applied sciences

MDPI

Article

A Memetic Algorithm for the Cumulative Capacitated Vehicle Routing Problem Including Priority Indexes

Samuel Nucamendi-Guillén †, Diego Flores-Díaz †, Elias Olivares-Benitez *,† and Abraham Mendoza †

Facultad de Ingenieria, Universidad Panamericana, Álvaro del Portillo 49, Zapopan, Jalisco 45010, Mexico; snucamendi@up.edu.mx (S.N.-G.); 0198749@up.edu.mx (D.F.-D.); amendoza@up.edu.mx (A.M.)

* Correspondence: eolivaresb@up.edu.mx

† These authors contributed equally to this work.

Received: 27 April 2020; Accepted: 1 June 2020; Published: 5 June 2020

Abstract: This paper studies the Cumulative Capacitated Vehicle Routing Problem, including Priority Indexes, a variant of the classical Capacitated Vehicle Routing Problem, which serves the customers according to a certain level of preference. This problem can be effectively implemented in commercial and public environments where customer service is essential, for instance, in the delivery of humanitarian aid or in waste collection systems. For this problem, we aim to minimize two objectives simultaneously, the total latency and the total tardiness of the system. A Mixed Integer formulation is developed and solved using the AUGMECON2 approach to obtain true efficient Pareto fronts. However, as expected, the use of commercial software was able to solve only small instances, up to 15 customers. Therefore, two versions of a Memetic Algorithm with Random Keys (MA-RK) were developed to solve the problem. The computational results show that both algorithms provided good solutions, although the second version obtained denser and higher quality Pareto fronts. Later, both algorithms were used to solve larger instances (20–100 customers). The results were mixed in terms of quality but, in general, the MA-RK v2 consistently outperforms the first version. The models and algorithms proposed in this research provide useful insights for the decision-making process and can be applied to solve a wide variety of business situations where economic, customer service, environmental, and social concerns are involved.

Keywords: open vehicle routing; integer programming; split deliveries; logistic distribution; mixed integer formulations

MSC: [2010] 90C11; 90B06

1. Introduction

In this paper, we study the biobjective variant of the Cumulative Capacitated Vehicle Routing Problem (CCVRP), a "customer-centric" variant of the classical Capacitated Vehicle Routing Problem (CVRP) [1] in which a fleet of k vehicles serves a set of customers by respecting their priority, defined as an index assigned to each node to indicate the preference to be served. Unlike the traditional VRP, which focuses on the impact that routing costs have on logistics and, in particular, in the transportation activities within the supply chain, the CCVRP rises as a particularization that covers broad objectives centered around service level issues. This problem is relevant in contexts where both customer satisfaction and company profits are prioritized. Due to the importance of sustainable business practices nowadays, there is a need to develop distribution strategies aimed at reducing the negative impact that transportation activities have on the environment. An important application can also be found in the context of emergency logistics, where the distribution of medical aid becomes crucial,

particularly when the distribution of different types of drugs or supplementary medical equipment have different levels of importance.

The CCVRP was introduced by Ngueveu et al. [2] as a generalization of the well known *k*-Traveling Repairmen Problem (*k*-TRP), with the objective of addressing problems in which customer metrics reflect the need for fast, equity and fair services. Since then, a diverse number of scenarios have been addressed: where a single vehicle can travel multiple trips [3,4], considering stochastic demand and split/unsplit deliveries [5], or when multiple depots are available [6]. Specifically, the CCVRP with priorities arises in the context of modeling situations in which customers have a differentiated level of attention and has attracted the interest of researchers over the past years due to its applicability in fields such as emergency logistics (i.e., level of damage), delivery logistics (i.e., delivery of different perishable items), personnel transportation, school bus transportation, machine scheduling considering due dates, maintenance operations including delivery dates, receiver wireless communications, etc.

As in the traditional VRP and its known variants, exact solution methods, heuristics, and metaheuristic algorithms are proposed to solve the CCVRP. Karagul et al. [7] pointed out that optimal solutions are possible to obtain in small-scale problems using exact solution methods. However, large-scale problems are difficult and time-consuming to solve to optimality. Hence, when it comes to real-life optimization, where problems are complex and deal with a significant amount of data, sometimes it is enough to find approximate solutions through the use of heuristic and metaheuristic methods.

Before the formal introduction of the problem in 2010, Kara et al. [8] developed a preliminary research studying a particular version of the CCVRP, named as CumVRP, in which the objective consists of minimizing the sumproduct of the arrival times and the demand of the node. Additionally, the CCVRP has been mainly studied from the monobjective function perspective. For this, several contributions involving mathematical models [9,10], exact algorithms [9,11], and heuristic and metaheuristic approaches [2,4,12–14] have been developed.

Previously, Sbihi and Eglese [15] dealt with the time-dependent routing problem (TDVRP) and applied this approach to green logistics. They established that the use of time-dependent models could obtain optimal solutions that produce less pollution by directing vehicles to roads where they can travel at faster speeds, which means away from congestion zones. However, this could represent traveling longer distances. Another benefit of this model is that it allows time window constraints to be satisfied more reliably. On the other hand, Kwon, Choi, and Lee [16] presented the heterogeneous vehicle routing problem that determines a set of vehicle routes that satisfies customer demands and vehicle capacities and minimize the sum of variable operating costs. An integer programming model was used, and for more complex problems, Tabu Search algorithms were developed to find solutions within a reasonable computational time. The results show that the amount of carbon emissions could be reduced by carbon trading without increasing the total costs.

Regarding the study of biobjective vehicle routing problems considering the customer-centric approach, few contributions can be found in the literature: the traveling repairman problem with profits (TRPP) [17], the *k*-Traveling Repairmen Problem with Profits (*k*-TRPP) under uncertainty [18], and a biobjective study of the minimum latency problem [19]. In the first work [17], the authors considered the *k*-TRP with the objective function that monotonically decreases over time. For this problem, they proposed a mathematical formulation. However, since the model should be solved separately for each path, the solution framework can be time-consuming, motivating the development of a Tabu-search procedure. The second research [18] addresses the *k*-TRP but enables the decision-maker to manage and control risk. For this, the authors proposed a mean-risk formulation and a heuristic approach based on an adaptive neighborhood search for the *k*-TRPP under uncertainty, providing high-quality results for small and medium-size instances. Finally, the most recent work [19] studies the Minimum Latency-Distance Problem (MLDP), analyzing the conflict between the total distance traveled by the vehicles and the total waiting time of the customers. The authors proposed a mathematical formulation and two heuristic procedures inspired on evolutionary algorithms, obtaining optimal

results for instances up to 40 customers and extending the computational experimentation over the metaheuristics for instances up to 256 customers.

Considering the aforementioned discussion, to the best of our knowledge, no tailored approach has been proposed in the literature for the problem considering the risk-averse perspective. For this reason, in this paper ,we study the Biobjective Cumulative Capacitated Vehicle Routing Problem considering Profits (BCCVRP-Pr). To handle this new problem, we propose a mixed-integer formulation and a MA-RK procedure to efficiently deal with instances of reasonable size.

Elshaer and Awad [20] studied the diversity of variants of vehicle routing problems solved using metaheuristics, where eleven papers were identified using memetic algorithms. In a recent example, Li et al. [21] proposed a hybrid metaheuristic that combines a memetic algorithm, sequential variable neighborhood descent, and a revised 2-opt method, for the plug-in hybrid electric vehicle routing problem. In addition, Zhang et al. [22] developed a multiobjective memetic algorithm for the vehicle routing problem with time windows. To our knowledge, only Ngueveu et al. [2] applied memetic algorithms to the cumulative capacitated vehicle routing problem. They created two versions of their procedures to solve a single objective problem to minimize the sum of arrival times at the customers.

On the other side, memetic algorithms have been combined with random keys in some applications. One example is the hybridization of He et al. [23], where memetic algorithms were combined with a biased random key genetic algorithm and adaptive large neighbourhood search to solve a scheduling problem. Other applications to scheduling problems using memetic algorithms and random keys were proposed by Li and Yin [24] and Ghrayeb and Damodaran [25], among others. Although this combination of memetic algorithms and random keys have been used to solve sequencing problems, such as in the traveling salesman problem [26], this combination has not been used to solve complex routing problems such as the one presented in this paper.

The literature described above shows the novelty of the combination of memetic algorithms and random keys to solve complex multiobjective routing problems.

The remainder of this paper is organized as follows. Section 2 describes the proposed mathematical formulation, whereas Section 3 presents the algorithm developed. Section 4 reports the results obtained from the experimentation using a set of benchmark instances. Finally, Section 5 summarizes the major findings and proposes future research directions.

2. Mathematical Formulation

In this section, we present the formal definition of the BCCVRP-Pr as well as its corresponding mathematical formulation. First, the following parameters and variables are considered:

Parameters

- N: Number of customers
- K: Number of vehicles available

The BCCVRP-Pr can be formally defined as an undirected graph $G = (V, A)$ where $V = \{0, 1, 2, ..., N\}$ denote the node set and A is the set containing all arcs. The node 0 corresponds to the depot and the rest of the nodes form the set of customers $V' = \{1, 2, ..., N\}$. Each arc $(i, j) \in A$ has an associated travel time c_{ij} and each node $j \in V'$ has an associated demand d_j. A heterogeneous fleet of K vehicles is available, each with capacity Q^k, $k \in \{1, 2, ..., K\}$. It is assumed that the vehicle's overall capacity is enough to satisfy the total demand of all the clients. In addition, to consider the priority, a precedence matrix P is defined in which $p_{ij} = 1$ represents that customer i should be serviced before customer j and $p_{ij} = 0$ means that customer i can be served after the customer j.

