The Heterogeneous Flexible Periodic Vehicle Routing Problem: Mathematical formulations and solution algorithms

Article *in* Computers & Operations Research · January 2022

DOI: 10.1016/j.cor.2021.105662

4 authors:

Diana L. Huerta-Muñoz
Autonomous University of Nuevo León

C. Archetti
ESSEC Business School in Paris

Elena Fernandez
Universitat Politècnica de Catalunya

Federico Perea
Universidad de Sevilla

The Heterogeneous Flexible Periodic Vehicle Routing Problem: Mathematical formulations and solution algorithms

Diana L. Huerta-Muñoz[*(1)] *Claudia Archetti*[(2)] *Elena Fernández* [(3)] *Federico Perea*[(4)]

[(1)] *Graduate Program in Systems Engineering*
Universidad Autónoma de Nuevo León, Mexico
dianahuerta@yalma.fime.uanl.mx
[(2)] *Department of Information Systems, Decision Sciences and Statistics*
ESSEC Business School, France
archetti@essec.edu
[(3)] *Department of Statistics and Operations Research*
Universidad de Cádiz, Spain
elena.fernandez@uca.es
[(4)] *Department of Applied Mathematics II*
Higher Polytechnic School, Universidad de Sevilla, Spain
perea@us.es

Abstract

The aim of this paper is to introduce the Flexible Periodic Vehicle Routing Problem with Heterogeneous Fleet, a variant of the Periodic Vehicle Routing Problem. Flexibility is introduced in service schedules and delivered quantities, heterogeneity comes from different vehicles capacities and speeds. Three Mixed-Integer Linear Programming formulations and a matheuristic, based on Kernel Search, are proposed. Computational tests are made to evaluate the performance of the three formulations and to assess the quality of the solutions provided by the matheuristic.

Keywords: Kernel Search; Matheuristics; Vehicle Routing; Heterogeneous Fleet.

1. Introduction

In this work we introduce the Heterogeneous Flexible Periodic Vehicle Routing Problem (HFPVRP), which generalizes the Flexible Periodic Vehicle Routing Problem (FPVRP) [see Archetti et al., 2017] in the sense that not all the available vehicles have the same characteristics. The effect of this generalization is twofold. On the one hand, under similar traffic conditions, vehicles of different types travel at possibly different average speeds, thus incurring different routing costs. On the other hand, the capacity of the vehicles may vary among different types.

The FPVRP is, in turn, a generalization of the Periodic Vehicle Routing Problem (PVRP) introduced by Beltrami and Bodin [1974], where the aim is to define a distribution plan to serve a set of customers with periodic demand over a given planning horizon. In particular, in the PVRP we have potential schedules for customers, on the basis of the visit frequency, and each customer will be visited at every time period according to the chosen schedule. The quantity delivered at each customer visit is constant. On the contrary, the FPVRP imposes no restriction related to the schedule. Each customer is associated with a total demand, which has to be served within the time horizon, and a maximum quantity that can be delivered in each time period. Then, any distribution plan that satisfies these two sets of constraints, together with the vehicle capacity constraints, defines a feasible schedule. Also, the quantity delivered at each visit is not fixed in advance and can thus vary from visit to visit. Applications of the FPVRP are related to distribution problems in which the binding constraints are those related to the total quantity that has to be delivered to each customer in the planning period and to the maximum quantity that can be delivered in each visit like, for example, delivery operations for interlibrary loan items [see Francis et al., 2006].

The study of the FPVRP and the HFPVRP goes in the direction of a growing research trend in the routing literature, namely, relaxing modeling assumptions and allowing more flexible settings than those of classical routing problems. Two well-known examples are the Split Delivery Vehicle Routing Problem (SDVRP) [see Archetti and Speranza, 2012], which generalizes the VRP by relaxing the single visit constraint, and the Maximum-Level policy in Inventory Routing Problems (IRPs), which relaxes the constraints of filling the inventory level up to capacity at each customer visit [see Archetti et al., 2007]. Indeed, Archetti et al. [2008] and Archetti and Speranza [2016] show that flexibility can produce remarkable total cost savings in the SDVRP and in the IRP setting, respectively. Specifically, Archetti and Speranza [2016] study the advantages of a distribution policy which relaxes constraints related to delivery frequency and fixed delivery quantity in the IRP, in the same vein as for the FPVRP with respect to the PVRP.

In the current work, we aim at continuing in this research trend by studying a generalization of the FPVRP that considers a heterogeneous fleet, rather than a homogeneous one. As discussed in Section 2, the existing literature on routing problems with heterogeneous fleets is much narrower than the one related to homogeneous fleets. However, many practical applications deal with heterogeneous fleets. For example, distribution activities may involve different types of vehicles that are deployed according to the characteristics of the service area (narrow vs. large streets, areas that forbid the access to fuel-engine vehicles, etc.). This is becoming more and more common, especially in distribution operations in urban areas.

The purpose of this paper is to explore formulations and solution approaches for the HFPVRP that are tailored to exploit the heterogeneous fleet structure. To this end, we follow two different approaches. First, we focus on three alternative mixed-integer linear programming (MILP) formulations for the HFPVRP. Even if these formulations progressively outperform each other, all of them clearly highlight the difficulty of the HFPVRP, as only very small-size instances can be solved to proven optimality in one hour. Thus the need for alternative algorithms that provide good quality solutions in shorter computing times becomes evident. For this reason, we also propose a Kernel Search matheuristic, which combines heuristically driven decisions with decisions driven by the information obtained from MILP formulations.

The contributions of this paper can be summarized as follows:

- We introduce the heterogeneous FPVRP and provide different mathematical formulations for the problem.
- We identify the best formulation through computational tests.

- We propose a solution approach based on the Kernel Search scheme.
- We provide an extensive computational analysis to assess the performance of the algorithm.

The rest of this paper is organized as follows. The relevant literature is overviewed in Section 2. A formal definition of the problem is provided in Section 3. Three different mathematical formulations are presented in Section 4, and the Kernel Search algorithm is described in Section 5. Computational experiments and results are presented and analyzed in Section 6, while conclusions are drawn in Section 7.

2. Literature review

Since the HFPVRP is a generalization of the more classical PVRP, we start by providing a brief overview of the literature on the PVRP, which was initially introduced by Beltrami and Bodin [1974]. Early mathematical formulations were proposed in Russell and Igo [1979], and Christofides and Beasley [1984]. The literature on this problem is extensive, and the interested reader is referred to the survey provided by Campbell and Wilson [2014].

Successive studies have progressively incorporated some sort of flexibility in the PVRP setting. In particular, Francis et al. [2006] introduced the PVRP with Service Choice (PVRP-SC) where the frequency of visit of each customer is not fixed and is, instead, one of the decisions of the problem. Still, once the frequency is determined, the visit schedule follows a defined calendar satisfying the corresponding frequency. Further studies on the problem are available in Francis and Smilowitz [2006], Francis et al. [2007, 2008]. Note that the FPVRP differs from the PVRP-SC in that any visit schedule (and frequency) is allowed. In addition, in the PVRP-SC, once the frequency of visit of a customer is fixed, the quantity delivered in each visit is constant and determined as the ratio between the total quantity and the number of visits. Instead, in the FPVRP, the quantity delivered has to be decided and can vary from visit to visit.

As mentioned above, the FPVRP was introduced in Archetti et al. [2017], where the authors theoretically showed that, by relaxing the constraints of complying with pre-defined visit schedules, it is possible to achieve remarkable cost savings. Different mathematical formulations were proposed and an extensive computational study was presented with the aim of showing the actual benefits achieved by introducing the above mentioned flexibility in the PVRP setting. In Archetti et al. [2018] the same authors proposed a matheuristic algorithm to solve the problem.

As already stated, the aim of the current work is to study the HFPVRP, i.e., the FPVRP with an heterogeneous fleet of vehicles. The literature on routing problems with an heterogeneous fleet is notably narrower than the one considering an homogeneous fleet. We refer to Koç et al. [2016] for a survey on the VRP with heterogeneous fleet. To the best of our knowledge, there are just a few contributions related to the PVRP with heterogeneous fleet, mainly related to real-case studies [see Abreu and Arroyo, 2015, Baptista et al., 2002, Angelelli and Speranza, 2002].

3. Problem Definition

In Archetti et al. [2017] the FPVRP is defined as follows. A complete directed graph $G = (N, A)$ is given, where the set of nodes is $N = \{0\} \cup C$, being $C = \{1, ..., n\}$ the set of customers, and 0 the depot where vehicles start and end their routes. A set of time periods $P = \{1, ..., p\}$ is given, where p is the planning horizon. Each customer $i \in C$ has a deterministic total demand W_i over the whole time horizon. Also, a maximum quantity w_i can be delivered to customer i in each visit. No split deliveries are allowed at any time period, i.e., each customer can be visited at most once in each time period. A homogeneous fleet of vehicles $M = \{1, ..., m\}$ is available, each with a capacity of Q units. Each time an arc $(i, j) \in A$ is traversed, a routing cost $c_{ij} \geq 0$ is incurred. We assume that costs c_{ij} satisfy the triangle inequality. The FPVRP consists in finding a set of routes that minimize the total routing costs, satisfying the vehicles capacities as well as the demand and maximum delivery quantity of all customers.

