The Fleet Size and Mix Vehicle Routing Problem with Backhauls: Formulation and Set Partitioning-based Heuristics

Said Salhi, Niaz Wassan and Mutaz Hajarat

Kent Business School, Centre for Logistics & Heuristic Optimisation (CLHO)
University of Kent, Canterbury, UK.
Email: {S.Salhi, N.A.Wassan, M.O.A.Hajarat}@kent.ac.uk

ABSTRACT

In this paper we present a new variant of the classical Vehicle Routing Problem – the Fleet Size and Mix Vehicle Routing Problem with Backhauls (FSMVRPB). An ILP formulation of the FSMVRPB is presented. Optimal solutions for small size instances are produced and upper and lower bounds are generated for larger ones. In this paper we also propose a Set Partitioning Problem (SPP) based heuristic. Three frameworks are developed and tested on a set of new FSMVRPB data instances which we generated. Computational results are presented which can be used for future benchmarking.

Keywords: Mixed Fleet Backhauling, ILP formulation, Set Partitioning Problem, Heuristics.

1. Introduction and Problem Definition

We introduce a new variant of the classical vehicle routing problem (VRP) called the Fleet Size and Mix Vehicle Routing Problem with Backhauls (FSMVRPB). In the VRP we are given a set of customers with known demands (delivery only), a fleet of homogeneous vehicles available (unlimited in numbers) at a depot. The problem is to find the minimum cost routes originating and terminating at the depot while satisfying customer demand subject to the vehicle capacity and the route length restrictions. The FSMVRPB variant combines the aspects of two previously developed versions of the routing problems, i.e., the Fleet Size and Mix Vehicle Routing Problem (FSMVRP) (Golden et al. 1984) and the Vehicle Routing Problem with Backhauls (VRPB) (Golden et al. 1985). The VRPB was then further developed by Toth and Vigo (1996) who also generated the VRPB data sets for benchmarking purposes. The FSMVRP is different from the VRP in one aspect, i.e., it considers a heterogeneous vehicle fleet (with different fixed and running unit costs) instead of homogeneous vehicles. The VRPB is also different from the VRP mainly in one aspect, i.e., the fleet used can consider pick-ups after the deliveries are

made. In our version of the VRPB all the deliveries are made before any pick-ups on a route; no route is allowed with only pick-up (back-haul) customers; there may be restrictions on the number of vehicles available that must be utilized (Toth and Vigo, 1997; Osman and Wassan, 2002 and Wassan, 2007). In the literature, the VRPB model is further extended to some other variants such as the Vehicle Routing Problem with Mixed Deliveries and Pick-ups (VRPMD) (Deif and Bodin, 1984; Salhi and Nagy, 1999 and Nagy and Salhi, 2005) and the Vehicle Routing Problem with Simultaneous Deliveries and Pick-ups (VRPSPD) (Min, 1989 and Nagy and Salhi, 2005). In the VRPMD, deliveries and pick-ups can be made in any order of the customers on a route, whereas in the VRPSPD the deliveries and pick-ups demands can come from the same customer. These extended models of the VRPB are not being considered here, hence we refer the interested reader to Wassan et al. (2008a) and Wassan et al. (2008b).

The aspects of the FSMVRPB are defined as follows:

- The customers are divided into two groups, i.e., delivery (linehaul) and pick-up (backhaul) customers.
- A heterogeneous fleet of vehicles (each type available in unlimited numbers) with different fixed costs according to their sizes.
- All the deliveries are made to the linehaul customers before any of the backhaul customers are serviced.
- No route is allowed to contain backhaul customers only. However, a route with only linehaul customers is allowed.
- Vehicle capacity constraints are employed. However, the constraints on the *route length* and on the *fixed number of vehicles that must be or could be utilized* are not considered in this case as considered in Toth and Vigo (1997).

The problem is to find a minimum cost mixed fleet set of routes while satisfying the customer demand and problem constraints.

The FSMVRP and the VRPB have been extensively investigated separately but not in a combined way. The FSMVRPB is a more realistic routing and distribution problem with a wide applicability for those logistic companies that wish to determine the composition of their vehicle fleet as well as operating their delivery routes efficiently so to achieve competitive advantage. Moreover, companies could enhance their green strategy by exploring the various facets of green logistic including fleet management, reverse logistic and green technology among others. For a more general and informative view on this hot and interesting topic, see the recent special issue in Transportation Research Part E by Sheu and Talley (2011). In this regard, this study deals with the fleet management issue including reverse logistic.

The paper is organised as follows: In the next section we present a brief literature review on the subject followed by the mathematical formulation in Section 3. Explanation of our solution method namely the Set Partitioning based heuristic is provided in Section 4. Details of the new data sets and computational results are given in Section 5. The conclusion and suggestions are provided in the last section.

2. Literature review

As mentioned earlier, this is one of the first studies of the FSMVRPB that deals with the problem formally; hence there are no published papers on this problem except a recent PhD thesis by Hajarat (2010). The only published study closest to ours is the interesting and practical paper by Tütüncü (2010). The author tackles the heterogeneous *fixed fleet* VRP with backhauls where the vehicle routing problem with heterogeneous fleet (VRPHE) model of Taillard (1999) is extended by reducing the number of vehicles per type and incorporating backhauls. A useful decision support system is developed for this purpose. Here we review the literature linked to both the FSMVRP and the VRPB as these two when combined make up this new combinatorial problem.

There are three versions of the FSMVRP that exist in the literature. These versions differ from each other based on whether or not the variable running cost per vehicle is constant and whether the number of available vehicles for each type is known or not. Here, we shall try to differentiate them while reviewing the algorithms and techniques proposed to solve them.

The first and the original version comes from Golden et al. (1984), where the unit variable costs are set to unity for all vehicles regardless of their types and the number of vehicles in each type is also unlimited. They adopted the savings technique of Clarke and Wright (1964) to develop algorithms and also provided two implementations of the giant tour based algorithm to solve the FSMVRP. Some of the recent studies for this version include Brandao (2009) who developed a deterministic tabu search scheme; Imran et al. (2009) who designed a variable neighbourhood search scheme that is combined with the multi-level approach of Salhi and Sari (1997); Liu et al. (2009) designed and implemented a genetic algorithm-based heuristic, and Subramanian et al. (2012) developed a hybrid approach where an iterated local search is used within the Set Partitioning model. These recent approaches produced very competitive results. In this paper the focus is kept on the original version of Golden et al. (1984).

The second version originates from Salhi et al. (1992) where the number of vehicles of each type is still unlimited but the variable cost is dependent on the vehicle type. The authors modified the savings based algorithm of Golden et al. (1984) to solve this variant. A recent study in this particular area focussing on multi-depots can be found in Salhi et al. (2013).

The third version, termed as VRPHE, comes from Taillard (1999). Here, the variable cost depends on the type of vehicle but there are restrictions on the number of vehicles available for each type. This is solved using a Column Generation Method. Recent studies for this version include Tarantilis et al. (2004) who implemented a threshold accepting metaheuristic; Li et al. (2007) developed a record-to-record travel algorithm; Li et al. (2010) integrated a multi-start adaptive memory programming and a path re-linking with tabu search, and Brandao (2011) produced an efficient implementation of a tabu search. These methodologies proved very successful in producing high quality results. For an overview of various approaches for the mix fleet VRPs, see Baldacci et al. (2009).

The VRPB has been studied since the 1980s by Golden et al. (1985). Exact methods based on ILP formulations are put forward by Toth and Vigo (1997) and Mingozzi et al. (1999) where optimal solutions for instances with up to 100 customers are obtained. Many classical heuristics and meta-heuristics have also been proposed. Some of the recent references include Brandao (2006) who developed a tabu search algorithm, and Wassan (2007) who developed a reactive tabu search enhanced by adaptive memory programming. These methods produced good quality results.

3. Mathematical formulation

The following mathematical formulation for the FSMVRPB is extended from the one of Lee et al. (2008) which is initially given for the FSMVRP by incorporating the presence of backhauls. Another way would be to start from a formulation of the VRPB and then introduce the possibility of heterogeneous types of vehicles. We opted for the former as it is relatively easier and much simpler to modify.

The following notations are used throughout.