The following additional parameters are considered:

- Q_{max}: Maximum capacity of any of the vehicles
- M: Maximum travel distance allowed (the same for all vehicles)

The goal is to find k tours that have only node 0 in the first position, and to cover all the nodes, while minimizing the sum of the latencies of the trips. The demand of all customers must be satisfied without exceeding the capacity of each vehicle. Customers should be served (preferably) according to their priority level, minimizing the total tardiness of the system.

For each path, the tardiness arises when a customer with lower priority is served before a customer with higher priority (even if both customers belong to different routes). In other words, the arrival time (t_i) for a customer with a lower priority index is less than the arrival time of a customer with a higher priority index (t_j). Qualitatively, tardiness is associated with customer dissatisfaction. However, because latency is estimated as a function of distance, then tardiness is obtained as the difference between their arrival times (when $p_{ij} = 1$), $I_{ij} = t_j - t_i$. Hence, the total tardiness of the system is computed, as shown in Equation (1):

$$Total\ tardiness = \sum_{i \in V'} \sum_{j \in V'} I_{ij} \tag{1}$$

Figures 1 and 2 graphically exhibit the conflict among the values of latency and tardiness for an instance that contains 12 nodes. The number above each node denotes the customer priority index (higher values indicate higher priorities). In Figure 1, the minimum total tardiness was obtained subject to the minimum total latency. Correspondingly, Figure 2 indicates the optimal total latency while assuring the minimum total tardiness.

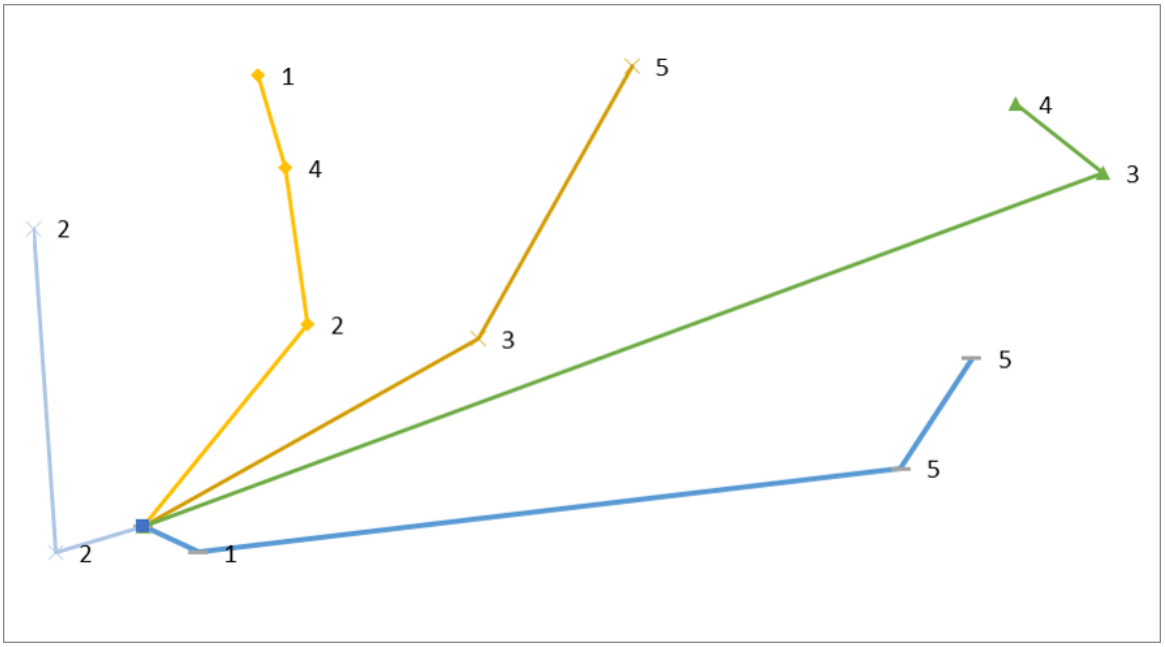

Figure 1. Solution routes minimizing the total latency: total latency = 1595.31; total tardiness = 4304.55.

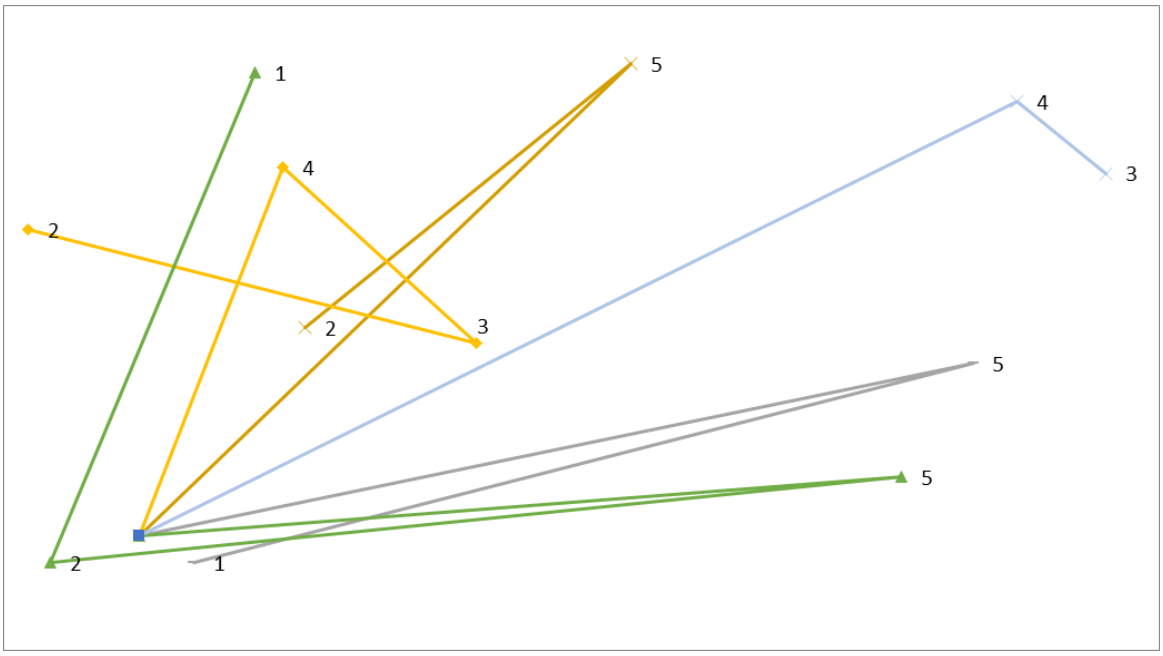

Figure 2. Solution routes minimizing the total tardiness: total latency = 2972.84; total tardiness = 0.

The formulation is based on the model presented in [10] for the classical CCVRP and following the single flow formulation proposed for the multiple traveling salesman problem [27].

Variables

$$w_{0j}^k = \begin{cases} 1, & \text{if the vehicle } k \text{ is assigned to a path from node 0 to customer } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{if the arc } (i,j) \text{ is in the path of a vehicle,} \\ 0, & \text{otherwise.} \end{cases}$$

y_{ij} = number of remaining nodes after the node i on a route if $x_{ij} = 1$; 0, otherwise.

v_{0j}^k = the sum of demands of the nodes after node 0 on the route k if $w_{0j}^k = 1$; 0, otherwise.

v_{ij} = sum of demands of the nodes after node i on a route if $x_{ij} = 1$; 0, otherwise.

t_{0j}^k = Arrival time of node j from node 0 on a route k if $w_{0j}^k = 1$; 0, otherwise.

t_{ij} = Cumulative time at node j on a route if $x_{ij} = 1$; 0, otherwise.

I_{ij} = Tardiness in the arrival time to node i if node j is served first$(p_{ij} = 1)$; 0, otherwise.