The HFPVRP is a natural extension of the FPVRP that arises when the available fleet of vehicles is heterogeneous. Let $V = \{1, ..., v\}$ be the set of vehicle types. Let also $M_h = \{1, ..., m_h\}$ be the set of vehicles of type $h \in V$, where m_h denotes the number of available vehicles of type h and $M = \bigcup_{h=1}^{v} M_h$. All vehicles of the same type are associated with the same capacity and the same routing costs, which are defined as follows:

1. c_{ij}^h: routing cost for a vehicle of type $h \in V$ to traverse arc (i, j).
2. Q^h: capacity of a vehicle of type $h \in V$.

For the sake of readability, and abusing notation, in the following we refer to c_{ij}^k as the routing cost of vehicle $k \in M$, and Q^k as the capacity of vehicle k. In the following, we assume that routing costs c_{ij}^h satisfy the triangle inequality for each vehicle type h. In addition, we define $\delta^+(i)$ and $\delta^-(i)$ as the set of arcs leaving and entering node i, respectively.

4. Mathematical formulations for the HFPVRP

In this section we present three MILP formulations for the HFPVRP. We start with a formulation that is a natural extension of the one proposed in Archetti et al. [2017] for the homogeneous case, and then propose two more formulations tailored to the HFPVRP.

4.1. Formulation with vehicle index variables

The first formulation that we consider uses the following decision variables:

- y_{ij}^{kt}: Binary routing variable that takes the value 1 if arc $(i, j) \in A$ is traversed by vehicle $k \in M$ at time period $t \in P$, and 0 otherwise.

- z_i^{kt}: Binary assignment variable that takes the value 1 if customer $i \in C$ is visited by vehicle $k \in M$ at time period $t \in P$, and 0 otherwise.

- q_i^{kt}: Continuous non-negative variable that indicates the quantity delivered to customer $i \in C$ by vehicle $k \in M$ at time period $t \in P$.

- u_i^t: Continuous non-negative variable that prevents subtours in Miller-Tucker-Zemlin (MTZ)-type constraints [Miller et al., 1960].

The first MILP proposed, denoted as Formulation 1 (F1), is:

$$\text{(F1)} \quad \min \sum_{t \in P} \sum_{k \in M} \sum_{(i,j) \in A} c_{ij}^k y_{ij}^{kt} \tag{1}$$

$$\text{s.t.:} \quad z_0^{kt} \leq 1 \qquad k \in M, t \in P \tag{2}$$

$$\sum_{k \in M} z_i^{kt} \leq 1 \qquad i \in C, t \in P \tag{3}$$

$$\sum_{(i,j) \in \delta^+(i)} y_{ij}^{kt} = z_i^{kt} \qquad i \in N, k \in M, t \in P \tag{4}$$

$$\sum_{(i,j) \in \delta^+(i)} y_{ij}^{kt} = \sum_{(j,i) \in \delta^-(i)} y_{ji}^{kt} \qquad i \in N, k \in M, t \in P \tag{5}$$

$$\sum_{t \in P} \sum_{k \in M} q_i^{kt} = W_i \qquad i \in C \tag{6}$$

$$q_i^{kt} \leq w_i z_i^{kt} \qquad i \in C, k \in M, t \in P \tag{7}$$

$$\sum_{i \in C} q_i^{kt} \leq Q^k z_0^{kt} \qquad k \in M, t \in P \tag{8}$$

$$u_j^t \geq u_i^t + 1 - (n-1)(1 - \sum_{k \in M} y_{ij}^{kt}) \qquad t \in P, (i,j) \in A, j \neq 0 \tag{9}$$

$$u_0^t = 1 \qquad t \in P \tag{10}$$

$$\sum_{k \in M} z_i^{tk} \leq u_i^t \leq (n-1) \sum_{k \in M} z_i^{tk} \qquad i \in C, t \in P \tag{11}$$

$$z_i^{kt} \geq 0 \qquad i \in N, k \in M, t \in P \tag{12}$$

$$u_i^t \geq 0 \qquad i \in C, t \in P \tag{13}$$

$$q_i^{kt} \geq 0 \qquad i \in C, k \in M, t \in P \tag{14}$$

$$y_{ij}^{kt} \in \{0, 1\} \qquad (i,j) \in A, k \in M, t \in P \tag{15}$$

The objective (1) defines the total routing cost. Inequalities (2)-(3) establish that, at each time period, each vehicle is used at most once and at most one vehicle serves each customer. Constraints (4) state that, for every vehicle and time period, one arc must leave every visited node and no arc can leave a non-visited node, whereas the equalities (5) guarantee flow conservation. Constraints (6) ensure that the total demand of each customer is satisfied, and inequalities (7) impose that, at any time period, no customer receives a quantity exceeding its maximum delivery quantity. Constraints (8) guarantee that the capacity of the vehicles is respected. The MTZ-type constraints (9) prevent subtours, and (10)-(11) set suitable bounds for the variables. Finally, (12)-(15) define the domain of the variables.

Because of constraints (3)-(4), the binary condition on the assignment variables z associated with customers $i \in C$ can be relaxed to continuous non-negative, as these variables will take binary values due to the binary nature of the y variables. Still, for the index $i = 0$ corresponding to the depot, we have to impose constraint (2), since the depot is not affected by constraints (3), which are not valid for $i = 0$ (they would imply that at most one vehicle can be used at each time period).

Several variations of F1 or alternative formulations can be derived. Below we present two of them, which outperformed other alternatives that we tested in preliminary experiments.

4.2. Formulation with aggregated assignment and quantity variables

The formulation in this section has two main differences with respect to F1. On the one hand, it uses aggregated assignment and quantity variables over all vehicles to identify the customers that are served at each time period and the quantities delivered. On the other hand, subtours are prevented using load variables associated with the traversed arcs. The formulation uses the same y and u variables as in F1, plus the following ones:

- $\widetilde{z}_i^t$: binary variable that takes the value 1 if customer $i \in C$ is visited at time period $t \in T$, and 0 otherwise.
- $\widetilde{q}_i^t$: continuous non-negative variable determining the quantity delivered to customer $i \in C$ at time period $t \in T$.
- l_{ij}^t: continuous non-negative variable that indicates the load of the vehicle when traversing arc $(i,j) \in A$ at time period $t \in P$. The definition of these variables implicitly takes into account that at each time period at most one vehicle will traverse each arc.

The reader may note the relation between some of these new variables and those used in F1: $\widetilde{z}_i^t = \sum_{k \in M} z_i^{kt}$ and $\widetilde{q}_i^t = \sum_{k \in M} q_i^{kt}$.

The second formulation we propose (F2) is:

$$
\begin{aligned}
\text{(F2)} \quad & \min \sum_{t \in P} \sum_{k \in M} \sum_{(i,j) \in A} c_{ij}^k y_{ij}^{kt} && && (16) \\
\text{s.t.:} \quad & \sum_{(0,j) \in \delta^+(0)} y_{0j}^{kt} \leq 1 && k \in M, t \in P && (17) \\
& \sum_{k \in M} \sum_{(i,j) \in \delta^+(i)} y_{ij}^{kt} = \widetilde{z}_i^t && i \in C, t \in P && (18) \\
& \sum_{(i,j) \in \delta^+(i)} y_{ij}^{kt} = \sum_{(j,i) \in \delta^-(i)} y_{ji}^{kt} && i \in N, k \in M, t \in P && (19) \\
& \sum_{t \in P} \widetilde{q}_i^t = W_i && i \in C && (20) \\
& \widetilde{q}_i^t \leq w_i \widetilde{z}_i^t && i \in C, t \in P && (21) \\
& l_{ij}^t \leq \sum_{k \in M} Q^k y_{ij}^{kt} && (i,j) \in A, t \in P && (22) \\
& \sum_{(i,j) \in \delta^+(i)} l_{ij}^t = \sum_{(j,i) \in \delta^-(i)} l_{ji}^t - \widetilde{q}_i^t && i \in C, t \in P && (23) \\
& u_j^t \geq u_i^t + 1 - (n-1)(1 - \sum_{k \in M} y_{ij}^{kt}) && i \in C, j \in C, t \in P && (24) \\
& u_0^t = 1 && t \in P && (25)
\end{aligned}
$$

$$\widetilde{z}_i^t \leq u_i^t \leq (n-1)\widetilde{z}_i^t \qquad i \in C, t \in P \tag{26}$$

$$u_i^t \geq 0 \qquad i \in C, t \in P \tag{27}$$

$$\widetilde{q}_i^t \geq 0 \qquad i \in C, t \in P \tag{28}$$

$$0 \leq \widetilde{z}_i^t \leq 1 \qquad i \in N, t \in P \tag{29}$$

$$l_{ij}^t \geq 0 \qquad (i,j) \in A, t \in P \tag{30}$$

$$y_{ij}^{kt} \in \{0,1\} \qquad (i,j) \in A, t \in P \tag{31}$$

While constraints analogous to (3) are no longer needed, since they are imposed by the rationale of the aggregated $\widetilde{z}$ variables, constraints (17), (18)-(19) and (20)-(21) play now a similar role to the F1 constraints (2), (4)-(5) and (6)-(7), respectively. The capacity constraints on the vehicles are now imposed using the load variables through inequalities (22). Load variables are also used in the load balance constraints (23), which regulate the load of the vehicles when traversing the arcs, and are now used to prevent subtours. Indeed, constraints (24)-(27) are no longer needed, although we keep them in the formulation with the only purpose of improving the bound and speeding up the process of optimally solving the instances. Note that (29) relaxes variables $\widetilde{z}$, analogously as in formulation F1.