L= number of linehaul customers, indexed $1,\ldots, L$ (depot is indexed by 0)

B= number of backhaul customers, indexed $L+1 ,\ldots, n$

n = total number of customers ($L + B$)

K = number of vehicle types ($v = 1,\ldots, K$)

Q_v = capacity of a vehicle of type v, ($Q_1 < Q_2 < Q_3 \ldots < Q_K$)

f_v = fixed cost of a vehicle of type v, ($f_1 < f_2 < f_3 \ldots < f_K$)

d_i = demand of customer i ($i = 1,..., n$)

c_{ij} = travelling cost (distance) between node i and node j

ψ_{ij} = the vehicle load on the arc from customer i to customer j

$x_{ij}^v \begin{cases} 1, \text{ if arc } (i, j) \text{ is traversed by vehicle type } v \\ 0, \text{ otherwise} \end{cases}$

Minimise

$$\sum_{v=1}^{K} f_v \sum_{j=1}^{L} x_{0j}^v + \sum_{v=1}^{K} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^v \qquad (1)$$

Subject to:

$$\sum_{i=0}^{n} \sum_{v=1}^{K} x_{ij}^v = 1 \qquad j = 1 ,\ldots, n \qquad (2)$$

$$\sum_{j=0}^{n} \sum_{v=1}^{K} x_{ij}^v = 1 \qquad i = 1 ,\ldots, n \qquad (3)$$

$$\sum_{i=0}^{n} x_{ip}^v = \sum_{j=0}^{n} x_{pj}^v \qquad v = 1 ,\ldots, K, \quad p=0 ,..., n \qquad (4)$$

$$\sum_{i=0}^{L} \psi_{ij} = \sum_{l=0}^{n} \psi_{jl} + d_j \qquad j = 1 ,\ldots, L \qquad (5)$$

$$\sum_{l=L+1}^{n} \psi_{jl} + \psi_{j0} = d_j + \sum_{i=1}^{n} \psi_{ij} \qquad j = L+1 ,\ldots, n \qquad (6)$$

$$\psi_{ij} = 0 \qquad i = 0 ,\ldots, L , j=0 \ \& j = L+1 ,\ldots, n \qquad (7)$$

$$\psi_{ii} = 0 \qquad i = 0 ,\ldots, n \qquad (8)$$

$$\sum_{i=L+1}^{n} \psi_{i0} = \sum_{i=L+1}^{n} d_i \qquad (9)$$

$$\sum_{j=1}^{L} \psi_{0j} = \sum_{j=1}^{L} d_j \qquad (10)$$

$$x_{0j}^v = 0 \qquad j = L+1 ,\ldots, n, \ v = 1 ,\ldots, K \qquad (11)$$

$$x_{ij}^{v} = 0 \qquad i = L+1 ,\ldots \text{n}, j=1,...,L, \ v = 1 ,\ldots, K \qquad (12)$$

$$\psi_{ij} \leq \sum_{v=1}^{K} x_{ij}^{v} Q_v \qquad i \neq j = 0, 1 ,\ldots, n \qquad (13)$$

$$x_{ij}^{v} \in \{0,1\} \qquad i = 0,. ..,n, \ j = 0 ,\ldots, n, v = 1 ,..., K \qquad (14)$$

$$\psi_{ij} \geq 0 \qquad i = 0 ,..., n, \ j = 0 ,\ldots, n \qquad (15)$$

In the above formulation, Equation (1) is the objective function which refers to minimising the total cost. The first part of this equation relates to the fixed cost of using different types of vehicles and the second part relates to the travelling cost. Constraints (2) and (3) ensure that each customer is visited exactly once and by one vehicle only. Constraints in (4) guarantee that each route is continuous and served by one type of vehicle. Note that (4) combined with (2) and (3) guarantee that there will be only one vehicle serving a customer. Constraints in (5) ensure the precedence relationship and are used to make sure that the demand of linehaul customers is satisfied by the delivery, while constraints in (6) are also precedence constraints which ensure that the backhaul pickups are satisfied. Constraints in (7) confirm that the load of all arcs from deliveries to any backhaul or to the depot is zero (vehicles are empty on those arcs). Constraints (8) guarantee that there is no load on arc (*i,i)*, whilst constraints in (9) impose that the total loads on all vehicles returning from the backhaul customers to the depot is exactly the sum of the demand of the backhaul customers. Similarly constraints in (10) ensure that the total load of all vehicles leaving the depot is exactly the sum of linehaul customers' demand. Constraints in (11) prevent any route starting from the depot to any backhaul customer and those in (12) guarantee that no vehicle goes from a backhaul to a linehaul. Constraints in (13) restrict the vehicle load from customer *i* to *j* from exceeding the capacity of the vehicle using the arc (*i,j)*. Constraints (14) and (15) define the binary and the continuous decision variables x_{ij}^{v} and ψ_{ij} respectively.

The FSMVRPB can be shown to be NP-hard as it can be reduced to two known NP hard problems namely the classical capacitated VRP by setting *K = 1* and *B = 0 (i.e., L=n),* and the VRPB by setting *K = 1*. In order to solve this combinatorial problem, we first provide optimal solutions for those instances with *n* = 20 and upper and lower bounds for the larger problems by setting a CPU time restriction of 2 hours on CPLEX. In the next section, we propose an efficient implementation of a hybrid heuristic that is based on set covering and set partitioning models to solve the FSMVRPB.

The above formulation requires $K(n+1)^2$ binary decision variables including $(KL(n-L-1)+Kn)$ variables set to *0*, $(n+1)^2$ continuous decision variables including $(n(L+2)+L^2)$ variables set to *0*, and $(n^2+n(K+5)+K+3)$ constraints. This renders the problem intractable even for powerful optimisers such as CPLEX as will be shown in the computational results section.

Given that $\psi_{ij} \geq 0$ and $x^v_{0j} \geq 0$ it can be noted that (7) can be replaced $\sum_{i=0}^{L} \psi_{ij} = 0$ for *j=0 & j = L+1* ,…, *n* and (11) by $\sum_{j=L+1}^{n} x^v_{0j} = 0$ for all *v* = 1 ,…, *K* . However, these new constrainsts will add extra complexity to the formulation as setting these variables to zero, as defined in (7) and (11), is relatively more efficient than letting the solver to determine their corresponding optimal values.

Some tightening of the formulation

It can be observed that for the case where backhauls are present in the formulation such as the classical VRPB and the FSMVRPB studied here, possible tightening of those constraints that distinguish between linehauls and backhauls may be worth investigating. We achieve this by not including the parts of the constraints where the decision variables will always have a zero value. These cases happen in constraints (2), (3) and (4). In addition, we distinguish between linehauls and backhauls when restricting the load on a given arc by redefining (13) into 4 distinct sets of constraints (13a-13d).

(i) Constraints set (2) can be replaced by the following two sets of constraints where the first one deals with linehauls and the second with backhauls:

$$\sum_{i=0}^{L}\sum_{v=1}^{K} x^v_{ij} = 1 \qquad j = 1,...,L \qquad (2a)$$

$$\sum_{i=1}^{n}\sum_{v=1}^{K} x^v_{ij} = 1 \qquad j = L+1,...,n \qquad (2b)$$

(ii) Constraints set (3) can also be replaced similarly with two sets:

$$\sum_{j=0}^{n}\sum_{v=1}^{K} x^v_{ij} = 1 \qquad i = 1,...,L \qquad (3a)$$

$$\sum_{j=L+1}^{n}\sum_{v=1}^{K} x^v_{ij} + \sum_{v=1}^{K} x^v_{i0} = 1 \qquad i = L+1,...,n \qquad (3b)$$

(iii) Constraints set (4) can also be replaced by three sets of constraints with (4a) and (4b) as in (i) and (ii) and the new (4c) relating to the depot.

$$\sum_{i=0}^{L} x_{ip}^{v} = \sum_{j=0}^{n} x_{pj}^{v} \qquad p = 1,...,L \quad v = 1,...,K \qquad (4a)$$

$$\sum_{i=1}^{n} x_{ip}^{v} = \sum_{j=L+1}^{n} x_{pj}^{v} + x_{p0}^{v} \qquad p = L+1,...,n \quad v = 1,...,K \qquad (4b)$$

$$\sum_{i=1}^{L} x_{0i}^{v} = \sum_{j=0}^{n} x_{j0}^{v} \qquad v = 1,...,K \qquad (4c)$$

(iv) Constraints set (13) can be replaced by the following 4 sets of constraints (13a)-(13d).

$$\psi_{0j} \le \sum_{v=1}^{K} x_{0j}^{v} Q_v \quad j = 1,...,L \qquad (13a)$$

$$\psi_{j0} \le \sum_{v=1}^{K} x_{j0}^{v} Q_v \quad j = L+1,...,n \qquad (13b)$$

$$\psi_{ij} \le \sum_{v=1}^{K} x_{0j}^{v} (Q_v - d_i) \qquad i, j = 1,...,L \qquad (13c)$$

$$\psi_{ij} \le \sum_{v=1}^{K} x_{ij}^{v} Q_v \qquad i, j = L+1,...,n \qquad (13d)$$

This reformulation has the same number of constraints as the previous one except that the new ones are less dense with a less number of variables involved in each constraint.