The corresponding mathematical model is stated as follows:

$$min\ F_1 = L = \sum_{i \in V'} c_{0i} y_{0i} + \sum_{i \in V'} \sum_{\substack{j \in V' \\ j \neq i}} c_{ij} y_{ij}, \tag{2}$$

$$min\ F_2 = T = \sum_{i \in V'} \sum_{\substack{j \in V' \\ j \neq i}} I_{ij}, \tag{3}$$

subject to:

$$\sum_{j \in V'} \sum_{k \in K} w_{0j}^k = K, \tag{4}$$

$$\sum_{i \in V'} x_{i0} = K, \tag{5}$$

$$\sum_{j \in V'} \sum_{k \in K} w_{0j}^k = 1, \qquad \forall k \in K, \tag{6}$$

$$\sum_{k \in K} w_{0j}^k + \sum_{i \in V'} x_{ij} = 1, \qquad \forall j \in V', \tag{7}$$

$$\sum_{j \in V} x_{ij} = 1, \qquad \forall i \in V', \tag{8}$$

$$y_{0j} + \sum_{\substack{i \in V' \\ i \neq j}} y_{ij} - \sum_{\substack{i \in V' \\ i \neq j}} y_{ji} = 1, \qquad \forall j \in V', \tag{9}$$

$$y_{0j} \geq \sum_{k \in K} w_{0j}^k, \qquad \forall j \in V', \tag{10}$$

$$y_{ij} \geq x_{ij}, \qquad \forall i \in V'; \forall j \in V'; i \neq j, \tag{11}$$

$$y_{0j} \leq (N - K + 1) \sum_{k \in K} w_{0j}^k, \qquad \forall j \in V', \tag{12}$$

$$y_{ij} \leq (N - K) \sum_{k \in K} x_{ij}, \qquad \forall i \in V'; \forall j \in V', \tag{13}$$

$$v_{0j}^k \geq d_j w_{0j}^k, \qquad \forall j \in V'; \forall k \in K, \tag{14}$$

$$v_{0j}^k \leq Q^k w_{0j}^K, \qquad \forall j \in V'; k \in K, \tag{15}$$

$$v_{ij} \geq d_j x_{ij}, \qquad \forall i \in V'; j \in V'; i \neq j, \tag{16}$$

$$v_{ij} \leq (Q_{max} - d_i) x_{ij}, \qquad \forall i \in V'; j \in V; i \neq j, \tag{17}$$

$$\sum_{k \in K} v_{0j}^k + \sum_{\substack{i \in V' \\ i \neq j}} v_{ij} - \sum_{\substack{i \in V \\ i \neq j}} v_{ji} = d_j, \qquad \forall j \in V', \tag{18}$$

$$t_{0j}^k = c_{0j} w_{0j}^k, \qquad \forall j \in V'; j \in V'; k \in K, \tag{19}$$

$$t_{0j}^k \leq M w_{0j}^k, \qquad \forall i \in V'; j \in V'; i \neq j; k \in K, \tag{20}$$

$$t_{ij} \geq c_{ij} x_{ij}, \qquad \forall i \in V'; j \in V'; i \neq j, \tag{21}$$

$$t_{ij} \leq (M - c_{ij}) x_{ij}, \qquad \forall i \in V'; j \in V; i \neq j, \tag{22}$$

$$\sum_{\substack{h \in V \\ h \neq i}} t_{ih} - \sum_{\substack{h \in V' \\ h \neq i}} t_{hi} - \sum_{k \in K} t_{0i}^k = \sum_{\substack{j \in V \\ j \neq i}} c_{ij} x_{ij}, \qquad \forall i \in V', \tag{23}$$

$$I_{ij} \geq p_{ij} \left(\sum_{k \in K} t_{0i}^k + \sum_{\substack{h \in V' \\ h \neq i}} t_{hi} - \sum_{\substack{h \in V' \\ h \neq j}} t_{hj} - \sum_{k \in K} t_{0j}^k \right), \qquad \forall i \in V'; j \in V'; i \neq j, \tag{24}$$

$$w_{0j}^{k} \in \{0,1\}, \quad \forall j \in V'; \forall k \in K, \tag{25}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in V'; \forall j \in V, \tag{26}$$

$$y_{ij} \geq 0, \quad \forall i \in V; \forall j \in V, \tag{27}$$

$$v_{0j}^{k} \geq 0, \quad \forall j \in V'; k \in K, \tag{28}$$

$$v_{ij} \geq 0, \quad \forall i \in V'; j \in V, \tag{29}$$

$$t_{0j}^{k} \geq 0, \quad \forall j \in V'; k \in K, \tag{30}$$

$$t_{ij} \geq 0, \quad \forall i \in V'; j \in V, \tag{31}$$

$$I_{ij} \geq 0, \quad \forall i \in V'; j \in V'. \tag{32}$$

In this formulation, the objective functions in Equations (2) and (3) seek a trade-off between the objective of the total latency and the total tardiness of the system. The constraints in Equations (4) and (5) ensure that exactly K arcs leave and return to the depot. The constraints in Equation (6) establish that a customer node can be visited for exactly one vehicle coming from the depot. Additionally, the constraints in Equations (7) and (8) impose that exactly one arc enters and leaves for each node associated with a customer. The constraints in Equations (10)–(13) help to avoid sub-tours and allow to calculate the position of the customer j on its respective route. The constraints in Equations (12) and (13) force variables y_{0j} and y_{ij} to be zero when $w_{0j}^{k} = 0$ and $x_{ij} = 0$, respectively. Regarding the capacity of the vehicles, the constraints in Equations (14) and (15) allow establishing lower and upper bounds for the cumulative demand of any route. These constraints are derived from a generalization of restrictions in Equations (10) and (11). The constraints in Equations (16) and (17) force variables v_{0j}^{k} and v_{ij} to be zero when variables $w_{0j}^{k} = 0$ and $x_{ij} = 0$, respectively. The constraints in Equation (18) ensure that the demand at each node i is fulfilled and in conjunction with Equations (16) and (17) estimate the load per vehicle. The constraints in Equations (19)–(23) control the arrival time to the nodes (customers). The constraints in Equation (24) estimate the tardiness (in case of violating the priority constraints). Finally, the constraints in Equations (25)–(32) establish the nature of the variables.

Reformulation Using Epsilon Constraint

In this subsection, we describe the characteristics of the model and the proposed reformulation. A fundamental task in multiobjective optimization is to find Pareto-optimal solutions. As a biobjective approach, we decided to implement a multiobjective method of resolution to generate an exact front of efficient solutions.

In the mathematical model, we note that the objective functions are separable. In other words, each of them involves different decision variables. On the one hand, y_{ij} allows estimating the total arrival time to the customers. On the other hand, I_{ij} computes the total tardiness for the case in which customers having a minor priority are served earlier than customers with higher priority index (regardless if they are located in the same route or belong to different routes).

To clarify this, the particular structure of the biobjective problem proposed herein is described below:

$$\min F_1 = L = \sum_{i \in V'} c_{0i} y_{0i} + \sum_{i \in V'} \sum_{\substack{j \in V' \\ j \neq i}} c_{ij} y_{ij} \tag{33}$$

$$\min F_2 = T = \sum_{i \in V'} \sum_{\substack{j \in V' \\ j \neq i}} I_{ij} \tag{34}$$

$$\text{subject to :}$$

$$Equations (4)–(24)$$

The aforementioned characteristics of the biobjective model are exploited by using an improved version of the ε-constraint method, named as AUGMECON2 [28], as a solution procedure. For every single routing decision in Equations (4)–(18), the minimum tardiness (min T) problem bounded by the constraints in Equations (19)–(24) is solved as principal objective function, transforming the latency function (L) into constraint. The results of the proposed method are shown in Section 4.

3. Metaheuristic Algorithm

This section is devoted to describe the metaheuristic approach capable of obtaining high quality solutions for small instances and able to deal with large size instances. The proposed method is based on a Memetic Algorithm (MA), a population procedure that has shown its effectiveness in solving sizeable combinatorial optimization problems by incorporating a local search procedure within a classical genetic algorithm. This procedure has been successfully applied for addressing the CCVRP [2], introducing efficient move evaluation procedures in operations $O(1)$ for some particular neighborhood structures. MAs have been also employed in solving other routing problems such as split delivery vehicle routing problems [29], capacitated location routing problems [30,31], vehicle routing problems with time windows [32], school bus routing problems [33], and green and healthcare routing problems [34,35].

3.1. Proposed Memetic Algorithm

Holland [36] was the first to propose Genetic Algorithms (GAs) inspired on ideas of evolution theory. Due to their simple and yet effective search procedure, several papers (e.g., [37]) describe their successful implementations in vehicle routing problems. In particular, Memetic algorithms (MAs) belong to the class of evolutionary algorithms that intensify the search by including local search within a classical genetic algorithm framework. According to Moscato and Cota [38], MAs are intrinsically concerned with exploiting all available knowledge about the problem under study. Due to this, a random key mechanism is included during the construction procedure in order to enhance the performance of the procedure. In this work, the MA proposed is adapted from the NSGA-II, successfully implemented in [39], and consists of the following procedures: initialization, recombination (crossover and local search), and classification in fronts. In our construction procedure, we include a random key that helps to generate a diverse initial population of feasible solutions of size N. Subsequently, during a predetermined number of successive generations (iterations), an offspring (P_t) of N individuals is generated from P_{t-1} through recombination and local search mechanisms, involving members representing tentative solutions (high-quality or non-dominated solutions) and members representing diverse solutions. After this, the individuals who belong to the previous and current generations are evaluated and grouped into fronts, according to the level of non-domination, as explained in [40]. To obtain the resulting offspring population of size N, the individuals are inserted into the set, starting with the one that belongs to the front of non-dominated solutions (F_0). Algorithm 1 shows the pseudocode of the overall MA with Random Keys procedure.

3.1.1. Constructive Procedure Based on Random Keys

The constructive procedure creates an initial population of feasible solutions based on generating a chain $S_a = \{1, 2, \cdots, n\}$ and, for each customer, an auxiliary random key R_a is used to encode the solution. Additionally, an empty set S_p is used to save the temporary assignment of the customers to the routes.