4.3. Formulation with aggregated routing variables

In the formulation that we introduce in this section we keep the aggregated $\widetilde{z}$ and $\widetilde{q}$ variables as well as the vehicle load variables (l) used in F2. However we use different routing variables, which are now aggregated over all the vehicles of the same type. The rationale behind this aggregation is that, at each time period, at most one vehicle of each type will visit each customer due to the assumptions that the graph is complete and that the triangle inequality holds. In principle, it could be possible that some arc be traversed by more than one vehicle of the same type at some time period. However, the triangle inequality assumption on the routing costs together with the assumption that the graph is complete, allow us to avoid such a case by applying shortcuts.

In particular, we introduce the following set of decision variables:

- $\widetilde{y}_{ij}^{ht}$: Binary routing variable that takes the value 1 if and only if arc $(i,j) \in A$ is traversed by a vehicle of type $h \in V$ at time period $t \in P$. Note the relation with the variables used in formulations F1 and F2: $\widetilde{y}_{ij}^{ht} = \sum_{k \in M_h} y_{ij}^{kt}$.

The resulting formulation (F3) is:

$$\text{(F3)} \quad \min \sum_{t\in P}\sum_{h\in V}\sum_{(i,j)\in A} c^h_{ij}\widetilde{y}^{ht}_{ij} \tag{32}$$

$$\text{s.t.:} \quad \sum_{(0,j)\in\delta^+(0)} \widetilde{y}^{ht}_{0j} \le m_h \qquad h\in V, t\in P \tag{33}$$

$$\sum_{h\in V}\sum_{(i,j)\in\delta^+(i)} \widetilde{y}^{ht}_{ij} = \widetilde{z}^t_i \qquad i\in C, t\in P \tag{34}$$

$$\sum_{(i,j)\in\delta^+(i)} \widetilde{y}^{ht}_{ij} = \sum_{(j,i)\in\delta^-(i)} \widetilde{y}^{ht}_{ji} \qquad i\in N, h\in V, t\in P \tag{35}$$

$$\widetilde{q}^t_i \le w_i z^t_i \qquad i\in C, t\in P \tag{36}$$

$$\sum_{t\in P} \widetilde{q}^t_i = W_i \qquad i\in C \tag{37}$$

$$\sum_{(i,j)\in\delta^+(i)} l^t_{ij} = \sum_{(j,i)\in\delta^-(i)} l^t_{ji} - \widetilde{q}^t_i \qquad i\in C, t\in P \tag{38}$$

$$l^t_{ij} \le \sum_{h\in V} Q^h \widetilde{y}^{ht}_{ij} \qquad (i,j)\in A, t\in P \tag{39}$$

$$u^t_j \ge u^t_i + 1 - (n-1)(1 - \sum_{h\in V} \widetilde{y}^{ht}_{ij}) \qquad i\in C, j\in C, t\in P \tag{40}$$

$$u^t_0 = 1 \qquad t\in P \tag{41}$$

$$\widetilde{z}^t_i \le u^t_i \le (n-1)\widetilde{z}^t_i \qquad i\in C, t\in P \tag{42}$$

$$u^t_i \ge 0 \qquad i\in C, t\in P \tag{43}$$

$$\widetilde{q}^t_i \ge 0 \qquad i\in C, t\in P \tag{44}$$

$$0 \le \widetilde{z}^t_i \le 1 \qquad i\in N, t\in P \tag{45}$$

$$l^t_{ij} \ge 0 \qquad (i,j)\in A, t\in P \tag{46}$$

$$\widetilde{y}^{ht}_{ij} \in \{0,1\} \qquad (i,j)\in A, h\in V \tag{47}$$

Constraints (33) impose that, at each time period, no more than m_h vehicles of type $h \in V$ leave the depot, and constraints (34)-(35) are analogous to (18)-(19). The remaining sets of constraints are the same as in F2, except for the MTZ constraints (40), which are now expressed in terms of the new routing variables $\widetilde{y}$.

5. A Kernel Search algorithm for the HFPVRP

The Kernel Search (KS) algorithm was proposed for the first time in Angelelli et al. [2010] for the multi-dimensional knapsack problem. Starting from this pioneering paper, different applications to other combinatorial optimization problems have been proposed (Angelelli et al. [2012], Guastaroba and Speranza [2014], Filippi et al. [2016], Carvalho and Nascimento [2018], Archetti et al. [2021]). KS embeds mathematical programming in a heuristic framework to obtain good quality solutions efficiently. The main idea is to split a whole set of integer/binary decision variables into several subsets called *kernel* and *buckets* and, in an iterative process, to solve a series of restricted MILPs (RMILPs) considering only the variables in the *kernel* and one *bucket* at a time, fixing to zero the remaining ones. The *kernel* is composed of the most promising variables, those that are more likely to be selected in a good solution. On the other hand, the *buckets* define a partition of the remaining variables sorted from the most promising to the least promising according to certain criteria. Each bucket is therefore a subset of variables which are not in the kernel.

5.1. Kernel Search for the HFPVRP

The scheme of the proposed KS for the HFPVRP, called KSHFP from now on, is shown in Algorithm 1. The input for the algorithm consists of the original set of routing variables $\mathbf{y}$ and the bucket size L_B. Two main phases are considered: the *initialization phase* and the *improvement phase*. In the first one (lines 1-2), a heuristic algorithm (H) is run to create an initial solution and to gather information about promising arc variables. The best solution found by H (denoted by s^{UB}) is used as a part of the initial kernel $\mathcal{K}$, and the optimal value of such solution (denoted as z^{UB}) is considered to compute a cut-off value $\texttt{CutOff} = z^{UB} - \epsilon$, where ϵ is a very small number. This `CutOff` is used to speed up the solver by discarding all the solutions whose objective value is greater than it. The remaining arc variables are obtained from a set of matrices, one per time period $t \in P$, where each element (i, j) of a matrix is associated with arc $(i, j) \in A$. These matrices are built when running H (see details in Section 5.1.1), and their entries indicate the number of times arc $(i, j) \in A$ is traversed at time period $t \in P$ in the solutions found by H. The matrices are used to populate the initial kernel and to create a bucket list $\mathcal{B}$, where arc variables are arranged by non-increasing values of the number of visits in the matrices. Note that $\mathcal{B}$ is a list, each of its elements being a bucket. At the end of the initialization, the original MILP is solved restricted to the variables in $\mathcal{K}$.

At each iteration of the *Improvement phase* (lines 4-17), a bucket $B_i \in \mathcal{B}$ of size L_B is selected. Then, the corresponding RMILP is solved by considering only the subset of variables in $\mathcal{K} \cup B_i$. If a better solution s is found (note that any feasible solution improves the incumbent because of the cut-off value), the current kernel $\mathcal{K}$, the best solution found s^{UB}, and the z^{UB} and `CutOff` values are updated. The MILP that we solve at each iteration of the algorithm is formulation (F3) described in Section 4.3, as it provides the best trade-off between solution quality and computing time (see the experiments section), among the formulations we tested.

The KSHFP stops when a maximum number of buckets N_b is analysed or a maximum time limit is reached. We describe with more detail the main components of the initialization and improvement phases in the following sections.

Algorithm 1 The KSHFP algorithm.

Input: $\mathbf{y}$: Original set of arc variables, L_B: Bucket size.

$\triangleright$ **Phase 1: Initialization**

$(s^{UB}, z^{UB}, \mathcal{K}, \mathcal{B}) \leftarrow$ `Initialization()`
$(s^{UB}, z^{UB}) \leftarrow \text{MILP}(\mathcal{K})$
`CutOff` $= z^{UB} - \epsilon$
$N_b = \left\lceil \frac{|\mathcal{B}|}{L_B} \right\rceil$ $\triangleright$ **Phase 2: Improvement**
$\ell = 1,\ B_1 = \emptyset$.
while $\ell \leq N_b$ and time limit is not reached **do**
    Construct B_ℓ from the bucket list $\mathcal{B}$.
    $(s, z) \leftarrow \text{RMILP}(\mathcal{K} \cup B_\ell, \texttt{CutOff})$
    if $z < z^{UB}$ **then**
        $\mathcal{K} \leftarrow \mathcal{K} \cup \{s\}$
        $s^{UB} \leftarrow s$
        $z^{UB} \leftarrow z$
        `CutOff` $= z^{UB} - \epsilon$
    end if
    $\mathcal{B} \leftarrow \mathcal{B} \setminus B_\ell$
    $++\ell$.
end while

Output:

s^{UB} : Best solution found.
z^{UB} : Best solution cost.

