Optimizing a Bi-Objective Vehicle Routing Problem that appears in industrial enterprises

Ana D. López-Sánchez, Julián Molina, Manuel Laguna, Alfredo G. Hernández-Díaz

Abstract

In this paper, a new solution method is implemented to solve a bi-objective variant of the Vehicle Routing Problem that appears in industry and environmental enterprises. The solution involves designing a set of routes for each day in a period, in which the service frequency is a decision variable. The proposed algorithm, the Muti-Start Multi-objective Local Search algorithm (MSMLS), minimizes total emissions produced by all vehicles and maximizes the service quality measured as the number of times that a customer is visited by a vehicle in order to be served. The MSMLS is a neighborhood-based metaheuristic that obtains high-quality solutions and that is capable of achieving better performance than other competitive algorithms. Furthermore, the proposed algorithm is able to perform rapid movements thanks to the easy representation of the solutions.

Keywords: Periodic Vehicle Routing Problem; Frequency; Service Choice; Multi-Start Algorithm; Local Search; Proximate Optimality Principle

1 Introduction

Governments commonly impose production suspensions on industrial enterprises depending on the emission levels in their counties due to inhabitants' discontent with the environmental pollution. Various measures are applied regarding production in order to satisfy environmental requirements, which have become crucial issues. This work focuses on industrial logistics enterprises. Industrial logistics services involve shipment of raw materials and end products by means of various modes of transport of vehicles (such as trains and lorries) in order to meet needs of enterprises working in different economic sectors.

In this work, the transportation of products by industrial enterprises to their customers is specifically addressed. The problem considered herein is known as the Bi-objective Periodic Vehicle Routing Problem with Service Choice (Bi-PVRP-SC). The model designs vehicle routes for every day of a planning horizon (for example, one week or month) for a fleet of capacitated vehicles, that begin and end at a single depot. Customers are visited as minimum a preset number of times over the planning horizon. The Bi-PVRP-SC seeks to minimize total emissions produced by all vehicles, since if the quantity is high, then the industrial enterprises will be subject to production suspensions, but also to maximize the service frequency since the customers consider that the service given by the enterprises will be better if they are served with a high frequency. Note that our model designs routes such that vehicles are assigned to customers and days; customers require a minimum number of visits during the planning period which depends on the demand rate. Of course, the two objectives are clearly in conflict: the minimization of the total emissions produced by all vehicles and the maximization of the service frequency. A better service level given to the customers implies a higher quantity of emissions produced by all vehicles since the customers will be served more frequently by vehicles and that will increase the environmental pollution. Otherwise, emissions will be reduced if the service frequency is minimized. Note that total emissions are proportional to the amount of greenhouse gas (GHG) emitted.

Formally, the Bi-PVRP-SC designs routes using vehicles with a limited capacity Q and on a complete graph, $G = (V, E)$, where $V = \{0\} \cup N$ is the vertex set and E is the edge set. Vertex 0 corresponds to the depot and the set of nodes N corresponds to the customers. A nonnegative cost c_{ij} is associated with each edge $(i, j) \in E$ and represents the travel cost (or the emissions) to travel from i to j. The problem considers service for a planning horizon T of $|T|$ consecutive days. Furthermore, each customer, $i \in N$, is characterized by the daily demand (or volume), d_i, and a minimum frequency of visits, γ_i and has a limited capacity, D_i. Consequently, the service is performed according to several possible service profiles (or schedule options) which are specified by the days in the planning horizon when the customers are served (either pick-up or delivery) and they implicitly determine the service frequency. Let S be the set of service profiles, each associated with a σ_{st} flag which takes the value 1 if service profile $s \in S$ is used in day $t \in T$, and 0 otherwise. This is useful when the demand is not homogeneously distributed every day. For example, if $|T| = 3$ then $|S| = 7$ and $\sigma_{st} = \{\{1, 0, 0\}, \{0, 1, 0\}, \{0, 0, 1\}, \{1, 1, 0\}, \{0, 1, 1\}, \{1, 0, 1\}, \{1, 1, 1\}\}$. Note that the first three service profiles imply a service frequency of one visit over the planning horizon (the first, the second, and the third day, respectively); the second three service profiles assume a service frequency of two visits (the first and the second day, the second and the third day, and the first and the third day, respectively); and the last service profile means that the service is carried out every single day over the three days on the planning horizon.

The Bi-PVRP-SC can be seen as a generalization of the Periodic Vehicle Routing Problem (PVRP). This Periodic or Period Vehicle Routing Problem is a generalization of the classic Vehicle Routing Problem (VRP), which was first proposed by [2] in the context of waste collection. Their solution approach constructs vehicle routes for a $t-$day planning horizon (for example, one week). During each day of the planning horizon, a fleet of capacitated vehicles performs routes that begin and end at a single depot. Customers are visited a preset number of times over the planning horizon, by following a schedule that is chosen from a menu of schedule options. Each schedule option represents a set of days on which a customer is visited. The objective of the PVRP is to find a set of routes for each vehicle over the planning horizon that minimizes the total travel cost while satisfying operational constraints (vehicle capacity and visit requirements). See also, [16] who addressed the same problem under the name of “The Assignment Routing Problem”. Later, [6] named it “the period routing problem” and provided the first mathematical formulation. [10] were the first authors to use the term “periodic”. The theoretical and practical interest in the PVRP explains the growth in publications for over forty years. For a comprehensive review, see [4] and for a recent related problem see [1].