Algorithm 1: Memetic algorithm with random keys.

```
begin
    it ← 0;
    Initialize a population (P_0) of σ chromosomes implementing the constructive procedure
      based on random keys;
    Sort P_0 in fronts following the non-dominated sorting approach;
    for (it = 1; it <= Maxiter; it + +) do
        Generate an offspring population P_it, of size N, from P_it−1, using selection, crossover
          and local-search operators;
        Combine parent and offspring population R_it = P_it ∩ P_it−1;
        Sort population using the non-dominated sorting approach, identify fronts
          F_j = (1, 2, ···), and calculate the crowding distance for each solution in F_j;
        Make T_it+1 ← ∅, j ← 1;
        while (|T_it+1| + |F_j| < N) do
            T_it+1 ← T_it+1 + F_j ;
            j ← j + 1;
            Sort solutions in F_j in decreasing order according to crowding distances, select the
              first N − |T_it+1| elements of F_j and add it to T_it+1;
        end
    end
end
```

Encoding mechanism: To encode the solution, a real number drawn randomly from [0, 1) is assigned to every single position in R_a. Figure 3 depicts an example of this mechanism.

Sa	1	2	3	4	5	6	7	8	9	10
Ra	0.50	0.60	0.58	0.21	0.61	0.05	0.96	0.26	0.76	0.15

Figure 3. Example of the encoding mechanism.

Decoding mechanism: The decoding mechanism is applied based on the information of the random key R_a. The R_a chain is sorted in a non-decreasing order and their respective positions in S_a are sorted correspondingly. As a result, a random ordered chain S_a is obtained. Figure 4 exhibits the decoding mechanism.

Sa	6	10	4	8	1	3	2	5	9	7
Ra	0.05	0.15	0.21	0.26	0.50	0.58	0.60	0.61	0.76	0.96

Figure 4. Example of the decoding mechanism.

Assignment mechanism: In every iteration, the algorithm selects the corresponding jth customer in S_a and systematically tries to insert it into into a temporary set S_p in the first available position (procuring to maintain feasibility in the capacity of the vehicles). For instance, if in the first potential route, two customers have been previously assigned, the next open position will be the third one. In the case that the customer cannot be inserted in the route due to the capacity constraints, the insertion will be evaluated in the next available route. It is important to emphasize that, since the number of routes is given in advance, the construction procedure considers a parallel routing mechanism. In other words, it performs the evaluation of feasible insertions over all of the routes. If so, the algorithm continues by selecting the next customer at S_a. Otherwise, the construction mechanism stops. If the algorithm reaches a feasible assignment, then $S \leftarrow S_p$ and the solution is inserted into the population

Q_t. Otherwise, the algorithm destroys the partial constructed solution in S_p and generates a new random key R_a (to sort R_a).

The entire constructive procedure finishes when all of the customers have been assigned into S_p or after having a successive number of attempts without producing a feasible solution. When a feasible assignment is reached, the set S represents a feasible initial solution of routes. The customers already inserted in S are removed from S_a. Figure 5 shows an illustrative example of a feasible assignment.

Route 1	**6**	**10**	**3**	
Route 2	**4**	**8**	**1**	**7**
Route 3	**2**	**5**	**9**	

Figure 5. Example of a feasible assignment.

After reaching a feasible assignment, the sequencing mechanism is applied to construct each route by respecting the order of insertion and, based on this, the corresponding calculations of the objective functions L (representing the total latency of the system) and T (total tardiness of the system, based on the priorities of the customers) are performed. Algorithm 2 shows the pseudocode for this algorithm.

Algorithm 2: Constructive procedure (S, L, D).

```
begin
    Data: S ← ∅, S_p ← ∅, S_a =← ∅, L ← 0, T ← 0
    Fill chain in S_a = {1, 2, ··· , n};
    Create an auxiliary random key chain R_a with values [0, 1);
    Sort customers in S_a in a non-decreasing order according with their corresponding random
     values in R_a;
    j = 0;
    while (S_a ≠ ∅) do
        flag = 0;
        while S_a ≠ ∅ or flag = 1 do
            Select the jth customer from S_a;
            l = 0;
            if The insertion of the jth customer is feasible to insert in route l then
                Insert the jth customer in the lth route S_p;
                Remove the assigned customer from S_a;
                j++;
                flag = 0;
            end
            else if l < K then
                l++;
            end
            else
                Destroy the partial solution, S_p ← ∅;
                Establish S_a = {1, 2, ··· , n};
                Create a new random key R_a ;
                flag = 1 ;
            end
        end
        Compute the values of total latency L and total tardiness T for the individual;
    end
end
```

3.1.2. Crossover Procedure with Local Search Strategies

The proposed crossover procedure consists of the combination of two solutions, A and B, to create a new solution C. A tournament selection operator is incorporated to diversify the creation of new solutions. After obtaining the new individual C, a selective local search mechanism can be applied to improve it.

The procedure receives the following inputs: the current population (Q), the updated population (R), and the number of children to create (N). Then, the mechanism starts by selecting two individuals from the current generation (P_t). The first individual will belong to the front F_0, whereas the second one will be chosen at random from the entire population (generation). Subsequently, the customers of the routes for each solution are grouped into a single big chain by following the assignment order (starting from the first customer belonging to the first route and ending at the last customer of the latter one). As a result, the chains for the corresponding chromosomes A and B are obtained.

The creation of the new individual C is based on the information of the random keys (R_a) of each parent (chromosome). Then, a probability for inheriting is assigned to each parent. These probabilities are complementary. In other words, if the probability of $P_A = \alpha$ is assigned for the first chromosome (A), then the second chromosomes will receive a probability of $P_B = 1 - \alpha$. Then, for every position to fill in the R_a belonging to the customer, the roulette wheel is spun (RN) to determine if the element belonging to the R_a for the first or second parent must be selected to insert in the child. If the value of $RN \leq P_A$, the ith element R_a is included in the child. Otherwise, the element belonging to the second parent is selected. The mechanism stops when all of the positions have been evaluated. Figure 6 illustrates and example of this mechanism.

Parent A	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
P_A=0.6	0.50	0.60	0.58	0.21	0.61	0.05	0.96	0.26	0.76	0.15
RN	**0.45**	**0.69**	**0.39**	**0.44**	**0.38**	**0.46**	**0.33**	**0.65**	**0.23**	**0.27**
Child RK	**0.50**	**0.43**	**0.58**	**0.21**	**0.61**	**0.05**	**0.96**	**0.60**	**0.76**	**0.15**
P_b=0.6	**0.72**	**0.43**	**0.73**	**0.77**	**0.08**	**0.68**	**0.25**	**0.60**	**0.62**	**0.69**
Parent B	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**

Figure 6. Example of child generation based on the roulette-tournament mechanism.

To enhance the creation of reasonable quality solutions, the probability assigned to the parent belonging to F_0 is always greater than 50%. Since the decoding procedure is a simple mechanism, it might occur that different random keys lead to an identical solution. Once the *Ra* for a child solution C is obtained, its feasibility is evaluated by calling back the constructive procedure. If the resulting solution is feasible, then the total latency L_C and tardiness T_C objectives are computed. On the contrary, the child is discarded, and a new second parent is selected (preserving the first individual) from P_t to restart the crossover procedure. Algorithm 3 depicts the pseudocode of the crossover mechanism.

Algorithm 3: Crossover procedure.

```
begin
    Data: Q, R
    for (h = 1; h <= N; h++) do
        Select randomly the first parent (A) from F_0 ∈ Q;
        Select randomly the second parent (B) from Q;
        Construct the big chain for each selected solution;
        i = 0;
        repeat
            Spin the wheel to obtain the value of probability;
            if probability ≤ β and customer in A is available then
                Select the customer of the chromosome A;
            end
            else if probability ≤ β and customer in B is available then
                Select the customer of chromosome B;
            end
            else
                i++;
            end
        until All positions in both parents have been evaluated;
    end
    if feasible then
        Decode the corresponding solution for the new individual;
        Compute the Latency (L) and Tardiness (T) values;
        Spin the roulette to obtain a rand number;
        if rand ≤ threshold then
            Apply the local search procedure over the individual C;
        end
        Insert the new solution C in R.;
    end
end
```

3.1.3. Local Search (LS) Procedure

The LS procedure is based on local search strategies, applied to intensify the search in pursuit of finding local minima. This procedure consists of five different neighborhood structures arranged into two classes, namely intra-route and inter-route mechanisms, performing them iteratively. This strategy has proved to be successful for a mono-objective version of the CCVRP [10]. Below, we describe each type of move:

- *Intra-route swap.* The procedure exchanges the positions of two customers belonging to the same route. For instance, if the customers to exchange belong to positions h and i, then arcs $(h-1,h)$, $(h,h+1)$, $(i-1,i)$ and $(i,i+1)$ are removed and replaced by arcs $(h-1,i)$, $(i,h+1)$, $(i-1,h)$ and $(h,i+1)$. It is important to remark that these movements do not affect feasibility in terms of capacity.
- *Intra-route reallocation.* This mechanism deletes a customer from its current position and reinserts it into another position on the same route.
- *Intra-route 2-opt.* In this operator, two non-adjacent edges $(h,h+1)$ and $(i,i+1)$ in the path $0,1,2,\ldots,h,h+1,\ldots,i,i+1,\ldots$ are deleted and replaced by (i,h) and $(h+1,i+1)$, resulting in the new path $0,1,2,\ldots,h,i,\ldots,h+1,i+1,\ldots$

- *Inter-routes interchange.* This strategy exchanges two customers belonging to different routes, as long as the move keeps feasibility (in terms of capacity).
- *Inter-routes reallocation.* For a given customer, the operator searches for the best position of the customer to move in any of the routes. If the best-identified position is different from the current one, the movement is performed.