Other modelling considerations

In this study, we consider that no route is made up of backhaul customers only. This is a restriction usually imposed in the literature but it can be relaxed easily in our model by just ignoring constraints (11). Also, if tightening is adopted the followings constraints need to be modified. In constraints (4b) $\sum_{i=1}^{n} x_{ip}^{v}$ needs to be replaced by $\sum_{i=0}^{n} x_{ip}^{v}$ and in constraints (4c) $\sum_{i=1}^{L} x_{0j}^{v}$ by $\sum_{i=1}^{n} x_{0j}^{v}$. Note that this is equivalent

to using the original constraints (4). In constraints (13a), $\psi_{0j} \leq \sum_{v=1}^{K} x_{0j}^{v} Q_v \;\; j = 1,...,n$ needs to be true for all customers not just linehauls as originally stated.

In this study, we also assume that the number of vehicles available for each type is unlimited and can be computed as a by-product using $\sum_{j=1}^{L} x_{0j}^{v}$ for each type *v*. In case, the number of vehicles available for each type is predefined, say $K_{v,}$ the following constraints $\sum_{j=1}^{L} x_{0j}^{v} \leq K_v$ need to be added to the above formulation. Also, if the overall number of vehicles is known, say NV, the following constraint $\sum_{v=1}^{K}\sum_{j=1}^{L} x_{0j}^{v} \leq NV$ will be required. In the special case where there is one type of vehicle only, the problem reduces to the VRP. This is obtained by setting $K = 1$ and using a two index variable instead of three by replacing x_{ij}^{v} with x_{ij} and removing all the summations over v.

4. Set Partitioning-based methodology

The process of our set partitioning based methodology involves the generation of a large set of linehaul giant tours, the construction of a directed *Cost Network* and its optimal partitioning, and the insertion of backhauls to create a pool of mixed fleet vehicle routes. For bigger size instances with a higher proportion of linehauls the pool becomes too large to be handled by the Set Partitioning model using CPLEX. Hence we have devised three frameworks to identify subsets of routes that we consider to be promising. These are explained in the following subsections.

4.1 Construction of the linehaul giant tours

Linehaul giant tours are generated using two widely used methods, i.e., the Sweep procedure of Gillett and Miller (1974) and the Nearest Neighbourhood (NN) method. The Sweep scheme starts by choosing the node closest to the *y*-axis. It then adds the remaining nodes by selecting them in a clock-wise manner until a tour that contains all linehauls is completed. The NN scheme starts by choosing a node closest to the depot, it then adds the next node closest to the last added node, and the process continues until all the linehaul nodes are added to the giant tour.

4.2 Construction of the linehaul cost network and its optimal partitioning

For each of the giant tours, a directed *cost network* similar to the one developed in Imran et al. (2009) is constructed. This was initially presented by Beasley (1983) for the VRP and Golden et al. (1984) for the FSMVRP.

The procedure of the construction of our cost network and its optimal partitioning is briefly given here for completeness. Consider a giant tour defined by the following sequence $\{0, i_1, i_2, ..., i_n, i_{n+1}\}$ where '0' and $'n+1'$ denote the depot and i_k refers to the k^{th} customer on the sequence. Let G (V, E) be a directed network with a vertex set V=$\{0, i_1, i_2, ..., i_n, i_{n+1}\}$ and an edge set $E = \{(i_r, i_s)$ with $r < s;\ r = 0,1,..., n\}$ with the cost of the arc (i_r, i_s) denoted by

$$C_{i_r, i_s} = \begin{cases} f_v + c_{0i_r} + \sum_{j=r}^{s-1} c_{i_j i_{j+1}} + c_{i_s 0} & \text{if } \sum_{j=r}^{s} d_{i_j} \le Q_K \\ \infty & \text{otherwise} \end{cases}$$

where f_v represents the fixed cost of the smallest vehicle (say *v*) that accommodates the total demand on the arc (i_r, i_s) (i.e., $Q_{v-1} \le \sum_{j=r}^{s} d_{i_j} \le Q_v$), and Q_K is the capacity of the largest vehicle.

The optimal fleet configuration is obtained by partitioning this directed cost network using Dijkstra's algorithm though dynamic programming can also be used. Note that the Dijkstra's partitioning-based approach has been used widely, see for instance Renaud et al. (1996) and Salhi and Sari (1997).

The selected routes from the optimal partitioning of each of the cost networks are then scrutinised by removing the duplicate routes. The remaining routes are stored into a pool which we call the *Route Reference Set*. The routes in this pool are then improved by using the local search procedure *2-Opt* method (Lin, 1965). Our implementation of the *2-Opt* procedure removes two non-adjacent arcs on a route, the partial tour between the two removed arcs is reversed and two new arcs are then added to connect to the reversed partial tour. This procedure could be extended to *3-Opt* where three arcs are involved instead, see Christofides and Eilon (1969). However, given the massive number of routes that need to be checked, this extra refinement can be too costly in terms of computation time. A Set Partitioning Problem (SPP) is solved based on the routes from the *Route Reference Set.* Our SPP model is described as follows.

Let M = the number of routes (the cardinality of the Route Reference Set)

$C_i = \text{Cost of route } i;\ i = 1,...,M$

$$a_{ij} = \begin{cases} 1 & \text{if customer } j \text{ belongs to route } i \\ 0 & \text{Otherwise} \end{cases}$$

$$x_i = \begin{cases} 1 & \text{if route } i \text{ is selected} \\ 0 & \text{Otherwise} \end{cases}$$

$$\text{Minimise} \quad \sum_{i=1}^{M} C_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{M} a_{ij} x_i = 1 \ ;\ j = 1,...,n \qquad (16)$$

$$x_i \in \{0,1\};\ i = 1,...,M$$

4.3 Insertion of backhauls

The backhaul customers are inserted into the linehaul solution obtained by the SPP one at a time by adopting the Sweep method. By selecting the linehaul routes systematically, the backhaul customers are swept clock-wise and added at the end of each route by deleting the arc which connects the last linehaul customer to the depot. The backhaul additions are performed while satisfying the capacity constraints of the vehicles already used in the linehaul routes. A FSMVRPB solution is obtained once all the backhauls are inserted. The *2-Opt* procedure is again used as the post-optimizer to seek further improvement on the FSMVRPB solution. Note that we discard those *2-Opt* moves which could break the precedence constraint of serving linehauls before backhauls on a route.

***Framework-1*: *A* simple approach**

The procedure of finding the optimal linehaul routes from the *Route Reference Set* and then adding backhauls using the Sweep method to obtain a FSMVRPB solution is termed as *Framework-1*. The flowchart of this framework is given in Figure 1 where LH routes refer to the routes made up of linehaul customers only, whereas LH/BH routes represent those routes that contain both linehaul and backhaul customer