5.1.1. Initialization phase

The main goal of the *Initialization phase* (see Algorithm 2) is to identify the most promising arc variables to be inserted in the kernel and define a criterion to sort the remaining arcs in the bucket lists. This is done by running a heuristic for the HFPVRP and gathering relevant information on arcs 'quality', as an indicator of their likelihood of being part of high-quality solutions. We use an adaptation of the solution algorithm proposed in Archetti et al. [2018] for the homogeneous FPVRP. This matheuristic starts from a feasible solution that is constructed by first determining a distribution plan through the solution of a MILP (DPMILP), and then applying the Lin-Kernighan (LK) algorithm [Lin and Kernighan, 1973] for constructing vehicle routes (lines 2-3).

Given that the adaptation to the heterogeneous case of the DPMILP is not straightforward, we now provide further details.

The initial solution given to the Tabu Search (TS) is generated by solving the DPMILP (line 2) and then, by solving the routing part using the Lin-Kernighan heuristic (line 3) on the solution provided by assignment DP. In order to represent an heterogeneous fleet in this step, we proposed the following 3-vehicle index formulation.

Data:

- $\tilde{c}_i^{kt}$: approximate costs of serving customer $i \in C$ by vehicle $k \in M$ in time period $t \in P$. We define $\tilde{c}_i^{kt} = f_h^k$, where f_h^k is the speed factor of vehicle k of type $h \in V$. Notice that at the very beginning of the initialization phase it is enough to start with any feasible solution. Considering that we have several vehicle types, where the main difference is the speed factor f_h^k, the initial approximate costs are set to these values in such a way that vehicles with smaller f_h^k represent a "less expensive but slow" option to serve a given customer, while the vehicles with higher f_h^k are considered a "fast but costly" alternative (in the *homogeneous case* it is similar to consider all these values equal to one). In the following iterations of the initialization, these approximate costs are computed according to the best solution found by the TS as the cheapest insertion cost of visiting a customer in a route performed at each time period (as it is done in Archetti et al. [2018]).

Variables:

- $z_i^{kt} = 1$ if customer $i \in C$ is assigned to vehicle $k \in M$ in time period $t \in P$.
- q_i^{kt}: quantity delivered to customer $i \in C$ by vehicle $k \in M$ in time $t \in P$.

Then, the DPMILP is as follows:

$$\min \sum_{t \in P} \sum_{k \in M} \sum_{i \in C} \tilde{c}_i^{kt} z_i^{kt} \tag{48}$$

$$\text{s.t.: } q_i^{kt} \leq w_i z_i^{kt} \qquad i \in C, k \in M, t \in P \tag{49}$$

$$\sum_{i \in C} q_i^{kt} \leq Q^k z_0^{kt} \qquad k \in M, t \in P \tag{50}$$

$$\sum_{k \in M} z_i^{kt} \leq 1 \qquad i \in C, t \in P \tag{51}$$

$$\sum_{t \in P} \sum_{k \in M} q_i^{kt} = W_i \qquad i \in C \tag{52}$$

$$q_i^{kt} \geq z_i^{kt} \qquad i \in C, k \in M, t \in P \tag{53}$$

$$z_i^{kt} \in \{0, 1\} \qquad i \in N, k \in M, t \in P \tag{54}$$

$$q_i^{kt} \geq 0 \qquad i \in C, k \in M, t \in P \tag{55}$$

The objective function (48) minimizes the total approximate costs for all visited customers during the time horizon. Constraints (49) and (50) avoid to exceed the maximum customer capacity and the vehicle capacity, respectively. Constraints (51) state that each customer must be visited at most once at each time period. Constraints (52) impose to satisfy customer demands. Constraints (53) force to deliver at least one unit of product if customer i is visited by vehicle k at time t. Constraints (54)-(55) define the domain of variables.

The solution of DPMILP provides the schedule of visits per each vehicle. Then, the LK heuristic is applied to determine the vehicle routes at each time period. Finally, a feasible solution s with an objective value $f(s)$, which is considered as the initial incumbent solution s^{UB} with value z^{UB}, is provided.

Once an initial feasible solution s is obtained, the TS heuristic is applied to possibly improve it ($\tilde{s}$) and to obtain the matrices with the information related to the most traversed arcs (line 5). If $\tilde{s}$ is better than the incumbent, the best solution is updated (lines 6 - 9). Finally, the objective function of the DPMILP is updated considering the new solution (line 10). This procedure is iterated until a maximum number of iterations is met. The initialization phase ends with the creation of the initial kernel $\mathcal{K}$ and the bucket list $\mathcal{B}$ (lines 12 - 13).

Some additional considerations must be taken into account during the initialization phase:

1. In the TS heuristic, each time a new solution is visited, the information related to the arcs traversed in this solution is stored. In particular, a set of matrices, one per time period $t \in P$, is built, where entry (i, j) of the matrix associated with time t reports the number of times arc (i, j) has been traversed at time t in TS solutions. We note that it may happen that many entries are equal to 0. In this case, the corresponding arc variables are ordered according to the their closeness to the kernel by computing the following formula:

$$\texttt{closeness}(i, j) = c_{ai} + c_{ij} + c_{jb}, \tag{56}$$

   where (i, j) is the arc with zero value in the matrix while a and b ($a \neq b$) are the closest nodes to i and j belonging to $\mathcal{K}$.
2. At the end of the initialization phase, the best solution found is used in two ways:
   - Its objective function value z^{UB} is used as an upper bound to the solutions found by KSHFP.
   - The arcs traversed in this solution, $A(s^{UB})$, are inserted in the kernel, together with the arcs that connect the nodes visited, $N(s^{UB})$, to the depot.

The procedure is iterated until a stopping criterion is met. We refer the reader to [Archetti et al., 2018] for further details on this matheuristic.

Algorithm 2 Initialization

Input: Instance data

while *maximum number of iter is not met* **do** $\triangleright$ Adapted solution algorithm

    DP$\leftarrow$ `DPMILP`()

    $s \leftarrow$ `LK`(DP)

    $s^{UB} \leftarrow s$, $z^{UB} = f(s)$

    $(\tilde{s}, \texttt{VisitMatrix}) \leftarrow \texttt{TS}(s)$

    if $f(\tilde{s}) < z^{UB}$ **then**

        $z^{UB} = f(\tilde{s})$

        $s^{UB} \leftarrow \tilde{s}$

    end if

    `updateObjFunction`($\tilde{s}$)

end while

$\mathcal{K} \leftarrow A(s^{UB}) \cup \{(i, 0), (0, i) : i \in N(s^{UB})\}$

$\mathcal{B} \leftarrow$ getBucketList(`VisitMatrix`)

Output: $(s^{UB}, z^{UB}, \mathcal{K}, \mathcal{B})$ $\triangleright$ Best solution, solution cost, kernel, and bucket list

Finally, the resulting ordered bucket list $\mathcal{B}$ will be partitioned into N_b buckets of size L_B, according to the values of the variables in the frequency matrices. The original MILP, based on F3 (Section 4.3), is solved considering only the variables in $\mathcal{K}$ and the z^{UB} to compute the cut-off value.

5.1.2. Improvement phase

At each iteration $\ell > 0$ of the improvement phase, a bucket B_ℓ is merged with the current kernel $\mathcal{K}$ and an RMILP (the original MILP plus some considerations explained in (i)-(iii)) is solved on the merged set of variables. Depending on the incumbent status at termination, a new constraint is added to the next RMILP, i.e., to the RMILP solved at iteration $\ell + 1$. Let us call s^ℓ the solution obtained when solving RMILP at iteration ℓ. Then:

(i) $s^{UB} \leftarrow s^\ell$ and $z^{UB} = f(s^\ell)$, which is used to compute the cut-off value for the next iteration.

(ii) If incumbent s^{UB} is *optimal*, then any feasible solution s^ℓ must include at least one arc from the current bucket ℓ. Thus, we add the constraint:

$$\sum_{(a,b,h,t) \in B_\ell} \widetilde{y}_{ab}^{ht} \geq 1.$$

(iii) If incumbent s^{UB} is *feasible*, then any feasible solution s^ℓ must include at least one arc from B_ℓ or kernel $\mathcal{K}$ (not selected in s^ℓ). Thus, we add the constraint:

$$\sum_{(a,b,h,t) \in \mathcal{K} \setminus \{s^{UB}\}} \widetilde{y}_{ab}^{ht} + \sum_{(a,b,h,t) \in B_l} \widetilde{y}_{ab}^{ht} \geq 1.$$

The reason why we restrict the MILP through the former two constraints is to speed up and diversify the search during the optimization by forcing the selection of at least one arc different from the ones of the solution obtained in the current iteration. The arcs selected in the new solution that belong to the current bucket are included in the kernel, and the remaining arcs are discarded. KSHFP ends when all buckets are evaluated or when the time limit is reached.