The two major strategies operate as follows: At first, the initial solutions are sent to the intra-local search procedure, where the intra-local search strategies are applied. Then, the local minimum is forwarded to perform inter-routes local search strategies. These procedures are iteratively implemented, while the current solution value L keeps improving. In each process, the reallocation movement is performed first, and the execution of the interchange movement next. The first improvement criterion (*FI*) is used. Algorithm 4 exhibits this process.

Algorithm 4: Local search (S, L, T).

```
begin
    repeat
        S* = S and L* = L, and T* = T;
        S',L' T' ← Intra-route local search (S,L, T);
        if The solution is non-dominated then
            S = S' and L = L', T* = T;
        end
        S",L" T"← Inter-routes local search (S,L, T);
        if The solution is non dominated then
            S = S" and L = L", T* = T;
        end
    until L* > L;
    return S*, L*, T*;
end
```

As observed, this mutation procedure seeks to insert improved individuals to the next generation, although the mechanism does not guarantee that the chromosome selected can be deeply improved. Because the size of this subset is relatively small, it is always possible to find a chromosome to be improved.

This procedure differentiates the two versions of the MA. For the first version (MA-RK v1), all of the feasible individuals generated by the crossover mechanism are sent to the LS procedure ($threshold = 1$). For the second version (MA-RK v2), the local search mechanism is applied only to a certain percentage of individuals, expecting to accelerate the performance of the algorithm in terms of CPU time.

4. Computational Results

This section is devoted to reporting the computational experiments conducted to assess the efficiency of the proposed approach. First, we provide the set of instances used to perform the tests, as well as the characteristics of the computational equipment used. Secondly, we present the parameter setting for our versions of the MA. Finally, the experimental results for both the mathematical formulation and the metaheuristic procedure are displayed, accompanied by the respective discussion.

4.1. Test Instances

The instances used to conduct the experimentation were adapted from the ones used in the literature to evaluate the multi depot VRP with heterogeneous fleet: Koulaeian et al. [41] (Kou15), Chunyu and Xiaobo [42] (CaX10), Gillett and Jhonson [43] (GaJ76-7–GaJ76-12), and Augerat et al. [44] (Pn16k8 and Pn23k8). Even though there are some instances proposed by Talliard [45] and Li et al. [46] for the classical Heterogeneous Fleet Vehicle Routing Problem, we decided to use these instances since some of them provide a reasonable size in the number of customers for assessing the model.

The size of the instances ranges from 12 to 100 nodes and from 2 to 10 vehicles. The generated problem instances are characterized by the following criteria: (i) number of customers; (ii) number of vehicles; (iii) coordinates (x,y) for all locations (including the depot); (iv) demand of each node; and (v) priority index for each node. Since the original instances consider multiple depots and do not consider any preference index between the customers to be served, we selected the first depot as the single-origin, and we included a priority parameter by assigning a numerical index within $\{1, \lceil \sqrt{n} \rceil\}$, based on a uniform distribution. The customers having the highest value of the index represent the ones that should be first served (highest level of priority).

Since the modified instances are set to deal with a different problem, and to facilitate the report of the results, we decided to rename them using the nomenclature "FNO-x", where x denotes a consecutive number assigned according to the rank assigned to the instance (in terms of the number of nodes, following a non-increasing order). For example, the instance Kou15 is the one with the lowest number of customers (12); therefore, it was renamed as FNO1. The remaining instances based on Pn16k8, CaX10, Pn23k8, GaJ76-7, GaJ76-8, GaJ76-9, GaJ76-11, and GaJ76-12 were renamed as FNO2, FNO3, FNO4, FNO5, FNO6, FNO7, FNO8, and FNO9 respectively. In the case of the instance GaJ76-10, it was named FNO10 because it has the largest size in the number of customers. These new instances are available by request.

All of the experiments were conducted using a PC Intel Core i7 @2.30 GHz with 16 GB of RAM Memory under Windows 10. The formulation was modeled using AMPL and solved using Gurobi 9.0. For each instance, we established a time limit of 7200 s (2 h). In the case of the MA-RK, both versions were coded using the C++ language. In the next subsection, the parameter tuning is presented.

4.2. Parameters Setting

In the case of the MA-RK v1, the values for the parameters corresponding to the size of the population N, the threshold value β, and the maximum number of generations D were set as 1000, 0.1, and 100, respectively. In the case of MA-RK v2, we set a threshold of 0.4. These values were obtained after performing a preliminary analysis over a subset of instances randomly selected. In addition, several tests with different number of iterations were conducted, finding that, for all the analyzed instances, the MA-RK stops improving when it reaches 80% of the maximum number of iterations, depending on the instance. Lastly, to evaluate consistency, each instance was executed 10 times, and the best front obtained is reported. The next sections show the numerical results computed over the test instances.

Experimental Results

The first set of experiments aims at evaluating the performance of the formulation concerning optimality (optimally solved instances), and the effectiveness of the MA-RK comparing the results with those obtained by the resolution of the model. We first present the results for Instances FNO1 and FNO2 (up to 15 nodes).

The following metrics were used to compare the performance of the exact and approximation procedures:

- Number of points on the front (the larger, the better)
- CPU time (in seconds) (the shorter, the better)
- k-distance [47] (the smaller, the better)

- Hypervolume [48] (the larger, the better)
- The coverage of the fronts [48] computed of one front over another, denoted by c(X′, X″) (the higher, the better)

These metrics have shown their successful implementation in biobjective VRPs [19,39].

The number of non-dominated points measures the ability of each method to find efficient fronts. Table 1 summarizes these results for Instances FNO1 and FNO2. Figures 7 and 8 show the Pareto front for each instance and method.

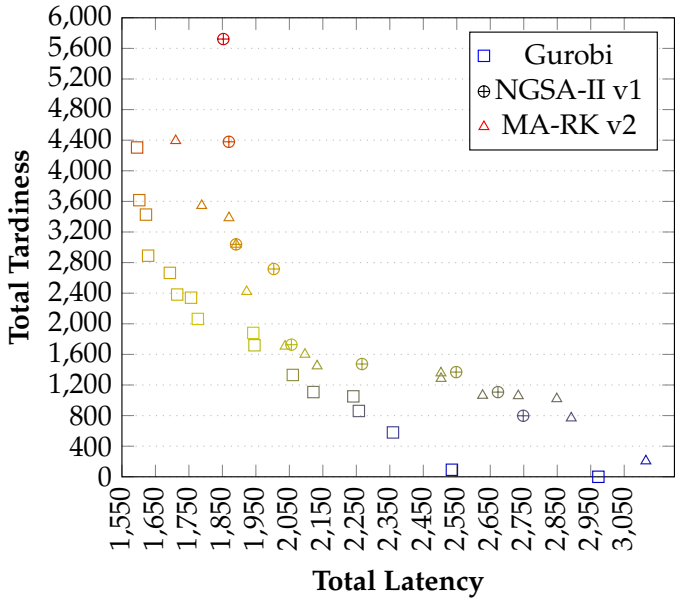

Figure 7. Pareto front for Instance FNO1.

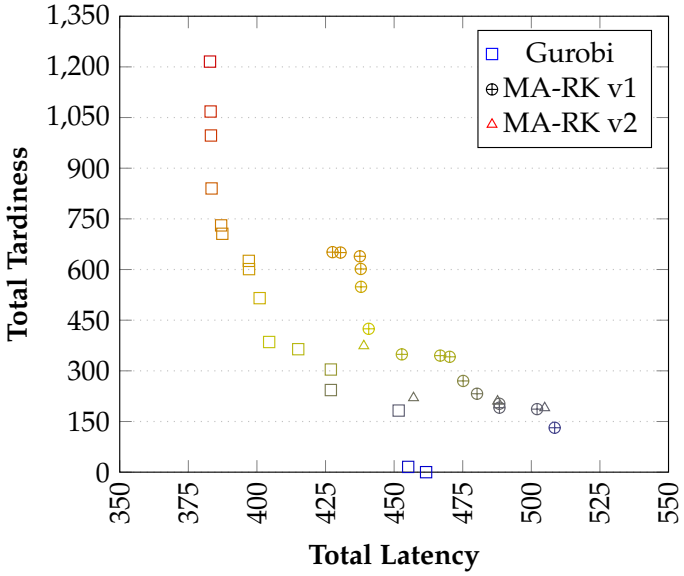

Figure 8. Pareto front for Instance FNO2.

A point to highlight in Figures 7 and 8 and Table 1 is that, for both instances, the Pareto front obtained by Gurobi is densely crowded. Additionally, notice that both versions of MA-RK performed differently over the solved instances. In particular, for Instance FNO1, the second version of the algorithm produced a front that is closer to the optimal front obtained by CPLEX. On the contrary,

for Instance FNO2, both algorithms produced fronts near to the optimal front and, in particular, the MA-RK v1 produced a more dense front than the MA-RK v2.

Table 1. Quantity of non-dominated points for Instances FNO1 and FNO2.