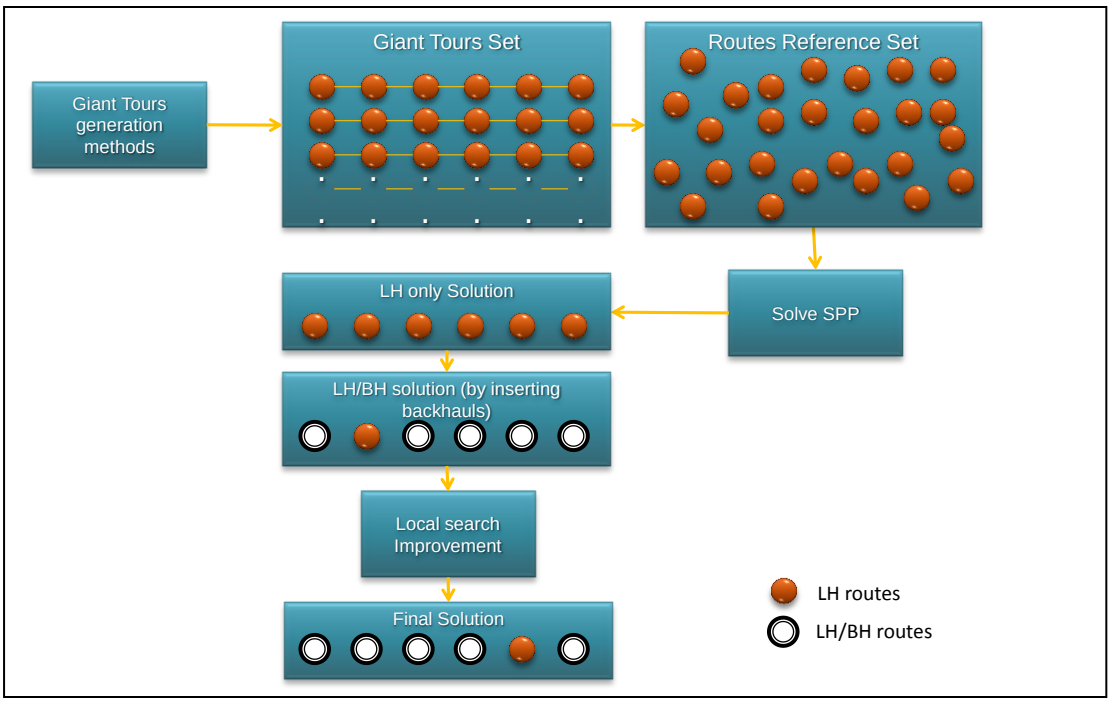

Figure 1: *Framework- 1*

***Framework-2*: The use of the SPP on LH and LH/BH pool of routes**

This framework is different from the previous one in the sense that it solves the Set Partitioning model after the backhaul customers are added to the linehaul routes in the *Route Reference Set* using the Sweep method. Here the backhauls are added one by one, i.e. each time a backhaul is added to the linehaul route a new combined linehaul/backhaul route is formed as long as the capacity of the vehicle is not violated. For example, three possible routes are generated from a linehaul route as shown in Figure 2.

As we also use anti-clock-wise Sweep to add the backhauls, several routes are generated from each linehaul route. Hence the size of the pool becomes very large as both the LH and the LH/BH routes constitute our new *Route Reference Set*. The flowchart of Framework-2 is given in Figure 3.

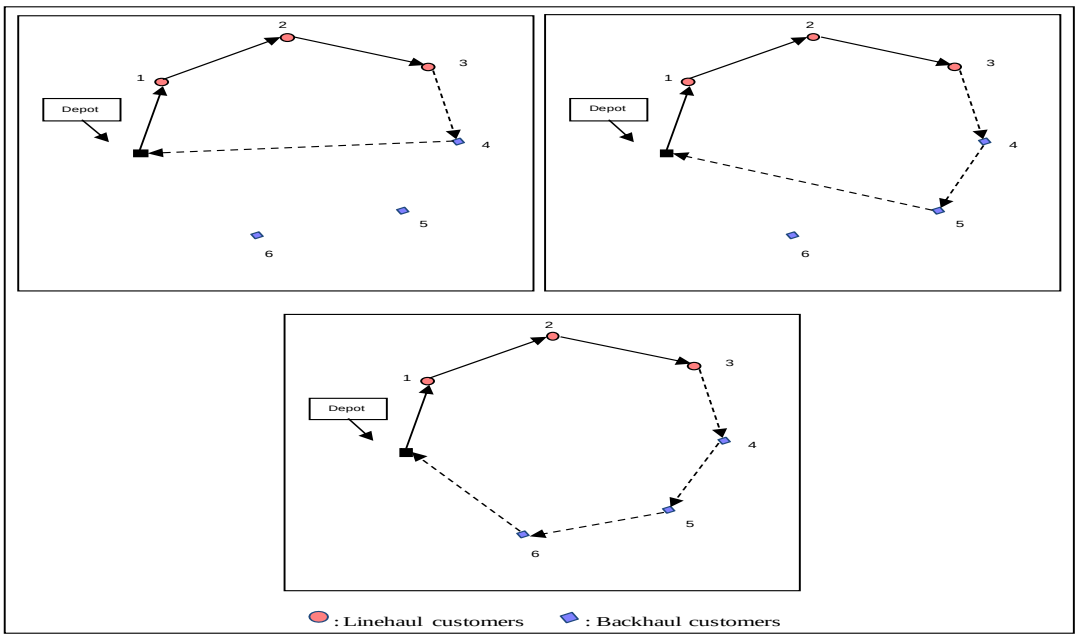

Figure 2: An example of adding backhauls one by one to generate combined linehaul/backhaul routes using clock-wise Sweep.

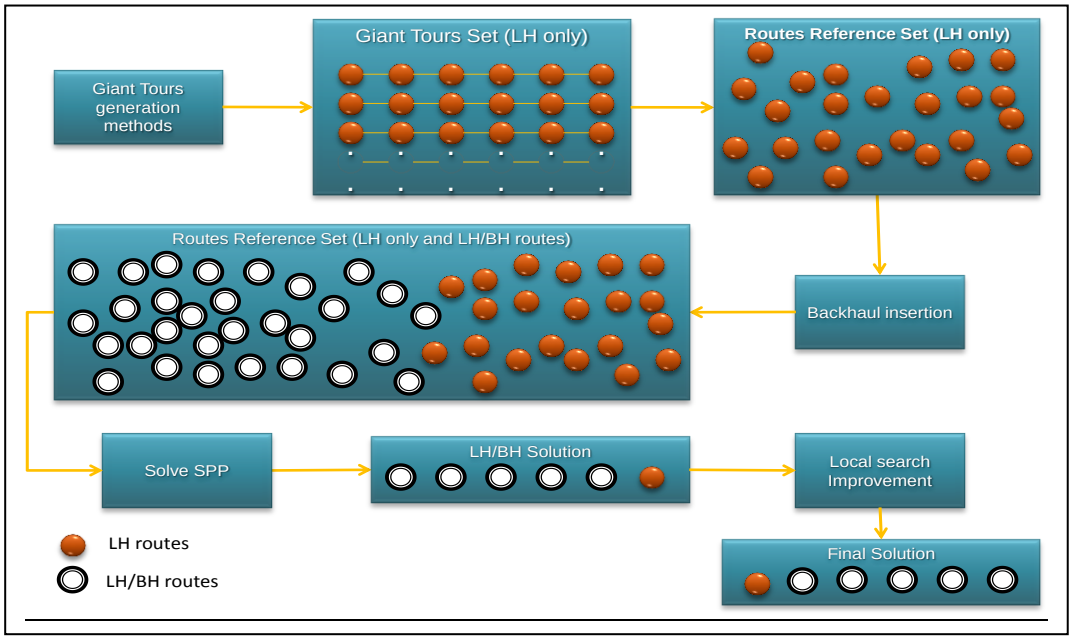

Figure 3: *Framework- 2*

***Framework-3*: A reduction scheme on the number of linehaul routes**

Framework-2 uses a huge number of routes in the pool and as a result some instances could not be solved due to an excess of memory when using CPLEX. Here, a different approach is tested in *Framework-3* where we aim to reduce the large number routes in the pool. In other words, instead of using all the linehaul routes and adding backhauls one by one to generate a huge number of routes as in previous frameworks, here a smaller elite set of LH routes is obtained, see Figure 4. This is achieved by firstly solving the Set Partitioning problem (SPP) once to obtain the best linehaul solution containing an optimal set of routes. These routes are then removed from the set of LH routes and stored into the pool which we refer to as *Routes Reference Set* (Elite LH only). More LH routes are then removed and added to this pool by solving repeatedly the Set Covering Problem (SCP) using the remaining routes in the set. The size of this elite pool is set to $2y^2$, where y denotes the number of routes in the solution obtained by the SPP. This setting was empirically found to be appropriate.