6. Computational experience

Computational experiments were conducted on a Workstation HP Intel(R)-Xeon(R) at 3.5GHz with 64 GB RAM (Win 10 Pro, 64 bits) with a processor with 6 cores, and considering only one thread. The solution algorithms have been implemented in C++ with ILOG Concert Technology API (CPLEX 12.10).

The reminder of this section is organized as follows. In Section 6.1, we provide a description of the test instances. Section 6.2 summarizes the results of the tests performed to evaluate the proposed formulations, and to calibrate the main KSHFP parameters (kernel and buckets size). Finally, Section 6.3 describes the final performance comparison of the different solution algorithms. All the instances used for the tests as well as detailed preliminary and final results can be accessed at https://github.com/DianaHuertaM/HFPVRP.git.

For all instances, we measure the quality of the solution produced by a given method M, S^M, by computing the *Relative Percentage Deviation* (RPD) of its objective function value, $f(S^M)$, with respect to the objective function value of the best-known solution, $f(S^*)$, defined as

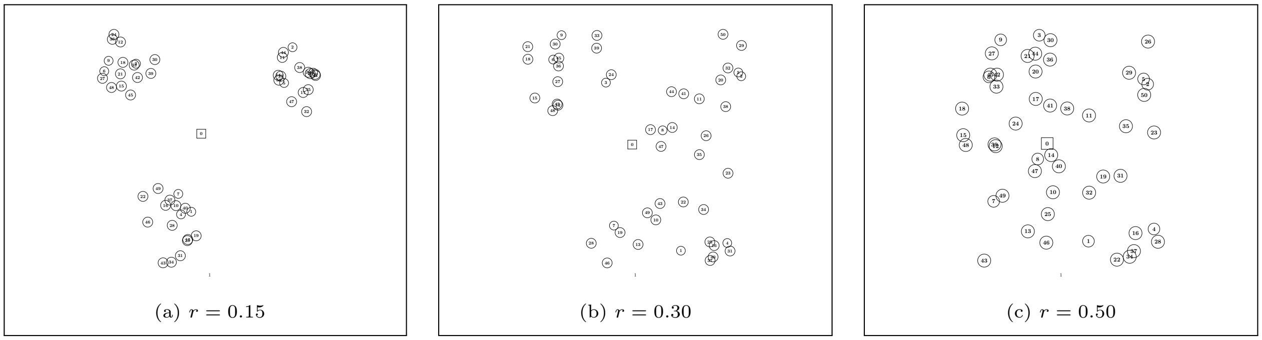
$$\text{RPD} = \frac{f(S^M) - f(S^*)}{f(S^*)} \times 100\%. \tag{57}$$

6.1. Benchmark instances

We generated two sets of HFVRP instances from the homogeneous FPVRP benchmark instances used in Archetti et al. [2017, 2018] (see Archetti et al. [2017] for a more detailed explanation about how they were generated for the homogeneous case):

- Calibration set: set of 30 instances, with $n \in \{10, 30, 50\}$ customers, time horizon of $|P| = 5$ days, and vehicle capacity of $Q = 300$. Customers are *clustered* in circular areas areas with radius $r = 0.50$, which determines the coverage area where customers are randomly located. When r is small (near zero) customers are clustered in small areas, while when r is close to one customers locations are more scattered (see Figure 1). There are 10 randomly generated instances for each value of n.

- Evaluation set: set of 90 instances, divided into small and large size. The small set consists of instances with $n \in \{10, 15, 20\}$, and radius $r \in \{0.15, 0.30, 0.50\}$). There are 5 randomly generated instances for each combination of n and r, having in total 45 instances. The vehicle capacity is set to $Q = \{200, 250, 300\}$ if $n = \{10, 15, 20\}$, respectively. The large set consists of instances with $n \in \{50, 75, 100\}$ and $r \in \{0.15, 0.30, 0.50\}$. There are 5 randomly generated instances for each combination of n and r, having in total 45 instances. The vehicle capacity is set to $Q = 500$. In both cases, small and large instances, the time horizon is $|P| = 5$ days.



(a) $r = 0.15$ (b) $r = 0.30$ (c) $r = 0.50$

Figure 1: Three 50-customer instances generated using different values of r.

In both sets:

- C, M and P are the same as in the homogeneous FPVRP instances, as well as all the parameters related to customers, i.e., W_i and w_i, $i \in C$.
- The number of vehicles m depends on capacities and demands, and varies between 4 and 21.
- There are two vehicle types, i.e., $V = \{1, 2\}$, defined as follows: If $k \leq m/2$, vehicle k is of type $h = 1$; If $k > m/2$, vehicle k is of type $h = 2$.
- Each vehicle type $h \in V$ is associated with a parameter f_h, which affects both the capacity of the vehicle, Q^h, as well as its routing costs $(c^h_{ij})_{(i,j)\in A}$. In particular, $Q^h = f_h Q$ and $c^h_{ij} = f_h c_{ij}$, where Q and $(c_{ij})_{(i,j)\in A}$ denote the capacity and routing costs of the original homogeneous instance, respectively. We set $f_1 = 0.8$ and $f_2 = 1.2$.

6.2. Calibration experiments

Preliminary experiments were performed to assess the effectiveness of the formulations presented in Section 4 and to calibrate the parameters of the KSHFP (kernel and bucket size). In all the preliminary experiments we used the calibration set.

6.2.1. Effectiveness of HFVRP formulations

In order to evaluate the performance of the HFVRP formulations presented in Section 4, all three formulations (F1, F2, and F3) were solved using CPLEX 12.10 with a time limit of 3600 sec.

The results obtained are summarized in Table 1. For each instance size (measured as its number of customers n), we report the following results for each tested formulation:

- The average Relative Percent Deviation (RPD%).
- The total number of feasible/optimal solutions found (#Feas/#Opt).
- The average MIP gap at termination (MIP-G%).

The last row of the table reports a summary of the results over all instances tested for which a feasible solution was found.

For instance sizes $n \leq 30$, all three formulations find at least a feasible solution. Still, only F3 produced a solution of proven optimality within the time limit. Moreover, for the instances with $n = 50$, only F3 produces a feasible solution in the time limit.

Note also that, for $n = 30$, F1 and F2 find just one feasible solution, with a very large MIP-G value. Thus, the results show a clear superiority of F3 as it is the formulation that produces the largest number of feasible/optimal solutions and yields the smallest values of MIP-G. Note that the number of feasible solutions found is an important factor to determine the best formulation to use in a heuristic approach, like the KSHFP.

Table 1: Comparison among models F1, F2, and F3

Instance size	F1			F2			F3		
	RPD%	**#Feas/#Opt**	**MIP-G%**	**RPD%**	**#Feas/#Opt**	**MIP-G%**	**RPD%**	**#Feas/#Opt**	**MIP-G%**
10	0.27	10/0	12.49	0.21	10/0	12.44	**0.05**	10/1	**2.22**
30	30.84	1/0	53.23	30.84	1/0	53.23	**0.00**	5/0	**10.09**
50	—	0/0	—	—	0/0	—	**0.00**	3/0	**20.07**
Summary	3.05	11/0	16.19	3.00	11/0	16.15	**0.03**	18/1	**7.38**

6.2.2. Calibration of kernel and bucket size

The second preliminary test focuses on finding the best combination for the kernel and bucket sizes used in the KSHFP. Below we indicate the alternatives that we considered.

- For the kernel size, we first point out that, during the initialization phase (see Section 5.1.1), we insert in the kernel the arc variables corresponding to the best solution found by the matheuristic plus the connection of the nodes to the depot. We then add additional variables to the kernel according to the following three strategies:
  - 33%$\mathcal{K}$**:** correspond to a kernel size of $|\mathcal{K}| = 33\%$ of the variables with positive values in visit matrices obtained in the initialization phase. The variables are selected according to a descending value of the term in the visit matrices.
  - 67%$\mathcal{K}$**:** a kernel size of $|\mathcal{K}| = 67\%$ of the variables with positive values in visit matrices obtained in the initialization phase.
  - $TS\mathcal{K}$**:** the initial kernel is composed of the arcs of the best solution of the TS plus the arcs that connect the visited nodes with the depot at each time period.
- For the bucket size, we test three different levels: $L_B = \{50, 100, 200\}$. Note that we take the first $\tilde{L}_B = \frac{L_B}{|V|}$ arcs from the bucket list, and replicate them by the number of vehicle types ($|V| = 2$). Therefore, the number of buckets will be $N_b = \left\lceil \frac{|\mathcal{B}|}{\tilde{L}_B} \right\rceil$.