Instance Name	n	k	Gurobi	MA-RK v1	MA-RK v2
FNO1	12	5	17	9	15
FNO2	15	8	16	15	4

Regarding the computational time, the summarized results are presented in Table 2. In this case, the exact method required around 1 h for solving the FNO1 instance (12 nodes), whereas the time required to solve the FNO2 (15 nodes) instance was almost four times as long. In particular, both versions of the metaheuristic required less than 1 s to obtain the solutions. This fact, in conjunction with the metric of the quantity of non-dominated points, supports the evidence that, in general, the MA-RK algorithm performed very well.

Table 2. Elapsed CPU time (in seconds) for Instances FNO1 and FNO2.

Instance Name	Gurobi	MA-RK v1	MA-RK v2
FNO1	3,641.179	0.177	0.125
FNO2	13,742.185	0.256	0.149

Regarding the density of the fronts, Table 3 shows the average k-distance value of all points on the efficient frontiers for each instance, with $k = 3$. Specifically, the MA-RK produced fronts with more density than AUGMECON2. In particular, for the FNO2 instance, it can be seen that the NSGA-v1 obtained the minimum values for the maximum and average distances, while the MA-RK v2 obtained a denser front for Instance FNO1.

Table 3. Maximum and average k-distances for the FNO1 and FNO2.

Instance Name	Exact		MA-RK v1		MA-RK v2	
	Max	Avg	Max	Avg	Max	Avg
FNO1	0.41624	0.117976	0.469738	0.211416	0.225514	0.106864
FNO2	0.180034	0.121934	0.167452	0.0844987	0.410824	0.320028

To verify the efficiency of the MA-RK (in both versions), we used the hypervolume metric. Table 4 displays the obtained results. Again, the MA-RK v2 provided a better value of hypervolume for Instance FNO1, while the MA-RK v1 performed better on Instance FNO2.

Table 4. Hypervolume for Instances FNO1 and FNO2.

Instance Name	Exact	MA-RK v1	MA-RK v2
FNO1	0.821569	0.596382	0.64612
FNO2	0.799919	0.454889	0.432928

Finally, we used the coverage measure (considering only the strict domination). Tables 5 and 6 exhibit the results. In these tables, a value of C(X′, X″) equal to 1 means that all points in the estimated efficient frontier X″ are strictly dominated by points in the estimated efficient frontier X′. As expected, the exact method dominates both algorithms entirely in terms of the space covered. Regarding the

metaheuristic procedures, for Instance FNO1, the MA-RK v2 was able to generate points that dominate almost 77.78% over the ones generated by the MA-RK v1. For Instance FNO2, the front of the MA-RK v2 dominates 33.33% of the points generated by the MA-RK v1 which, in turn, dominates 25% of the points generated by the MA-RK v2.

Table 5. Coverage of two sets value for Instance FNO1.

X′/X″	Exact	MA-RK v1	MA-RK v2
Exact	0	1	1
MA-RK v1	0	0	0.066
MA-RK v2	0	0.778	0

Table 6. Coverage of two sets value for Instance FNO2.

X′/X″	Exact	MA-RK v1	MA-RK v2
Exact	0	1	1
MA-RK v1	0	0	0.25
MA-RK v2	0	0.333	0

We observe that, for Instances FNO1 and FNO2, when the minimum value of latency is obtained, the maximum amount of total tardiness rises to 3.2 times the cost of latency, which might translate to a higher level of customer dissatisfaction. On the contrary, when the minimum value of total tardiness is reached, the overall latency of the system rises up to 1.4 times the optimal (minimum) value. In this case, the increment of latency translates into a significant increase in the cost and, therefore, in a reduction of profit. In addition, it is obvious that the prioritization of the customers generates an unbalance in their demand, especially for the case of having routes where relatively few customers can have significantly high amounts of demand compared to the rest.

Another aspect to highlight is that the decision-making process can be seen from two perspectives: (1) savings in tardiness costs can represent up to 72% of the total costs; and (2) savings in latency produce savings for up to 28% of the total costs. In other words, according to the objective function, if a preference must be defined a priori, tardiness must be more important than latency.

4.3. Experimental Results for Larger Instances

The experimentation considering large size instances was conducted in both versions of the algorithm. The complementary experimentation involves instances of up to 100 nodes. Tables 7–12 display the results of our computational experimentation.

In Table 7, Column 1 displays the name of the instance. Columns 2 and 3 indicate the size in terms of the number of nodes and the number of routes. Columns 4 and 5 report the number of non-dominated solutions obtained by each algorithm. For Tables 9 and 10, Column 1 shows the name of the instance, whereas Columns 2 and 3 report the value of the corresponding algorithms over the evaluated metric. Specifically, for Table 11, two columns are used to indicate the maximum and average k-distances for each procedure.

Following the same sequence used in the previous section, the first metric to compare is the quantity of nondominated points. Table 7 reports the number of non-dominated points obtained by each algorithm (Pareto front). According to the information there, MA-RK v2 was able to obtain a higher quantity of non-dominated points. In some instances, the number of points reported almost doubled the amount of the ones obtained by MA-RK v1.This can be explained by the fact that MA-RK v2 generates more diverse solutions, since the process of intensification is selective.

Table 7. Quantity of non-dominated points for large-size instances.

Instance Name	n	k	MA-RK v1	MA-RK v2
FNO3	20	3	8	5
FNO4	22	8	8	6
FNO5	75	4	9	11
FNO6	75	7	5	18
FNO7	75	10	7	13
FNO8	75	7	10	5
FNO9	75	8	13	11
FNO10	100	5	8	12

In Table 8, the minimum and maximum values for each objective are shown. According to the information obtained, for most of the instances, the MA-RK v2 reports better values for the total latency. In addition, the MA-RK v2 produces better values of tardiness for most of the instances. In summary, the selective version of the MA-RK clearly outperforms the MA-RK v1.

Table 8. Minimum and maximum values for both objectives functions for large-size instances.

Instance Name	Type of Objective	MA-RK v1		MA-RK v2	
		Min	Max	Min	Max
FNO3	Latency	5327.35	7987.42	**4852.72**	**7335.90**
	Tardiness	4853.59	19,782.90	**3296.29**	**11,060.70**
FNO4	Latency	**660.85**	1102.98	759.182	**975.53**
	Tardiness	**233.65**	**1159.62**	290.55	2072.82
FNO5	Latency	8275.60	**10,376.60**	**6788.84**	10,460.50
	Tardiness	27,515.3	66,396.30	**23,903.30**	**51,934.70**
FNO6	Latency	7366.98	**9251.54**	**6933.74**	16,841.5
	Tardiness	20,883.80	31,319.70	**11,558.3**	**16,063.70**
FNO7	Latency	12,511.70	15,026.20	**10,027.10**	**11,253.50**
	Tardiness	82,739.9	**124,013.00**	**67,365.4**	144,892
FNO8	Latency	13,683.50	16,203.90	**12,833.5**	**14,208.1**
	Tardiness	116,976.00	199,977.00	**101,302**	**154,659**
FNO9	Latency	9617.94	14,949.70	**9150.29**	**11,955.2**
	Tardiness	69,126.70	130,754.00	**58,969.5**	**123,515.00**
FNO10	Latency	31,092.50	40,164.90	**26,484.2**	**35,107.6**
	Tardiness	343,722.00	530,071.00	**272,748.00**	**335,684.00**

Regarding the performance of the algorithms, it can be noticed that, for larger instances, the MA-RK v2 clearly outperforms the MA-RK v1. To better illustrate this, the fronts of Instances FNO8 and FNO10 are displayed in Figures 9 and 10.

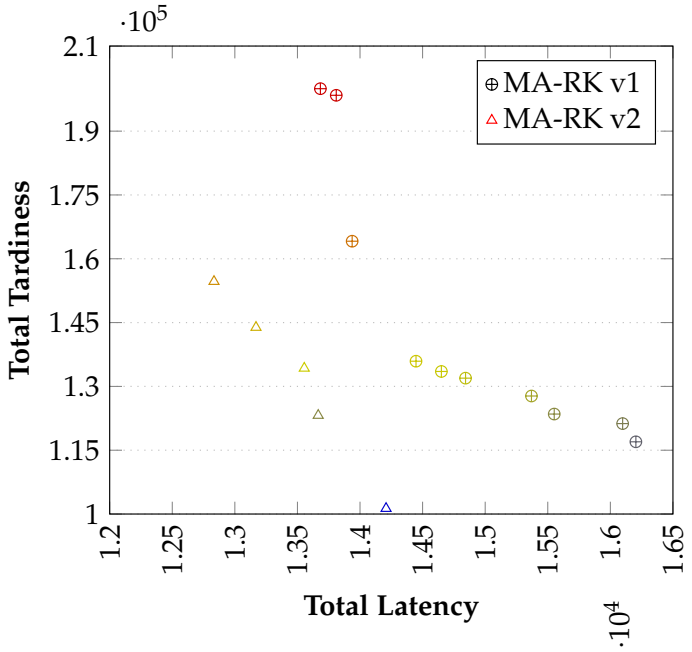

Figure 9. Pareto fronts for Instance FNO8.

