search: Methods and applications. *Annals of Operations Research*, *175*, 367–407.

Koç, Ç., Bektaş, T., Jabali, O., & Laporte, G. (2016). Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, *249*, 1–21.

Imran, A., Salhi, S., & Wassan, N. A. (2009). A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, *197*, 509–518.

Lee, Y., Kim, J., Kang, K., & Kim, K. (2008). A heuristic for vehicle fleet mix problem using tabu search and set partitioning. *Journal of the Operational Research Society*, *59*, 833–841.

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

Li, X., Tian, P., & Aneja, Y. (2010). An adaptive memory programming metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *Part E: Logistics and Transportation Review*, *46*, 1111–1127.

Lin, S. (1965). Computers solutions of the traveling salesman problem. *Bell System Technical Journal*, *44*, 2245–2269.

Liu, J., Smith, A. E., & Qian, D. (2016). The vehicle loading problem with a heterogenous transport fleet. *Computers and Industrial Engineering*, *97*, 137–145.

Liu, S. (2013). A hybrid population heuristic for the heterogeneous vehicle routing problems. *Transportation Research Part E*, *54*, 67–78.

Melian, B., & Mladenović, N. (2007). Applications of variable neighborhood search. *IMA Journal of Management Mathematics*, *18*, 99–221.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, *24*, 1097–100.

Penna, P., Subramanian, A., & Ochi, L. (2013). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, *19*, 201–232.

Prins, C. (2002). Efficient heuristics for the heterogeneous fleet multitrip vrp with application to a large-scale real case. *Journal of Mathematical Modelling and Algorithms*, *1*, 135–150.

Prins, C. (2009). Engineering applications of artiïňĄcial intelligence. *Two Memetic Algorithms for Heterogeneous Fleet Vehicle Routing Problems*, *22*, 916–928.

Rahmani, D., & Ramezanian, R. (2016). A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions. *Computers and Industrial Engineering*, *98*, 360–372.

Salhi, S., & Sari, M. (1997). A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, *103*, 95–112.

Soylu, B. (2015). A general variable neighborhood seaerch heuristic for multiple traveling salesman problem. *Computers and Industrial Engineering*, *90*, 390–401.

Subramanian, A., Vaz Penna, P., Uchoa, E., & Ochi., L. (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, *221*, 285–295.

Subramanian, A., Penna, P.H.V., Uchoa, E., & Ochi, L.S. (2012). A hybrid algorithm for the fleet size and mix vehicle routing problem. In *International conference on industrial engineering and systems management*.

Taillard, E. D. (1999). A heuristic column generation method for heterogeneous fleet VRP. *RAIRO. Rech. Opér.*, *33*, 1–14.

Tarantilis, C., Kiranoudis, C., & Vassiliadis, V. (2003). A list based threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *Journal of the Operational Research Society*, *54*, 65–71.

Tarantilis, C., Kiranoudis, C., & Vassiliadis, V. (2004). A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *European Journal of Operational Research*, *152*, 148–158.

Tarantilis, C. D., & Kiranoudis, C. T. (2007). A flexible adaptive memory-based algorithm for reallife transportation operations: Two case studies from diary and construction company. *European Journal of Operational Research*, *179*, 806–822.

Teodorovic, D., Krcmarnozic, E., & Pavkovic, G. (1995). The mixed fleet stochastic vehicle routing problem. *Transportation Planning and Technology*, *19*, 31–34.

Yaman, H. (2006). Formulations and valid inequalities for the heterogeneous vehicle routing problem. *Mathematical Programming*, *106*, 365–390.