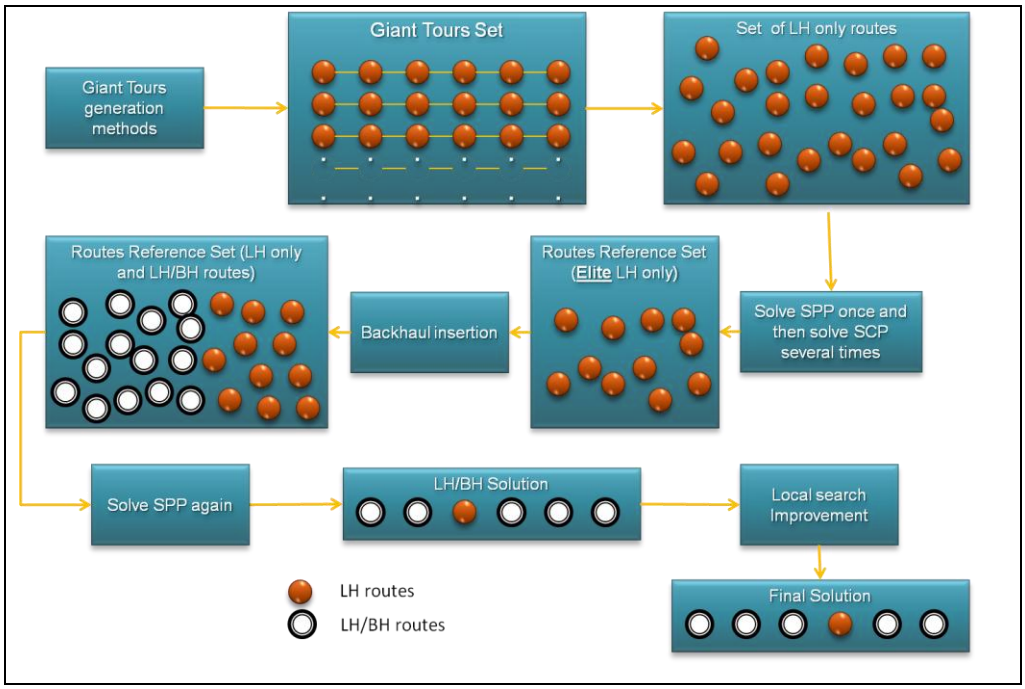

Figure 4: *Framework-3*

Note that the SCP solutions at this stage of the search may be infeasible as the model allows a customer to be on more than one route. Nevertheless our purpose is to choose a pool of promising routes that are added to the *Routes Reference Set*. The reason behind solving SPP once is that there will always be at least one feasible solution in the pool. The backhaul customers are then added to the elite *Routes*

Reference Set pool of routes in the same manner as in *Framework-2*. The final best solution is obtained by solving the SPP on this LH and LH/BH pool. The flowchart and the steps of *Framework-3* are given in Figures 4 and 5, respectively. The SCP model is the same as the SPP except that Equation (16) is replaced with the following Equation (17).

$$\sum_{i=1}^{M} a_{ij} x_i \geq 1 \qquad \forall j = 1, ..., n \qquad (17)$$

Step 1: Generate giant tours for linehaul customers.

Step 2: For each giant tour, construct the corresponding directed *Cost Network* and solve the shortest path problem to find the optimal set of LH feasible mix fleet routes.

Step 3: Obtain the best linehaul solution containing the optimal set of routes by solving the SPP, and set y = the number of routes found in the SPP solution.

Step 4: Remove the SPP solution routes from the set of LH routes and store them into the elite pool *Routes Reference Set*.

Step 5: Remove more promising routes from the remaining routes in the set of LH routes and add them to the elite pool by solving the Set Covering Problem repeatedly until $2y^2$ routes are pooled.

Step 6: Add the backhaul customers to the linehaul routes in the elite pool as described in *Framework-2*.

Step 7: Use SPP again to obtain the best final FSMVRPB solution.

***Step 8*:** Apply the 2-opt local search to seek further improvement.

Figure 5: The main steps of *Framework-3*

In Step 1, we used both the Nearest Neighbour and the Sweep heuristics to generate the giant tours. In Step 2, the shortest path is solved using Dijkstra's algorithm (note that dynamic programming can also be used efficiently here as the network is directed).

Additional Experiments/Fine-Tuning

In this study, the main parameter used is $2y^2$ that determines the size of the pool in *Framework-3*. This was found empirically to be appropriate after conducting several experiments using various sizes. We, however, performed some fine-tuning experiments on *Framework-2* to check if any further improvement was possible on the existing procedures. Note that the volume of routes produced in our heuristic is very large. On one hand, this provides an opportunity for a better quality solution but on the other hand, it creates computational issues, especially when it comes to using CPLEX as part of the methodology.

Hence, attempts to find a balanced compromise between the solution quality and the computational burden was explored. We report two such efforts.

Case 1: In this case, some of the giant tours which we considered not to be promising are excluded from the search. We identified such giant tours on the basis of excessively long arcs that form longer arcs routes which may have a small chance of being chosen at a later stage. In brief, any giant tour that has an arc larger than $\mu + \theta\sigma$ is removed where μ and σ refer to the overall average of the giant tours arc length averages and their standard deviations respectively, and $\theta \in [-2, 2]$ is a parameter emphasizing the percentage of tours removed. Using basic statistics, for instance if $\theta = 0$ there will be approximately 50% of tours removed, $\theta = 1$ will lead to approximately 15% of tours to be removed whereas $\theta = 2$ will allow about 2.5% of tours to be removed only. If $\theta < 0$ more than half of the number of giant tours will be removed. Several experiments were conducted with different pool sizes (in percentage) determined by the values of θ to achieve a better compromise between the solution quality and the computer time/storage used. The results of two such experiments are presented here. As compared to *Framework-2,* a saving of around 34% in CPU time was achieved with 50% reduction in the number of giant tours (i.e., $\theta = 0$), however the solution quality reduced by 1.24% on average. A 15% reduced pool of giant tours (i.e., $\theta = 1$) achieved around 13% reduction in the CPU time while producing 0.9% inferior solution on average. Overall, this analysis showed that when comparing *Case 1* with *Framework-3*, even with a reduction of 50% in the number of giant tours, the former still spends twice as much time while yielding inferior solutions on average.

Case 2: In this case, the size of the LH/BH pool was reduced by randomly choosing a subset of routes only. We tested different randomly chosen pool sizes and found significant compromises on solution quality. For instance, reducing the pool size by 30% produced an average solution that was around 6% worse on average as compared to the solution produced by *Framework-2*. The results are obviously even worse when compared to our best framework *Framework-3.*

5. Computational Experiments

We first provide the details of our computer specifications and the new data sets which we constructed for this variant. We then provide optimal solutions for the smaller instances (n=20) together with upper and lower bounds for large instances followed by detailed heuristics results and analysis.

5.1 New Data Sets, Computer specifications

Computer specifications and software used:

The optimal solutions are obtained using the IBM ILOG CPLEX 12.3 software on Intel Core 2Duo PC with a 2.80GHz processor and 8GB of RAM. The heuristic algorithms are coded in Microsoft Visual C++ 6 and run on an Intel Pentium 4 with a 3.0 GHz processor and 2.5 GB of RAM personal computer.

New data instances:

The FSMVRPB instances are generated using the mixed fleet data set of Golden et al. (1984) and the VRPB data set of Toth and Vigo (1997). In these instances, the mix fleet attributes (distances, fixed and running costs, various vehicle capacities) are taken from the first study and the backhaul percentages are from the latter. It should be noted that the original source of this data set is the VRP data initially proposed by Christofides and Elion (1969). We generate a set of 36 FSMVRPB instances using the 12 test problems of Golden et al. (1984) ranging from 20 to 100 customers. We denote them by HWS, short for Hajarat, Wassan and Salhi. For each instance, following the conventions used in Toth and Vigo (1997), three new instances are generated using linehaul/backhaul (LH/BH) percentages of 50/50, 67/33 and 80/20, by taking every first customer as a backhaul in two, three and five customers, respectively. For ease of reference we present in Table 1 the data of these 36 FSMVRPB instances. The HWS instances can be obtained from CHLO (2013).

(Insert Table 1 around here)

First we provide our optimal solution and lower/upper bounds generated by CPLEX and then present our heuristic results. In addition, we also report and compare our results against the best solution from the literature for the special case of this problem, i.e., the fleet size and mix VRP (FSMVRP) where only delivery customers are considered.