By combining these two factors, we have 9 different parameter settings for the KSHFP. Table 2 shows the average RPD with respect to the best solution found among all options for each instance size. We observe that, on average, all kernel size options combined with the bucket size $L_B = 50$ obtain better results than the other combinations.

Table 2: Comparison among several options of kernel and bucket sizes

Instance size	Average RPD								
	$33\%\mathcal{K}$			$67\%\mathcal{K}$			$TS\mathcal{K}$		
	$L_B = 50$	$L_B = 100$	$L_B = 200$	$L_B = 50$	$L_B = 100$	$L_B = 200$	$L_B = 50$	$L_B = 100$	$L_B = 200$
10	0.11%	**0.01%**	0.07%	0.05%	0.09%	0.09%	0.50%	0.59%	0.22%
30	0.98%	1.27%	1.40%	2.98%	2.60%	2.60%	**0.27%**	0.71%	0.79%
50	0.94%	1.06%	0.88%	2.24%	2.92%	2.87%	**0.66%**	0.87%	1.18%
Total average	0.68%	0.78%	0.78%	1.76%	1.87%	1.85%	**0.48%**	0.73%	0.73%

In general, the best performance is given by the option $TS\mathcal{K}$ with $L_B = 50$, which are the values that will be used to evaluate the final performance of the KSHFP.

Further details on the calibration tests can be consulted at the repository provided above.

6.3. Evaluation experiments

Final tests have been carried out to assess the performance of the KSHFP, as calibrated above. We used the 90 instances of the evaluation set, with a time limit of 7200 seconds. For these tests we considered three alternative solution approaches:

(A) MathHFP: the adapted version of the matheuristic proposed by Archetti et al. [2018] (see Section 5.1.1).

(B) MILP-R: *Reduced-Formulation* F3. Since F3 cannot optimally solve the instances in the evaluation set, we solve F3 restricted to a smaller subset of arc variables y and l, which are chosen from the results of MathHFP, in a similar way as explained in Section 5.1.1. In particular, we consider the subset of $\widetilde{y}_{ij}^{ht}$ variables associated with arcs with a value greater than 0 in the visit matrices. Furthermore, the best solution obtained by the MathHFP is used as an initial solution (option MIPStart in CPLEX).

(C) KSHFP. The Kernel Search solution algorithm presented in Section 5 with the best parameters values from the calibration tests, i.e., initial kernel $TS\mathcal{K}$ with $L_B = 50$.

It is worth mentioning that (A), with a maximum computing time limited to a fraction of the total computing time allowed in (A), is used the initialization phase of both (B) and (C).

Table 3 shows the average performance of the three methods, per instance size. For each method, the Average RPD (ARPD) and the total computing times (Time) are presented. Detailed results per instance are provided in the Appendix.

We can observe that when instance size is small (up to size 20) method (B) outperforms, on average, the other methods in terms of solution quality. However, when instance size increases, better gaps are obtained with (C). In particular, for the largest instances with 100 customers, KSHFP provides an ARPD which is almost one third of the one provided in MILP-R. This shows that, exploiting the information provided by the initialization phase through a kernel scheme, i.e., by a repetitive solution of small-size MILPs, provides a more scalable and effective solution approach than the one in which a larger MILP, including the entire information, is solved at once, especially when the instance size grows. Moreover, KSHFP always outperforms MathHFP. We emphasize MathHFP is an adaptation to the heterogeneous case of a solution algorithm that was originally designed for the homogeneous case. However, MathHFP proved to be effective in solving the homogeneous FPVRP (see Archetti et al. [2018]). Thus, this proves that moving to the heterogeneous case is not straightforward and requires ad-hoc formulations and approaches.

Table 3: Comparison of the performance among solution methods - per instance size

n	**(A) MathHFP**		**(B) MILP-R**			**(C) KSHFP**	
	ARPD	**Time**	**ARPD**	**MIPGap**	**Time**	**ARPD**	**Time**
10	6.20%	42.58	0.02%	2.23	6574.03	0.88%	180.69
15	6.35%	106.14	0.07%	5.07	7199.52	0.54%	1031.79
20	5.20%	212.31	0.19%	4.10	7199.61	0.27%	2244.38
50	6.44%	2477.68	3.65%	16.30	7199.59	0.00%	7198.59
75	2.40%	6942.53	2.82%	14.97	7200.19	0.03%	7199.29
100	2.16%	8112.39	4.19%	15.26	7200.30	1.49%	7200.26
Total Avg	4.79%	2982.27	1.82%	9.66	7095.54	**0.54%**	**4175.83**

An ANOVA analysis showed significant differences between the performance of the different algorithms (p-values below 10^{-7}). Focusing on the differences between algorithms (A) and (C), we performed a t-test with paired data (instances). Again, the differences found are statistically significant: p-value 3×10^{-4} in the small set in favor of algorithm (A), and p-value 7×10^{-15} in the large set in favor of algorithm (B). For more detailed information about these analyses we refer the reader to the repository provided for the online material.

7. Conclusions

In this paper we introduced the Heterogeneous Flexible Periodic Vehicle Routing Problem. In this variant of the Periodic Vehicle Routing Problem, flexibility is introduced in service schedules and delivered quantities, and heterogeneity comes from different vehicles capacities and speeds. We proposed three mixed-integer linear programming formulations and developed a matheuristic based on Kernel Search to solve the problem efficiently.

Numerical results from computational experiments show that the best formulation is the one taking advantage of the aggregation over different vehicles' types. In terms of solution algorithms, a comparison between the KSHFP with two other heuristic approaches has been performed. The two approaches are obtained by adapting a former heuristic proposed for the homogeneous FPVRP and by solving the best formulation over a restricted and promising set of variables, respectively. The results show that KSHFP outperforms the competitors. Thus, this is a further successful application of KS for the solution of a complex routing problem. In addition, it shows that KS is a powerful solution approach which can beat other matheuristic schemes. In fact, the two heuristic approaches we compared the KSHFP with are indeed matheuristics.

Acknowledgments

The research of D.L. Huerta-Muñoz has been partially supported by the Mexican National Council for Science and Technology (CONACYT). The research of E. Fernández has been partially funded by the Spanish MINECO and ERDF funds through grant MTM2019-105824GB-I00. The research of F. Perea has been partially funded by the Spanish "Ministerio de Ciencia, Innovación y Universidades" through grants RTI2018-094940-B-100 and MTM2016-74983, and from the "Generalitat Valenciana" through grant AICO/2020/049. This support is gratefully acknowledged. The authors also wish to thank the anonymous reviewers whose comments helped improving a former version of the paper.

References

Abreu, R. C. and Arroyo, J. E. C. (2015). An application of ILS heuristic to periodic vehicle routing problem with heterogeneous fleet and fixed costs. In *2015 Latin American Computing Conference (CLEI)*, pages 1–9. IEEE.

Angelelli, E., Mansini, R., and Speranza, M. G. (2010). Kernel search: A general heuristic for the multi-dimensional knapsack problem. *Computers & Operations Research*, 37(11):2017–2026.

Angelelli, E., Mansini, R., and Speranza, M. G. (2012). Kernel search: A new heuristic framework for portfolio selection. *Computational Optimization and Applications*, 51(1):345–361.

Angelelli, E. and Speranza, M. (2002). The application of a vehicle routing model to a waste collection problem: Two case studies. In Klose, A., Speranza, M., and L.N., V. W., editors, *Quantitative Approaches to Distribution Logistics and Supply Chain Management. Lecture Notes in Economics and Mathematical Systems, vol 519*, pages 269–286. Springer, Berlin, Heildeberg.

Archetti, C., Bertazzi, L., Laporte, G., and Speranza, M. G. (2007). A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science*, 41(3):382–391.

Archetti, C., Fernández, E., and Huerta-Muñoz, D. L. (2017). The flexible periodic vehicle routing problem. *Computers and Operations Research*, 85:58–70.

Archetti, C., Fernández, E., and Huerta-Muñoz, D. L. (2018). A two-phase solution algorithm for the flexible periodic vehicle routing problem. *Computers and Operations Research*, 99:27–37.

Archetti, C., Guastaroba, G., Huerta-Muñoz, D. L., and Speranza, M. G. (2021). A kernel search for the multivehicle inventory routing problem. *International Transaction in Operational Research*, 1(1):1.

Archetti, C., Savelsbergh, M. W., and Speranza, M. G. (2008). To split or not to split: That is the question. *Transportation Research Part E: Logistics and Transportation Review*, 44(1):114–123.

Archetti, C. and Speranza, M. (2012). Vehicle routing problems with split deliveries. *International Transactions in Operational Research*, 30:3–22.

Archetti, C. and Speranza, M. G. (2016). The inventory routing problem: the value of integration. *International Transactions in Operational Research*, 23(3):393–407.