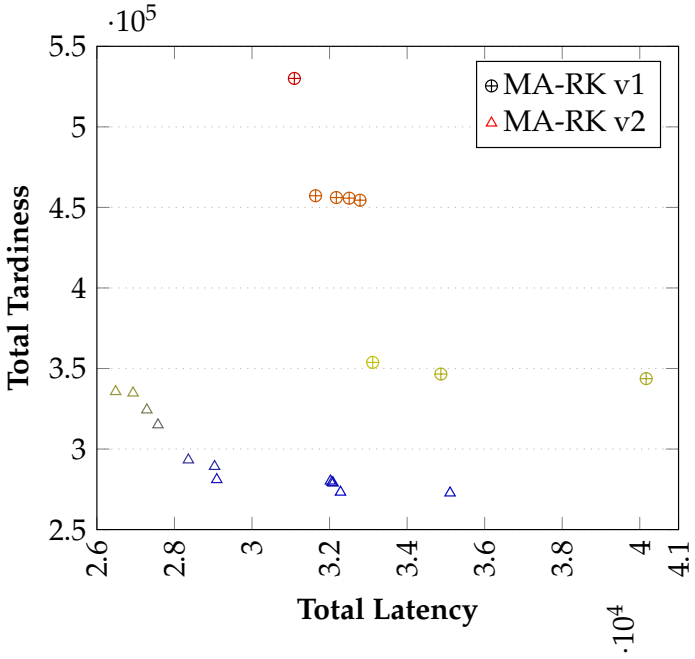

Figure 10. Pareto fronts for Instance FNO10.

Due to this, the metric of the execution time was evaluated to verify if any of the versions performs more quickly. Table 9 displays the elapsed CPU time for the best execution.

Table 9. Elapsed CPU time for the rest of the instances.

Instance Name	MA-RK v1	MA-RK v2
FNO3	0.262	0.817
FNO4	0.768	1.087
FNO5	6.415	6.586
FNO6	9.919	11.837
FNO7	62.262	70.162
FNO8	47.039	47.819
FNO9	48.816	54.017
FNO10	96.541	97.29

As expected, the required time increases as the size of the instances increased. However, both versions of the metaheuristic were working within a similar computational performance range.

As for the third metric, the hypervolume, the results of the algorithm are displayed in Table 10. There, we do not have enough evidence to confirm that any algorithm outperforms the other. What can be confirmed is that MA-RK v2 produced higher values of hypervolume for seven out of eight instances. However, for the instance where the MA-RK v1 obtained better values, the difference against the NSGA was small.

Table 10. Hypervolume for the rest of the instances.

Instance Name	MA-RK v1	MA-RK v2
FNO3	0.680387	0.836718
FNO4	0.752206	0.648306
FNO5	0.397265	0.823822
FNO6	0.642281	0.929850
FNO7	0.472473	0.646997
FNO8	0.433841	0.847476
FNO9	0.527335	0.836342
FNO10	0.390825	0.959196

Regarding the density of the frontiers, Table 11 shows the results obtained. From these results, it can be noticed that, in most of the cases, the MA-RK v2 produced lower values for the maximum distances (more compactness). However, the MA-RK v1 algorithm produced better values for the average distances.

Table 11. Maximum and average *k*-distances for the large-size instances.

Instance Name	MA-RK v1		MA-RK v2	
	Max	Avg	Max	Avg
FNO3	0.579687	0.308544	0.503147	0.389821
FNO4	0.418041	0.256448	0.643338	0.344496
FNO5	0.483586	0.166288	0.742506	0.168127
FNO6	0.516558	0.298917	0.483949	0.107271
FNO7	0.342773	0.179116	0.628917	0.176940
FNO8	0.371050	0.212086	0.386249	0.248088
FNO9	0.606253	0.161301	0.574856	0.179261
FNO10	0.387176	0.187895	0.221685	0.0603717

Finally, Table 12 reports the values obtained for the set coverage metric. The first column refers to the name of the instance, and the rest show comparisons in coverage between the algorithms. Again, MA-RK v2 performed better than MA-RK v1, by dominating the entire front provided by MA-RK v1. This confirms that the selective version of the MA-RK clearly dominates.

Table 12. Coverage of two sets value for the rest of the instances.

Instance Name	X′/X″	Exact	
		MA-RK v1	MA-RK v2
FNO3	MA-RK v1	0	0
	MA-RK v2	0.875	0
FNO4	MA-RK v1	0	0.333
	MA-RK v2	0.500	0
FNO5	MA-RK v1	0	0
	MA-RK v2	0.889	0
FNO6	MA-RK v1	0	0
	MA-RK v2	0.800	0
FNO7	MA-RK v1	0	0
	MA-RK v2	1	0
FNO8	MA-RK v1	0	0
	MA-RK v2	1	0
FNO9	MA-RK v1	0	0
	MA-RK v2	1	0
FNO10	MA-RK v1	0	0
	MA-RK v2	1	0

In summary, we conclude that, even when both metaheuristics provide good results in a reasonable computational time, the MA-RK v2 consistently outperforms the non-selective MA-RK version.

5. Conclusions and Future Work

This study addressed the biobjective Cumulative Capacitated Vehicle Routing Problem. This problem mainly arises in commercial contexts such as the delivery of perishable goods, in which there are differentiated based on priority indexes. In the case of a pooled transportation service, it might help to estimate the trade-off between delivering the orders in the same sequence as customers board the vehicle and the minimum arrival time of the system. For this problem, a mixed-integer programming formulation and two metaheuristic algorithms were developed. A commercial optimization software was able to solve the model for small size instances, whereas the algorithms showed their effectiveness by providing feasible results in a reasonable amount of computational time.

The algorithms showed their efficiency by providing good quality fronts for the small size instances. Additionally, for larger instances, both algorithms provided good values for the multiobjective metrics evaluated. Although none of the algorithms outperformed each other, the MA-RK v2 obtained fronts with a higher quantity of points, more density, and more coverage of the sets. However, the MA-RK v1 was slightly faster. One intuition is that MA-RK v1 is more intensive in the local search, stagnating in local optima, while MA-RK v2 maintains the diversity, allowing to escape from local optima and populating the Pareto-fronts. However, more computational experiments are needed to clarify this effect.

In summary, all procedures provided a positive contribution to a more sustainable balance between economic and customer service objectives. Our results provide useful insights for business applications in terms of considering customer satisfaction and gaining a valuable sustainable advantage given that the reduction in the traveled time translates into a reduction of CO_2 emissions.

Future research lines include the design of routes using congested environments with travel speed variation. This fact can be addressed either by modifying the objective function to include the variability in the travel time during the day or using a risk aversion approach, by adding a profit to each node associated with the order of visit. In addition, considerations involving time windows as priority metrics, and demand uncertainty can be worth consideration, as well as factors such as labor costs or balance of the total traveled distance among routes, which seem to dominate the overall cost.

Author Contributions: Conceptualization, S.N.-G. and E.O.-B.; methodology, S.N.-G. and D.F.-D.; software, S.N.-G. and D.F.-D.; validation, S.N.-G., D.F.-D., and E.O.-B.; formal analysis, S.N.-G., D.F.-D., and E.O.-B.; investigation, S.N.-G. and D.F.-D.; resources, S.N.-G. and E.O.-B.; data curation, S.N.-G. and D.F.-D.; writing—original draft preparation, S.N.-G.; writing—review and editing, E.O.-B. and A.M.; visualization, S.N.-G.; supervision, S.N.-G. and E.O.-B.; project administration, S.N.-G. and E.O.-B.; and funding acquisition, S.N.-G., E.O.-B., and A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Universidad Panamericana through the grant "Fomento a la Investigación 2019", under project code UP-CI-2019-ING-GDL-08.

Acknowledgments: This work was supported by the Universidad Panamericana through the grant "Fondo Fomento a la Investigación UP 2019", under project code UP-CI-2019-ING-GDL-08.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dantzig, G.B.; Ramser, J.H. The truck dispatching problem. *Manag. Sci.* **1959**, *6*, 80–91. [CrossRef]
2. Ngueveu, S.U.; Prins, C.; Calvo, R.W. An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Comput. Operations. Res.* **2010**, *37*, 1877–1885. [CrossRef]
3. Martínez-Salazar, I.; Angel-Bello, F.; Alvarez, A. A customer-centric routing problem with multiple trips of a single vehicle. *J. Oper. Res. Soc.* **2015**, *66*, 1312–1323. [CrossRef]
4. Rivera, J.C.; Afsar, H.M.; Prins, C. A multistart iterated local search for the multitrip cumulative capacitated vehicle routing problem. *Comput. Optim. Appl.* **2015**, *61*, 159–187. [CrossRef]
5. Gaur, D.R.; Mudgal, A.; Singh, R.R. Improved approximation algorithms for cumulative VRP with stochastic demands. *Discret. Appl. Math.* **2018**. [CrossRef]
6. Lalla-Ruiz, E.; Voß, S. A POPMUSIC approach for the Multi-Depot Cumulative Capacitated Vehicle Routing Problem. *Optim. Lett.* **2020**, *14*, 671–691. [CrossRef]
7. Karagul, K.; Sahin, Y.; Aydemir, E.; Oral, A. A Simulated Annealing Algorithm Based Solution Method for a Green Vehicle Routing Problem with Fuel Consumption. In *Lean and Green Supply Chain Management: Optimization Models and Algorithms*; Paksoy, T., Weber, G.W., Huber, S., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 161–187, doi:10.1007/978-3-319-97511-5_6. [CrossRef]
8. Kara, İ.; Kara, B.Y.; Yetiş, M.K. Cumulative vehicle routing problems. In *Vehicle Routing Problem*; IntechOpen: Rijeka, Croatia, 2008.
9. Rivera, J.C.; Afsar, H.M.; Prins, C. Mathematical formulations and exact algorithm for the multitrip cumulative capacitated single-vehicle routing problem. *Eur.J. Oper. Res.* **2016**, *249*, 93–104. [CrossRef]
10. Nucamendi-Guillén, S.; Angel-Bello, F.; Martínez-Salazar, I.; Cordero-Franco, A.E. The cumulative capacitated vehicle routing problem: New formulations and iterated greedy algorithms. *Expert Syst. Appl.* **2018**, *113*, 315–327. [CrossRef]
11. Lysgaard, J.; Wøhlk, S. A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem. *Eur. J. Oper. Res.* **2014**, *236*, 800–810. [CrossRef]
12. Ribeiro, G.M.; Laporte, G. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Comput. Oper. Res.* **2012**, *39*, 728–735. [CrossRef]
13. Ozsoydan, F.B.; Sipahioglu, A. Heuristic solution approaches for the cumulative capacitated vehicle routing problem. *Optimization* **2013**, *62*, 1321–1340. [CrossRef]
14. Ke, L.; Feng, Z. A two-phase metaheuristic for the cumulative capacitated vehicle routing problem. *Comput. Oper. Res.* **2013**, *40*, 633–638. [CrossRef]
15. Sbihi, A.; Eglese, R.W. Combinatorial optimization and Green Logistics. *4OR* **2007**, *5*, 99–116, doi:10.1007/s10288-007-0047-3. [CrossRef]