We used percentage deviation to assess the tightness of the lower and upper bounds as well as the quality of our heuristics. For each instance the deviation is computed as follows:

$Deviation(\%) = \frac{F - F_{best}}{F_{best}}.100$ where F_{best} refers to the largest LB (or smallest UB or the best of the heuristic solution values), and F is the value for the LB, UB or the heuristic solution value. The overall average deviation is computed as the average over all the instances.

5.2 Optimal solution and lower/upper bounds

We run CPLEX for a maximum execution time of *two* hours using the *breadth first* strategy (also known as *best-bound*). In Table 2, we report for each of the instances the optimal solution in bold if found and the corresponding CPU time used. We also report the lower and the upper bounds found within the two hours maximum allowed time. CPLEX has obtained 10 optimal solutions for the 12 small instances (n=20) and one for the 50 customers instance (HWS17). We conducted these experiments using both formulations namely without and with tightening. With regard to lower bounds, the modified formulation, with the new constraints (2a-2b, 3a-3b, 4a-4c, and 13a-13d), provide tighter lower bounds on average with a deviation of 0.044% worse compared to 0.403% for the first formulation. In addition, the largest deviation for the modified formulation is 0.48% only (instance HWS26) whereas the other formulation produces a 7.61% (instance HWS16). However, the first formulation appears to generate relatively better upper bounds on average with a deviation about 0.5% lower than the modified one.
One obviously can argue about the usefulness of considering these two formulations separately and running them for 2 hours each instead of considering one only but for 4 hours. Though, this was not the aim of this exercise, this can easily be tested as long as CPLEX does not run out of memory.

It is worth noting that optimality could not be guaranteed within the time limit of two hours for two of the smaller instances namely HWS10 and HWS12. We then decided for these particular small instances to let CPLEX run for a longer time to obtain the optimal solutions. According to Table 2, the average duality gap between upper and lower bounds, including optimal solutions, is around 14%. This high value is mainly due to the larger instances ($n \geq 75$). This information may not necessarily be useful as our lower bounds could be too loose or the obtained upper bound may be not that good.

(Insert Table 2 around here)

5.3 Heuristics results and analysis

Frameworks:

In Table 3 results of the three frameworks are compared with those obtained by the constructive heuristic namely the *Sweep-alpha* method developed in Hajarat (2010). In this method the LH giant tours generated by the Sweep procedure are partitioned into feasible mix fleet routes by calculating a desirability factor "*alpha*" that relates to the total distance (cost) of a route, the corresponding cost per unit demand and the relative vehicle fixed cost per unit of distance travelled. Backhauls are then added to tours using the Sweep method.

The results are compared on the basis of average solution quality and the CPU time. All frameworks performed better on the solution quality as compared to the *Sweep-alpha* approach; however the later spent relatively much less computer time. Among the frameworks, *Framework-1* operated fast but fell behind on solution quality (a deviation of about 6%) as compared to the other two frameworks. *Framework*-2 produced the best quality solution as it operates on a very large pool size (around 10 times bigger than *Framework-1*), hence it spent a significant amount of computer time and could not solve two instances due to an excess memory problem when using CPLEX. *Framework-3* on the other hand found the right balance by building an operative pool size to solve all the instances within a reasonable computing time while compromising a little on the solution quality with a deviation of about 2%. However, *Framework-3* outperformed the *Sweep-alpha* heuristic yielding a percentage improvement of over 6% while still requiring a reasonably small amount of CPU time. This heuristic also behaves rather well considering it yields an average duality gap, with respect to the lower bound, to just above 5%.

(Insert Table 3 around here)

5.4 Performance of the proposed heuristic on FSMVRPB with |B|=0

For further comparison, we tested our approach on a special class of FSMVRPB namely the standard Fleet Size and Mix VRP (FSMVRP). This is described in Section 2 as the original version (Golden *et al.* (1984)), where all customers are treated as linehauls (i.e., |B|=0). In Table 4, we report the overall best solutions obtained from several algorithms and the ones found by the proposed heuristic. Even though our methodology was originally designed to solve the FSMVRPB and not the FSMVRP, our Set Partitioning based heuristic shows to perform reasonably well given that the overall best results from the

literature were on average just over 2% better than ours. These results show that our heuristic is flexible enough to cater for similar and related problems that incorporate heterogeneous vehicle fleet.

(Insert Table 4 around here)

6. Conclusion and Suggestions

The contributions of the paper to the literature are the introduction of a new variant of the classical VRP which we call the Fleet Size and Mix Vehicle Routing Problem with Backhauls (FSMVRPB); the generation of a large number of new instances (36 in total) ranging in size from 20 to 100; a new ILP formulation of the problem together with some constraints tightening; optimal solutions for small size instances as well upper and lower bounds for larger ones; and finally an efficient design of a Set Partitioning based heuristic. The proposed heuristic contains three frameworks. The first one did not include the backhauls in the pool fleet composition which affected the solution quality but proved to be faster than the other two. The second incorporated both linehauls and backhauls in the pool fleet composition. This enhanced the quality of solutions but required a large CPU time besides being unable to solve all instances due to the huge number of routes generated. The third one restricts the number of routes in the pool efficiently making it the most appropriate in terms of both the solution quality and the computational time. The proposed heuristic proved to be a good performer as it yields solutions that are on average 5% over the obtained lower bounds only. We therefore think that the results produced for this new problem are not only of interest to practitioners but to academics for benchmarking purposes. The heuristic is also tested on a special case (the heterogeneous VRP namely the FSMVRP) and performed reasonably well.

This study will hopefully entice other researchers to explore this interesting practical routing problem even further. This could be achieved by exploring the identification of promising routes or giant tours in more depth. Another approach would be to develop efficient meta-heuristics or a hybridisation of heuristics and exact methods. Other related routing problems that could be worth examining in the future may include multi-depots, time windows, constraint on loading and unloading sequences, among others.

Acknowledgments- We would like to thank the editor and the referees for their valuable comments and suggestions that improved both the content as well as the presentation of the paper.

References

Beasley, J., 1983. Route First-Cluster Second Methods for Vehicle Routing. Omega 11**,** 403-408.

Baldacci, R., Battarra, M., Vigo, D., 2008. Routing a heterogeneous fleet of vehicles. In: Golden, B., Raghavan, S., Wasil, E. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, pp. 3–27.

Brandao, J., 2006. A new tabu search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research 173, 540-555.

Brandao, J., 2009. A deterministic tabu search algorithm for the fleet 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 & Operations Research 38, 140-151.

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

Christofides, N., and Eilon, S., 1969. An algorithm for the vehicle dispatch problem, Operational Research Quarterly 20, 309-318.

CHLO, 2013. Datasets – Routing data sets - Mix fleet VRPB. Center for Logistics and Heuristic Optimization, Kent Business School, University of Kent, UK, http://www.kent.ac.uk/kbs/research/research-centres/clho/>.

Clarke, G., Wright, J., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 42, 39-51.

Deif, I., Bodin, L., 1984. Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauls. In: Kidder AE. (Eds.), Proceedings of the Babson Conference on Software Uses in Transportation and Logistics Management, Babson Park, 75-96.

Dijkstra, E.W., 1959. A Note on Two Problems in Connection with Graphs. Numerische Mathematik 1, 269-271.

Gillett, E., Miller, L., 1974. A Heuristic Algorithm for the Vehicle-Dispatch Problem. Operations Research 22, 39-51.

Golden, B., Alfaro, J., Baker, E., Schaffer, J., 1985. The vehicle routing problem with backhauling: Two approaches", Proceedings of the 21st Annual meeting of the Southeast Institute of Management Science, (pp. 90-92).

Golden, B., Assad, A., Levy, L., Gheysens, F.G., 1984. The fleet size and mix vehicle routing problem, Computers & Operations Research 11, 49-66.

Hajarat, M., 2010. Heuristics for the fleet size and mix vehicle routing problem with backhauls. PhD dissertation, University of Kent, UK.

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

Lee, Y.H., Kim, J.I., Kang, K.H., Kim, K.H., 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.P., 2010. An adaptive memory programming meta-heuristic for the heterogeneous fixed fleet vehicle routing problem. Transportation Research Part E 46, 1111-1127.

Lima, C., Goldbarg, M., Goldbarg, E., 2004. A memetic algorithm for the heterogeneous fleet vehicle routing problem. Electronic Notes in Discrete Mathematics 18, 171-176.

Lin, S., 1965. Computer solutions of the travelling salesman problem. Bell Syst. Comp. Journal 44, 2245-2269.