Baptista, S., Oliveira, R. C., and Zúquete, E. (2002). A period vehicle routing case study. *European Journal of Operational Research*, 139(2):220 – 229.

Beltrami, E. J. and Bodin, L. D. (1974). Networks and vehicle routing for municipal waste collection. *Networks*, 4(1):65–94.

Campbell, A. M. and Wilson, J. H. (2014). Forty years of periodic vehicle routing. *Networks*, 63(1):2–15.

Carvalho, D. M. and Nascimento, M. C. (2018). A kernel search to the multi-plant capacitated lot sizing problem with setup carry-over. *Computers & Operations Research*, 100:43–53.

Christofides, N. and Beasley, J. E. (1984). The period routing problem. *Networks*, 14(2):237–256.

Filippi, C., Guastaroba, G., and Speranza, M. G. (2016). A heuristic framework for the bi-objective enhanced index tracking problem. *Omega*, 65:122–137.

Francis, P. and Smilowitz, K. (2006). Modeling techniques for periodic vehicle routing problems. *Transportation Research Part B: Methodological*, 40(10):872–884.

Francis, P., Smilowitz, K., and Tzur, M. (2006). The period vehicle routing problem with service choice. *Transportation Science*, 40(4):439–454.

Francis, P. M., Smilowitz, K. R., and Tzur, M. (2007). Flexibility and complexity in periodic distribution problems. *Naval Research Logistics*, 2(54):136–150.

Francis, P. M., Smilowitz, K. R., and Tzur, M. (2008). The period vehicle routing problem and its extensions. *The Vehicle Routing Problem: Latest Advances and New Challenges*, 43:73–102.

Guastaroba, G. and Speranza, M. G. (2014). A heuristic for BILP problems: The single source capacitated facility location problem. *European Journal of Operational Research*, 238(2):438–450.

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

Lin, S. and Kernighan, B. (1973). An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21:498–516.

Miller, C., Zemlin, R., and Tucker, A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329.

Russell, R. and Igo, W. (1979). An assignment routing problem. *Networks*, 9(1):1–17.

Appendix A.

Tables A.4-A.5 summarize the results obtained on the evaluation set. For all methods, we present the best solution found (Sol), the overall computing time (Time), and the RPD with respect to the best-known solution. Also, for (B) we provide the MIP gap (MIP-G) and for (C) we report the iteration (IterKS) in which the KSHFP found the best solution. We observe that, on average, the best RPD is obtained by (B) for the small instances. However, for the large instances the best results are given by (C). (A) outperformed the other two methods only in very few instances.

Table A.4: Comparison of the performance among solution methods - Small instances

Instance	r	**(A) MathHFP**		**(B) MILP-R**			**(C) KSHFP**			**BEST**	**ARDP**		
		Sol	**Time**	**Sol**	**MIPGap**	**Time**	**Sol**	**Time**	**IterKS**		**(A)**	**(B)**	**(C)**
FPVRP_n10k5t5_1		20994	32.75	19679	2.63	7199.63	19645	442.62	8	19645	6.87%	0.17%	0.00%
FPVRP_n10k4t5_2		12835	40.94	12033	6.07	7199.80	12224	133.42	10	12033	6.67%	0.00%	1.59%
FPVRP_n10k5t5_3		13165	64.14	12277	1.06	7199.22	12425	204.62	1	12277	7.23%	0.00%	1.21%
FPVRP_n10k4t5_4		13528	55.28	12527	3.49	7199.42	12652	124.34	6	12527	7.99%	0.00%	1.00%
FPVRP_n10k8t5_5		26882	47.22	26211	0.01	3763.34	26486	65.20	3	26211	2.56%	0.00%	1.05%
FPVRP_n15k10t5_1		35853	129.35	34989	0.18	7199.51	34927	216.25	16	34927	2.65%	0.18%	0.00%
FPVRP_n15k6t5_2		18247	120.30	16974	5.34	7199.57	17011	984.98	7	16974	7.50%	0.00%	0.22%
FPVRP_n15k10t5_3	**0.15**	25906	113.65	24742	3.09	7199.21	24811	3014.37	12	24742	4.70%	0.00%	0.28%
FPVRP_n15k8t5_4		31835	90.71	30163	1.66	7199.28	30163	190.59	4	30163	5.54%	0.00%	0.00%
FPVRP_n15k7t5_5		25119	132.96	23215	5.59	7199.13	23627	320.42	4	23215	8.20%	0.00%	1.77%
FPVRP_n20k10t5_1		24611	225.92	23951	4.98	7199.70	23973	4624.56	19	23951	2.76%	0.00%	0.09%
FPVRP_n20k12t5_2		36001	240.72	34405	2.08	7199.64	34383	1956.89	16	34383	4.71%	0.06%	0.00%
FPVRP_n20k11t5_3		24186	213.29	23474	3.27	7199.60	23523	1420.80	36	23474	3.03%	0.00%	0.21%
FPVRP_n20k10t5_4		35952	238.31	33857	3.84	7199.31	33897	739.13	32	33857	6.19%	0.00%	0.12%
FPVRP_n20k10t5_5		30507	198.32	29171	4.06	7199.56	29108	3047.27	27	29108	4.81%	0.22%	0.00%
Avg			129.59			6970.39		1165.70			5.43%	**0.04%**	0.50%
FPVRP_n10k6t5_1		18865	33.11	18173	2.21	7199.39	18384	192.66	14	18173	3.81%	0.00%	1.16%
FPVRP_n10k6t5_2		14539	60.36	13778	3.61	7199.19	14062	150.67	2	13778	5.52%	0.00%	2.06%
FPVRP_n10k5t5_3		13411	41.75	13268	0.23	7199.16	13370	215.08	3	13268	1.08%	0.00%	0.77%
FPVRP_n10k5t5_4		14015	38.56	13444	1.06	7199.10	13435	509.35	16	13435	4.32%	0.07%	0.00%
FPVRP_n10k8t5_5		18935	60.91	18693	1.22	7199.08	18801	104.88	1	18693	1.29%	0.00%	0.58%
FPVRP_n15k9t5_1		27696	118.91	27094	2.31	7199.62	27361	342.25	0	27094	2.22%	0.00%	0.99%
FPVRP_n15k9t5_2		30711	122.34	29657	0.99	7199.53	29757	368.49	6	29657	3.55%	0.00%	0.34%
FPVRP_n15k7t5_3	**0.3**	26860	106.78	25293	1.61	7199.94	25413	249.23	6	25293	6.20%	0.00%	0.47%
FPVRP_n15k7t5_4		18909	111.75	17207	5.94	7199.92	17224	559.94	8	17207	9.89%	0.00%	0.10%
FPVRP_n15k6t5_5		21412	84.60	20191	5.52	7199.57	20032	6078.81	16	20032	6.89%	0.79%	0.00%
FPVRP_n20k10t5_1		29564	193.56	28287	4.03	7200.07	28179	1075.87	22	28179	4.92%	0.38%	0.00%
FPVRP_n20k12t5_2		32756	242.37	31561	3.99	7199.50	31651	1812.94	57	31561	3.79%	0.00%	0.29%
FPVRP_n20k10t5_3		27782	200.69	25450	5.48	7199.51	25560	7196.49	13	25450	9.16%	0.00%	0.43%
FPVRP_n20k13t5_4		44642	218.34	42966	1.02	7199.49	43261	415.05	16	42966	3.90%	0.00%	0.69%
FPVRP_n20k12t5_5		37948	180.36	35594	2.21	7199.83	35632	724.24	8	35594	6.61%	0.00%	0.11%
Avg			120.96			7199.53		1333.06			4.88%	**0.08%**	0.53%
FPVRP_n10k4t5_1		12449	25.36	11093	4.31	7199.14	11448	243.87	16	11093	12.22%	0.00%	3.20%
FPVRP_n10k4t5_2		14859	36.09	14186	2.97	7199.68	14186	83.20	2	14186	4.74%	0.00%	0.00%
FPVRP_n10k4t5_3		14035	31.60	12857	2.11	7199.04	12929	69.96	5	12857	9.16%	0.00%	0.56%
FPVRP_n10k5t5_4		14475	41.19	13751	2.44	7199.19	13751	105.57	1	13751	5.27%	0.00%	0.00%
FPVRP_n10k3t5_5		12813	29.50	11213	0.01	1256.09	11216	64.89	8	11213	14.27%	0.00%	0.03%
FPVRP_n15k4t5_1		15744	104.24	15072	11.16	7199.07	15238	462.63	7	15072	4.46%	0.00%	1.10%
FPVRP_n15k4t5_2		15362	68.41	13668	7.24	7199.69	13965	299.06	7	13668	12.39%	0.00%	2.17%
FPVRP_n15k4t5_3	**0.5**	16843	105.44	15428	8.08	7199.88	15427	1588.85	21	15427	9.18%	0.01%	0.00%
FPVRP_n15k5t5_4		19105	103.07	17820	7.07	7199.41	17915	460.71	11	17820	7.21%	0.00%	0.53%
FPVRP_n15k5t5_5		20090	79.61	19189	10.34	7199.50	19222	340.25	19	19189	4.70%	0.00%	0.17%
FPVRP_n20k14t5_1		33010	195.98	31668	1.41	7199.90	32042	316.70	8	31668	4.24%	0.00%	1.18%
FPVRP_n20k10t5_2		28936	214.71	28028	4.97	7199.30	27854	965.74	26	27854	3.88%	0.62%	0.00%
FPVRP_n20k7t5_3		25693	230.42	23677	7.61	7199.80	23761	836.09	19	23677	8.51%	0.00%	0.35%
FPVRP_n20k10t5_4		26127	209.56	24992	9.37	7199.46	24617	7199.20	27	24617	6.13%	1.52%	0.00%
FPVRP_n20k11t5_5		38314	182.11	36360	3.27	7199.54	36554	1334.77	22	36360	5.37%	0.00%	0.53%
Avg			110.49			6803.25		958.10			7.45%	0.14%	**0.66%**
Total Avg			120.35			6991.06		1152.29			5.92%	**0.09%**	0.56%