16. Kwon, Y.J.; Choi, Y.J.; Lee, D.H. Heterogeneous fixed fleet vehicle routing considering carbon emission. *Transp. Res. Part D Transp. Environ.* **2013**, *23*, 81–89, doi:10.1016/j.trd.2013.04.001. [CrossRef]
17. Dewilde, T.; Cattrysse, D.; Coene, S.; Spieksma, F.C.; Vansteenwegen, P. Heuristics for the traveling repairman problem with profits. *Comput. Oper. Res.* **2013**, *40*, 1700–1707. [CrossRef]
18. Bruni, M.; Beraldi, P.; Khodaparasti, S. A heuristic approach for the k-traveling repairman problem with profits under uncertainty. *Electron. Notes Discret. Math.* **2018**, *69*, 221–228. [CrossRef]
19. Arellano-Arriaga, N.A.; Molina, J.; Schaeffer, S.E.; Álvarez-Socarrás, A.; Martínez-Salazar, I.A. A biobjective study of the minimum latency problem. *J. Heuristics* **2019**, *25*, 431–454. [CrossRef]
20. Elshaer, R.; Awad, H. A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Comput. Ind. Eng.* **2020**, *140*, doi:10.1016/j.cie.2019.106242. [CrossRef]
21. Li, X.; Shi, X.; Zhao, Y.; Liang, H.; Dong, Y. SVND enhanced metaheuristic for plug-in hybrid electric vehicle routing problem. *Appl. Sci.* **2020**, *10*, 441, doi:10.3390/app10020441. [CrossRef]
22. Zhang, K.; Cai, Y.; Fu, S.; Zhang, H. Multiobjective memetic algorithm based on adaptive local search chains for vehicle routing problem with time windows. *Evol. Intell.* **2019**, doi:10.1007/s12065-019-00224-7. [CrossRef]
23. He, L.; Guijt, A.; de Weerdt, M.; Xing, L.; Yorke-Smith, N. Order acceptance and scheduling with sequence-dependent setup times: A new memetic algorithm and benchmark of the state of the art. *Comput. Ind. Eng.* **2019**, *138*, doi:10.1016/j.cie.2019.106102. [CrossRef]
24. Li, X.; Yin, M. A hybrid cuckoo search via Lévy flights for the permutation flow shop scheduling problem. *Int. J. Prod. Res.* **2013**, *51*, 4732–4754, doi:10.1080/00207543.2013.767988. [CrossRef]
25. Ghrayeb, O.; Damodaran, P. A hybrid random-key genetic algorithm to minimize weighted number of late deliveries for a single machine. *Int. J. Adv. Manuf. Technol.* **2013**, *66*, 15–25, doi:10.1007/s00170-012-4302-1. [CrossRef]
26. Samanlioglu, F.; Ferrell, W.; Kurz, M. An interactive memetic algorithm for production and manufacturing problems modelled as a multiobjective travelling salesman problem. *Int. J. Prod. Res.* **2012**, *50*, 5671–5682, doi:10.1080/00207543.2011.593578. [CrossRef]
27. Gavish, B.; Graves, S.C. *The Travelling Salesman Problem and Related Problems*; Massachusetts Institute of Technology, Operations Research Center: Cambridge, MA, USA, 1978.
28. Mavrotas, G.; Florios, K. An improved version of the augmented ε-constraint method (AUGMECON2) for finding the exact pareto set in multiobjective integer programming problems. *Appl. Math. Comput.* **2013**, *219*, 9652–9669, doi:10.1016/j.amc.2013.03.002. [CrossRef]
29. Boudia, M.; Prins, C.; Reghioui, M. An effective memetic algorithm with population management for the split delivery vehicle routing problem. In *International Workshop on Hybrid Metaheuristics*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 16–30.
30. Karaoglan, I.; Altiparmak, F. A memetic algorithm for the capacitated location-routing problem with mixed backhauls. *Comput. Oper. Res.* **2015**, *55*, 200–216. [CrossRef]
31. Kechmane, L.; Nsiri, B.; Baalal, A. A memetic algorithm for the capacitated location-routing problem. *Int. J. Adv. Comput. Sci. Appl.* **2016**, *7*. [CrossRef]
32. Nalepa, J.; Blocho, M. Adaptive memetic algorithm for minimizing distance in the vehicle routing problem with time windows. *Soft Comput.* **2016**, *20*, 2309–2327. [CrossRef]
33. Sales, L.P.A.; Melo, C.S.; Bonates, T.O.; Prata, B.A. Memetic algorithm for the heterogeneous fleet school bus routing problem. *J. Urban Plan. Dev.* **2018**, *144*, 04018018. [CrossRef]
34. Decerle, J.; Grunder, O.; El Hassani, A.H.; Barakat, O. A memetic algorithm for a home health care routing and scheduling problem. *Oper. Res. Health Care* **2018**, *16*, 59–71. [CrossRef]
35. Peng, B.; Zhang, Y.; Gajpal, Y.; Chen, X. A Memetic Algorithm for the Green Vehicle Routing Problem. *Sustainability* **2019**, *11*, 6055. [CrossRef]
36. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1992.
37. Prins, C. A simple and effective evolutionary algorithm for the vehicle routing problem. *Comput. Oper. Res.* **2004**, *31*, 1985–2002. [CrossRef]
38. Moscato, P.; Cotta, C. An accelerated int;roduction to memetic algorithms. In *Handbook of Metaheuristics*; Springer: Boston, MA, USA, 2019; pp. 275–309.

39. Martínez-Salazar, I.A.; Molina, J.; Ángel-Bello, F.; Gómez, T.; Caballero, R. Solving a biobjective transportation location routing problem by metaheuristic algorithms. *Eur. J. Oper. Res.* **2014**, *234*, 25–36. [CrossRef]
40. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
41. Koulaeian, M.; Seidgar, H.; Kiani, M.; Fazlollahtabar, H. A Multi Depot Simultaneous Pickup and Delivery Problem with Balanced Allocation of Routes to Drivers. *Int. J. Ind. Eng. Theory Appl. Pract.* **2015**, *22*, 223–242.
42. Chunyu, R.; Xiaobo, W. Research on Multi-vehicle and Multi-Depot Vehicle Routing Problem with Time Windows for Electronic Commerce. In Proceedings of the 2010 International Conference on Artificial Intelligence and Computational Intelligence, Sanya, China, 23–24 October 2010; Volume 1, pp. 552–555, doi:10.1109/AICI.2010.121. [CrossRef]
43. Gillett, B.E.; Johnson, J.G. Multi-terminal vehicle-dispatch algorithm. *Omega* **1976**, *4*, 711–718. [CrossRef]
44. Augerat, P.; Belenguer, J.M.; Benavent, E.; Corberán, A.; Naddef, D.; Rinaldi, G. *Computational Results with a Branch and Cut Code for the Capacitated Vehicle Routing Problem*; Technical Report; IMAG, Institut National Polytechnique: Grenoble, France, 1995.
45. Taillard, É.D. A heuristic column generation method for the heterogeneous fleet VRP. *RAIRO-Oper. Res.* **1999**, *33*, 1–14. [CrossRef]
46. Li, F.; Golden, B.; Wasil, E. A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Comput. Oper. Res.* **2007**, *34*, 2734–2742. [CrossRef]
47. Zitzler, E.; Laumanns, M.; Thiele, L. SPEA2: Improving the strength Pareto evolutionary algorithm. In *EUROGEN 2001, Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*; International Center for Numerical Methods in Engineering (CIMNE): Barcelona, Spain, 2002; pp. 95–100.
48. Zitzler, E.; Thiele, L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Trans. Evol. Comput.* **1999**, *3*, 257–271. [CrossRef]