Liu, S., Huang, W., Ma, H., 2009. An effective genetic algorithm for the fleet size and mix vehicle routing problem. Transportation Research Part E 45, 434-445.

Min, H., 1989. The multiple vehicle routing problem with simultaneous delivery and pickup, Transportation Research Part A 23, 377–386.

Mingozzi, A., Giorgi, S., Baldacci, R., 1999. An exact method for the vehicle routing problem with backhauls. Transportation Science 33 (3), 315-329.

Nagy, G. and Salhi, S. (2005) Heuristic Algorithms For Single And Multiple Depot Vehicle Routing Problems With Pickups And Deliveries. European Journal of Operational Research, 162(1), 126-141.

Osman, I., Wassan, N., 2002. A Reactive Tabu Search for the Vehicle Routing Problem with Back-hauls. Journal of Scheduling, 5(4), 263-285.

Subramanian, A., Penna, P.H.V., Uchao, E., Ochi, L.S., 2012. A Hybrid algorithm for the Heterogeneous Fleet Vehicle Routing Problem. European Journal of Operational Research 221, 285-295.

Renaud, J., Boctor, F.F., Laporte, G., 1996. An Improved Petal Heuristic for the Vehicle Routeing Problem. Journal of the Operational Research Society 47, 329-336.

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.

Salhi, S., Nagy, G., 1999. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. Journal of the Operational Research Society 50, 1034–1042.

Salhi, S., Sari, M., Saidi, D., Touati, N., 1992. Adaptation of Some Vehicle Fleet Mix Heuristic. OMEGA 20, 653-660.

Salhi, S., Imran, A., Wassan, N., 2013. The Multi-Depot Vehicle Routing Problem with Heterogeneous Vehicle Fleet: Formulation and a Variable Neighborhood Search Implementation. Computers and Operations Research, (To appear).

Sheu, J.B., Talley, W.K., 2011. Green Supply Chain Management: Trends, Challenges, and Solutions, Transportation Research Part E 47, 791-792.

Taillard, E., 1999. A heuristic column generation method for the heterogeneous fleet vehicle routing problem. RAIRO Recherche Opérationelle 33, 1-14.

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.

Toth, P., Vigo, D., 1996. A heuristic algorithm for the vehicle routing problem with backhauls. Transportation Analysis, Editors: Lucio Bianco and Paolo Toth, 585-608.

Toth, P., Vigo, D., 1997. An exact algorithm for the vehicle routing problem with backhauls. Transportation Science 31(4), 372-385.

Tütüncü, Y., 2010. An interactive GRAMPS algorithm for the heterogeneous fixed fleet vehicle routing problem with and without backhauls. European Journal of Operational Research 201, 593-600.

Wassan, N., 2007. Reactive Tabu Adaptive Memory Programming Search for the Vehicle Routing Problem with Backhauls. Journal of the Operational Research Society 58(12), 1630-1641.

Wassan, N., Osman, I., 2002. Tabu Search Variants for the Mix Fleet Vehicle Routing Problem. Journal of the Operational Research Society, 53(7), 768-782.

Wassan, N., Nagy, G., Ahmadi, S., 2008a. A heuristic method for the vehicle routing problem with mixed deliveries and pickups. Journal of Scheduling 11(2), 149-161.

Wassan, N., Wassan, A., Nagy, G., 2008b. A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. Journal of Combinatorial Optimization 15(4), 368-386.

Table 1: FSMVRPB instances data.

No	*N*	*L*	*B*	Q_A	f_A	Q_B	f_B	Q_C	f_C	Q_D	f_D	Q_E	f_E	Q_F	f_F
HWS1	20	10	10	20	20	30	35	40	50	70	120	120	225		
HWS2	20	13	7	20	20	30	35	40	50	70	120	120	225		
HWS3	20	16	4	20	20	30	35	40	50	70	120	120	225		
HWS4	20	10	10	60	1000	80	1500	150	3000						
HWS5	20	13	7	60	1000	80	1500	150	3000						
HWS6	20	16	4	60	1000	80	1500	150	3000						
HWS7	20	10	10	20	20	30	35	40	50	70	120	120	225		
HWS8	20	13	7	20	20	30	35	40	50	70	120	120	225		
HWS9	20	16	4	20	20	30	35	40	50	70	120	120	225		
HWS10	20	10	10	60	1000	80	1500	150	3000						
HWS11	20	13	7	60	1000	80	1500	150	3000						
HWS12	20	16	4	60	1000	80	1500	150	3000						
HWS13	50	25	25	20	20	30	35	40	50	70	120	120	225	200	400
HWS14	50	33	17	20	20	30	35	40	50	70	120	120	225	200	400
HWS15	50	40	10	20	20	30	35	40	50	70	120	120	225	200	400
HWS16	50	25	25	120	1000	160	1500	300	3500						
HWS17	50	33	17	120	1000	160	1500	300	3500						
HWS18	50	40	10	120	1000	160	1500	300	3500						
HWS19	50	25	25	50	100	100	250	160	450						
HWS20	50	33	17	50	100	100	250	160	450						
HWS21	50	40	10	50	100	100	250	160	450						
HWS22	50	25	25	40	100	80	200	140	400						
HWS23	50	33	17	40	100	80	200	140	400						
HWS24	50	40	10	40	100	80	200	140	400						
HWS25	75	37	38	50	25	120	80	200	150	350	320				
HWS26	75	50	25	50	25	120	80	200	150	350	320				
HWS27	75	60	15	50	25	120	80	200	150	350	320				
HWS28	75	37	38	20	10	50	35	100	100	150	180	250	400	400	800
HWS29	75	50	25	20	10	50	35	100	100	150	180	250	400	400	800
HWS30	75	60	15	20	10	50	35	100	100	150	180	250	400	400	800
HWS31	100	50	50	100	500	200	1200	300	2100						
HWS32	100	66	34	100	500	200	1200	300	2100						
HWS33	100	80	20	100	500	200	1200	300	2100						
HWS34	100	50	50	60	100	140	300	200	500						
HWS35	100	66	34	60	100	140	300	200	500						
HWS36	100	80	20	60	100	140	300	200	500						

n: number of customers; *L*: number of linehaul customers; *B*: number of backhaul customers; Q_v: capacity of vehicle *v*; f_v: fixed cost of vehicle type *v; v = A, B … F*

Table 2: Optimal, LB/UB bounds with CPLEX (2 hour time limit using best-bound strategy)

Problem details				CPLEX Solution (Formulation without tightening)		CPLEX solution (Formulation with tightening)	
Problem	Size	L	B	UB	Optimal/LB (Time in secs)	UB	Optimal/LB (Time in secs)
HWS1	20	10	10	**720.57**	***720.57*** *(294)*	***720.57***	***720.7*** *(175)*
HWS2	20	13	7	**818.12**	***818.12*** *(1064)*	***818.12***	***818.12*** *(1498)*
HWS3	20	16	4	**848.23**	***848.23*** *(548)*	***848.23***	***848.23*** *(1242)*
HWS4	20	10	10	**4342.48**	***4342.48*** *(4)*	***4342.48***	***4342.48*** *(2)*
HWS5	20	13	7	**5357.98**	***5357.98*** *(69)*	***5357.98***	***5357.98*** *(85)*
HWS6	20	16	4	**5421.63**	***5421.63*** *(1567)*	***5421.65***	***5421.63*** *(2249)*
HWS7	20	10	10	**729.50**	***729.50*** *(57)*	***729.50***	***729.50*** *(38)*
HWS8	20	13	7	**838.11**	***838.11*** *(101)*	***838.11***	***838.11*** *(156)*
HWS9	20	16	4	**890.76***	*880.14*	***890.76***	***890.76*** *(6130)*
HWS10	20	10	10	**4349.13**	***4349.13*** *(7)*	***4349.13***	***4349.12*** *(4)*
HWS11	20	13	7	**5363.58**	***5363.58*** *(64)*	***5363.58***	***5363.58*** *(52)*
HWS12	20	16	4	**5497.97***	*5452.08*	***5497.98***	*5465.40*
HWS13	50	25	25	1685.97	*1481.96*	*1632.16*	*1480.75*
HWS14	50	33	17	1817.67	1704.11	1898.58	1705.44
HWS15	50	40	10	2073.93	1929.35	2060.37	1937.24
HWS16	50	25	25	5592.97	5115.66	5558.37	5537.24
HWS17	50	33	17	**6547.93**	**6547.93** *(3477)*	**6547.93**	**6547.93** *(5076)*
HWS18	50	40	10	7590.69	6860.21	7693.79	7026.15
HWS19	50	25	25	*1643.36*	1509.32	1656.35	1509.41
HWS20	50	33	17	2088.72	1925.55	2073.07	1926.03
HWS21	50	40	10	2363.80	2157.31	2381.68	2196.32
HWS22	50	25	25	1717.55	1694.88	1717.55	1697.38
HWS23	50	33	17	2179.09	2029.01	2206.23	2028.35
HWS24	50	40	10	2618.39	2337.11	2485.95	2335.56
HWS25	75	37	38	1475.40	1228.95	1546.75	1228.48
HWS26	75	50	25	1599.49	1348.49	1503.14	1341.97
HWS27	75	60	15	1862.38	1463.34	1617.19	1466.24
HWS28	75	37	38	2038.02	1487.99	2322.16	1486.89
HWS29	75	50	25	2779.15	1665.78	1887.32	1660.15
HWS30	75	60	15	2665.26	1847.10	2757.74	1853.72
HWS31	100	50	50	7615.37	4881.46	8766.06	4870.54
HWS32	100	66	34	7491.00	5903.18	7180.29	5901.57
HWS33	100	80	20	9006.20	6889.46	8721.29	6886.44
HWS34	100	50	50	2929.31	2374.40	3324.57	2369.80
HWS35	100	66	34	3332.50	2728.35	4539.46	2730.67
HWS36	100	80	20	3755.16	3140.05	4206.21	3139.87