Table A.5: Comparison of the performance among solution methods - Large instances

Instance	r	**(A) MathHFP**		**(B) MILP-R**			**(C) KSHFP**			**BEST**	**ARDP**		
		Sol	**Time**	**Sol**	**MIPGap**	**Time**	**BestSol**	**Time**	**IterKS**		**(A)**	**(B)**	**(C)**
FPVRP_n50k9t5_1		41025	2001.19	38717	9.66	7199.21	38350	7193.90	24	38350	6.98%	0.96%	0.00%
FPVRP_n50k7t5_2		35893	5186.09	33061	10.20	7199.39	32209	7199.65	30	32209	11.44%	2.65%	0.00%
FPVRP_n50k8t5_3		34348	1461.10	32312	13.23	7199.43	31677	7200.00	28	31677	8.43%	2.00%	0.00%
FPVRP_n50k8t5_4		30685	1418.58	29238	12.38	7199.71	29110	7199.30	14	29110	5.41%	0.44%	0.00%
FPVRP_n50k10t5_5		34853	1685.13	33500	9.95	7199.99	32795	7199.79	18	32795	6.28%	2.15%	0.00%
FPVRP_n75k15t5_1		57211	4716.41	55307	8.15	7200.17	55539	7200.17	26	55307	3.44%	0.00%	0.42%
FPVRP_n75k14t5_2		65060	6537.57	66145	11.59	7200.20	64371	7200.01	19	64371	1.07%	2.76%	0.00%
FPVRP_n75k16t5_3	**0.15**	67618	6271.21	67682	8.55	7200.33	66929	7200.02	21	66929	1.03%	1.13%	0.00%
FPVRP_n75k17t5_4		68878	6559.63	69374	8.43	7200.20	68495	7197.08	26	68495	0.56%	1.28%	0.00%
FPVRP_n75k16t5_5		59640	6243.66	58132	10.40	7200.55	57188	7199.38	45	57188	4.29%	1.65%	0.00%
FPVRP_n100k17t5_1		71082	8567.81	71983	13.80	7200.49	71249	7200.54	4	71082	0.00%	1.27%	0.23%
FPVRP_n100k18t5_2		82172	8782.87	85758	14.40	7200.00	87301	7200.34	24	82172	0.00%	4.36%	6.24%
FPVRP_n100k20t5_3		84355	9509.24	84454	10.03	7200.59	83142	7199.88	0	83142	1.46%	1.58%	0.00%
FPVRP_n100k21t5_4		77235	7673.54	77746	10.76	7200.43	76135	7200.58	16	76135	1.44%	2.12%	0.00%
FPVRP_n100k18t5_5		95482	8606.43	97320	12.46	7200.06	92779	7199.95	20	92779	2.91%	4.89%	0.00%
Avg			5681.36			7200.05		7199.37			3.65%	1.95%	**0.46%**
FPVRP_n50k11t5_1		55931	2774.19	55820	16.21	7199.85	53708	7199.61	28	53708	4.14%	3.93%	0.00%
FPVRP_n50k10t5_2		46748	2020.69	46090	18.36	7199.34	43743	7191.26	36	43743	6.87%	5.37%	0.00%
FPVRP_n50k10t5_3		45543	2650.07	44850	16.15	7199.50	43363	7197.27	25	43363	5.03%	3.43%	0.00%
FPVRP_n50k9t5_4		44823	2718.34	44194	20.09	7199.81	41529	7200.06	34	41529	7.93%	6.42%	0.00%
FPVRP_n50k9t5_5		48867	3374.8	48582	21.66	7199.32	46460	7199.54	21	46460	5.18%	4.57%	0.00%
FPVRP_n75k14t5_1		72897	8386.44	73383	142.589	7199.94	71779	7199.89	21	71779	1.56%	2.23%	0.00%
FPVRP_n75k15t5_2		78519	7828.66	78867	148.121	7199.95	76739	7200.32	25	76739	2.32%	2.77%	0.00%
FPVRP_n75k14t5_3	**0.3**	62756	8275.07	63508	18.554	7200.25	60434	7200.46	0	60434	3.84%	5.09%	0.00%
FPVRP_n75k15t5_4		66483	6673.73	67347	15.509	7199.58	64629	7200.15	27	64629	2.87%	4.21%	0.00%
FPVRP_n75k13t5_5		69568	6498.55	69811	147.766	7200.05	67924	7200.07	29	67924	2.42%	2.78%	0.00%
FPVRP_n100k19t5_1		96829	9018.67	100806	164.347	7200.93	96974	7200.52	0	96829	0.00%	4.11%	0.15%
FPVRP_n100k19t5_2		92691	8200.12	92713	156.332	7200.06	90689	7200.61	0	90689	2.21%	2.23%	0.00%
FPVRP_n100k19t5_3		94205	7809.12	93618	142.037	7200.47	90350	7200.14	20	90350	4.27%	3.62%	0.00%
FPVRP_n100k18t5_4		85856	6675.18	102022	198.768	7200.67	98456	7200.72	11	85856	0.00%	18.83%	14.68%
FPVRP_n100k18t5_5		91477	10139	92865	14.012	7200.1	92482	7199.74	28	91477	0.00%	1.52%	1.10%
Avg			6202.84			7199.99		7199.36			4.74%	3.24%	**1.06%**
FPVRP_n50k10t5_1		47181	2509.49	46933	174.319	7199.53	44987	7200.22	26	44987	4.88%	4.33%	0.00%
FPVRP_n50k10t5_2		37819	1869.78	37814	210.177	7199.81	35863	7199.56	36	35863	5.45%	5.44%	0.00%
FPVRP_n50k9t5_3		42232	2004.08	41140	198.908	7199.54	38976	7199.62	15	38976	8.35%	5.55%	0.00%
FPVRP_n50k9t5_4		41194	2345.76	40506	215.979	7199.75	39244	7199.45	21	39244	4.97%	3.22%	0.00%
FPVRP_n50k12t5_5		53885	3145.94	53403	166.085	7199.7	51210	7199.59	31	51210	5.22%	4.28%	0.00%
FPVRP_n75k14t5_1		66064	8446.69	65576	193.711	7200.46	63395	7199.39	4	63395	4.21%	3.44%	0.00%
FPVRP_n75k17t5_2		75503	8537.11	76188	160.044	7200.45	74016	7200.32	25	74016	2.01%	2.93%	0.00%
FPVRP_n75k13t5_3	**0.5**	66507	7487.83	69202	226.235	7200.4	65627	7199.8	21	65627	1.34%	5.45%	0.00%
FPVRP_n75k15t5_4		65980	6463.61	66402	184.858	7200.4	64409	7200.15	16	64409	2.44%	3.09%	0.00%
FPVRP_n75k13t5_5		58512	5211.71	58968	231.139	7199.88	57004	7192.07	21	57004	2.65%	3.45%	0.00%
FPVRP_n100k18t5_1		84682	5545.47	82064	203.023	7200.02	77648	7200.19	0	77648	9.06%	5.69%	0.00%
FPVRP_n100k18t5_2		84628	8102.54	85654	182.664	7200.25	82850	7200.59	0	82850	2.15%	3.38%	0.00%
FPVRP_n100k18t5_3		87249	7299.19	87717	139.418	7199.93	85479	7200.3	13	85479	2.07%	2.62%	0.00%
FPVRP_n100k17t5_4		89738	7873.39	87510	149.605	7200.16	86014	7199.87	23	86014	4.33%	1.74%	0.00%
FPVRP_n100k18t5_5		83323	7883.31	85279	197.712	7200.31	81246	7199.89	0	81246	2.56%	4.96%	0.00%
Avg			5648.39			7200.04		7199.40			3.97%	4.11%	**0.00%**
Total Avg			7200.03			5844.20		7199.38			3.55%	3.67%	**0.51%**