***:** Optimal solution were observed when extra time was allowed and other CPLEX parameters attempted (such as depth first).
Time**:** CPU time in seconds (recorded if the time was less than 2 hours)
Underline**:** shows the tighter bounds (upper or lower).

Table 3: Comparison of three frameworks results.

		Sweep		*Framework-1*		*Framework-2*		*Framework-3*		*Heuristic Best*
Problem	***n***	***Solution***	***CPU***	***Solution***	***CPU***	***Solution***	***CPU***	***Solution***	***CPU***	
HWS1	20	765.29	0.20	811.77	0.84	**726.48**	0.61	**726.48**	0.59	**726.48**
HWS2	20	919.50	0.30	889.41	0.42	**818.12**	1.09	**818.12**	0.89	**818.12**
HWS3	20	937.39	0.41	880.38	0.66	**848.59**	1.64	**848.59**	0.98	**848.59**
HWS4	20	4398.06	0.28	4381.13	0.27	**4350.65**	0.91	**4350.65**	0.48	**4350.65**
HWS5	20	5427.87	0.34	5406.41	0.38	**5366.39**	2.75	**5366.39**	0.75	**5366.39**
HWS6	20	6331.74	0.42	5911.26	0.66	**5875.23**	3.44	**5875.23**	1.3	**5875.23**
HWS7	20	791.50	0.20	776.75	0.25	**767.93**	0.58	788.65	0.69	**767.93**
HWS8	20	959.18	0.27	915.53	0.36	**872.97**	1.39	900.27	0.47	**872.97**
HWS9	20	945.88	0.45	995.25	0.63	**903.18**	2.09	917.93	0.77	**903.18**
HWS10	20	4429.37	0.24	4423.98	0.25	**4365.44**	0.88	4379.64	0.36	**4365.44**
HWS11	20	5878.81	0.27	5927.87	0.38	**5414.5**	2.72	5857.03	0.47	**5414.5**
HWS12	20	6340.10	0.38	6336.83	0.61	**5928.78**	4.94	6321.41	0.75	**5928.78**
HWS13	50	1752.72	1.66	1728.51	3.47	**1625.7**	17.88	**1625.7**	11.81	**1625.7**
HWS14	50	1920.65	4.53	1902.36	10.55	**1811.63**	26.19	1825.99	18.06	**1811.63**
HWS15	50	2120.89	8.95	2207.59	20.83	**2018.93**	38.42	2037.43	39.08	**2018.93**
HWS16	50	5629.79	1.83	5634.7	3.56	**5561.67**	330.34	**5561.67**	4.5	**5561.67**
HWS17	50	7075.39	4.53	6679.18	9.89	**6570.39**	996.55	6596.97	16.03	**6570.39**
HWS18	50	8152.40	9.66	7697.89	30.38	**7599.08**	1120.5	7634.94	54.75	**7599.08**
HWS19	50	1729.30	1.97	1744.49	3.44	**1704.41**	39.81	1751.59	5.02	**1704.41**
HWS20	50	2182.47	4.44	2245.85	11.89	**2037.23**	84.95	2063.87	12.72	**2037.23**
HWS21	50	2509.58	9.53	2550.63	17.17	**2340.09**	103.52	2435.18	23.45	**2340.09**
HWS22	50	1841.28	2.11	1817.79	3	**1774.71**	18.41	1990.6	4.3	**1774.71**
HWS23	50	2322.46	4.97	2317.93	8.31	**2166.52**	64.77	2189.97	9.38	**2166.52**
HWS24	50	2550.88	9.25	2638.87	17.44	**2430.88**	49.72	2539.94	18.8	**2430.88**
HWS25	75	1453.86	6.73	1456.01	16	**1332.02**	1006.28	**1332.02**	109.89	**1332.02**
HWS26	75	1568.82	20.81	1508.29	101.88	**1421.04**	1779.88	1431.97	133.33	**1421.04**
HWS27	75	1684.76	44.22	1679.28	185.36	**1534.65**	1996.59	**1534.65**	241.45	**1534.65**
HWS28	75	1976.80	6.67	1880.97	39.69	**1617.85**	1351.92	1660.03	37.47	**1617.85**
HWS29	75	2153.57	21.13	2044.62	121.09	**1799.76**	1513.3	1827.05	68.13	**1799.76**
HWS30	75	2467.32	45.31	2182.3	219.17	**1990.46**	2662.15	1993.36	242.78	**1990.46**
HWS31	100	5746.00	20.42	4943.29	71.14	**5201.81**	4257.41	**5201.81**	335.72	**4943.29**
HWS32	100	6898.01	85.70	6358.1	214.92	-		**6035.96**	1866.3	**6035.96**
HWS33	100	8410.82	146.88	7723.52	649.82	-		**7601.09**	950.66	**7601.09**
HWS34	100	2813.74	20.45	**2465.41**	55.31	2646.52	2871.74	2671.66	119.67	**2465.41**
HWS35	100	3378.77	65.69	3280.1	140.64	**2971.98**	651.8	**2971.98**	471.84	**2971.98**
HWS36	100	3782.32	150.08	3806.15	305.51	**3533.9**	1729.3	3565.49	829.44	**3533.9**
ARPD (%)		**8.41**		**6.09**		**0.37**		**2.08**		
Avg *CPU*			**19.48**		**62.94**		**668.66**		**156.47**	
Min *CPU*			**0.2**		**0.25**		**0.58**		**0.36**	
Max *CPU*			**150**		**649**		**4257**		**1836**	

RPD: Relative Percentage Deviation from the best [RPD = (Heuristic Solution-H-Best)/H-Best*100]
ARPD: Average of RPD
CPU: CPU time in seconds.

Table 4: Performance of the proposed heuristic on the special case the standard FSMVRP: Comparison against the best algorithms published in the literature.

Problem	***Our solution***	***Overall Best solutions***
3	961.03	961.03 [WO,CT,B,I]
4	6945.93	6437.33 [WO,CT,B,I]
5	1008.59	1007.05 [WO,CT,B,I]
6	6973.89	6516.47 [WO,CT,B,I]
13	2434.48	2406.36 [CT,B,I]
14	9120.35	9119.03 [CT,B,I]
15	2644.55	2586.37 [WO,CT,B,I]
16	2782.54	2720.43 [CT,I]
17	1761.17	1734.53 [B]
18	2403.25	2369.65 [B,I]
19	8860.19	8659.74 [WO]
20	4105.86	4039.49 [CT]
ARPD (%)	2.308	

WO: Wassan and Osman (2002); CT: Choi and Tcha (2007); B: Brandao (2009); I: Imran *et al*. (2009)
RPD: Relative Percentage Deviation from the best [RPD = (Heuristic Solution - Best) / Best*100]
ARPD: Average of RPD