There exists a wide variety of models for the PVRP which differ in how they define the allowable service profiles, that is, the rules employed to determine when and how often to visit each customer. The literature reports three main ways in which service profiles are generated. The first consists of a predetermined set of permitted alternatives. The second specifies the frequency, that is, the number of times that each customer is visited within the planning horizon. The third enforces constraints on the minimum and the maximum required spacing of times between visits. For instance, if a customer requires two visits during a planning horizon of five days, then:

- under the first option (used in [2] and [6]), the allowable service profiles may be to visit on days 1 and 5, or days 1 and 4, or days 2 and 4;
- under the second option (used in [5]), the frequency is set to twice during the 5-day planning horizon, resulting in 5 profiles with visits on days 1 and 3, days 1 and 4, days 2 and 4, days 2 and 5 or days 3 and 5; and
- under the third option (see [10] and [12]), two successive services are spaced by at least one day and at most two days over the planning horizon.

However, the Bi-PVRP-SC introduces a new representation that produces schedules of a more realistic nature. For each customer, this new representation imposes a minimum frequency of visits and a maximum capacity and is directly applicable in more areas apart from industrial enterprises, for instance, in the context of waste collection, where the containers have a limited capacity and the sanitary authorities impose a minimum frequency of collection. Similarly, in the grocery business, stores must be visited before they run out of product (due to their limited capacity to stock items) and at the same time a minimum service is required since certain products cannot be stored in large quantities (due to inventory policies or because they are perishable).

Another variant is the Periodic Vehicle Routing Problem with Service Choice (PVRP-SC) introduced by [9]. The PVRP-SC is defined as the problem of finding a set of routes for each vehicle and for each day over a planning horizon in order to minimize total travel cost minus service benefit while satisfying operational constraints (vehicle capacity and frequency of visits) by suggesting efficient single-objective solution methods. The authors compute the service benefit function as the node stopping costs and a demand-weighted service benefit. The focal point of adding service choice is that it can improve customer service, represented either by savings in holding cost or by the customer’s willingness to pay for a more frequent (better) service. Thanks to service choice, some customers may receive better service than the minimum required. We turn this problem into a bi-objective formulation, by separating the total travel cost from the service benefit, with the main difference that we estimate the service benefit as the service frequency instead of as the previous authors. We do believe that the costs or benefits to estimate the service benefit function depend on each enterprise and this proxy is more appropriated. To the best of our knowledge the Bi-PVRP-SC has yet to be tackled in the literature. [12] address a similar problem although they solve a bi-objective waste collection problem in which they minimize the sum of fixed and variable costs throughout the planning horizon (where, fixed costs depend on the class of vehicle used and the day) and service is measured as the waste accumulated throughout the planning horizon.

As [9] state, the PVRP-SC is similar to the Inventory Routing Problem (IRP) since its solutions determine both frequency of visits and route configuration. The difference is that in the PVRP-SC the service quantities are determined by the schedule (i.e., all demand accumulated since the last visit), while in the IRP, the service quantities are decision variables that are independent from the frequency of visits. Furthermore, in the PVRP-SC, the minimum service requirements are known in advance; specifically, the service level is related to the customer and depends on a parameter determined by the customer's volume and the chosen service. In the IRP, however, the service-related costs are modeled as holding costs and therefore are associated with each specific item without imposing minimum frequencies.

In order to solve the Bi-PVRP-SC, a dual-phase Multi-Start Multi-objective Local Search (MSMLS) algorithm is proposed, see [3] and [14] where the authors use multi-start local search algorithms to solve related single-objective problems. In the first phase, feasible solutions are generated in order to obtain an initial approximation of the Pareto front. The second phase strives to improve the approximation of the Pareto front by performing local searches that employ several neighborhoods. We use the classic definition of an efficient solution as one for which no single-objective function value can be improved without deteriorating another objective function value. Therefore, in a bi-objective optimization problem in which, without loss of generality, the goal is the minimization of both objectives, f_1 and f_2, a solution x^* *dominates* x if $f_1(x^*) \leq f_1(x)$ and $f_2(x^*) \leq f_2(x)$ and either $f_1(x^*) < f_1(x)$ or $f_2(x^*) < f_2(x)$. According to this definition, a solution is *efficient* if there is no other solution that dominates it. Hence, the goal is to find the set of non-dominated solutions that approximates the set of efficient solutions, $\hat{E}$.

As describe above, the main contributions of this paper are as follows:

- We address the PVRP-SC but from a different perspective, instead of considering the problem as a single-objective optimization problem, we have solved the problem as a bi-objective optimization one: total emissions produced by all vehicles and maximizes the service quality measured as the number of times that a customer is visited by a vehicle in order to be served.

- From a methodological perspective, we propose a multi-objective approach based on Multi-Start Local Searches. The heuristic approach is able to generate service profile alternatives during the search process instead of operating on a predetermined set.

- Through our numerical study, we confirm that the proposed algorithm outperforms other competitive algorithms widely used when more than one objective functions are considered.

The paper is organized as follows. Section 2 describes the proposed algorithm and Section 3 shows the adaptation of the NSGA-II algorithm to our problem. Following explanations of the two approaches, Section 4 shows the computational results. Finally, Section 5 summarizes our findings and discusses future work.

2 Multi-Start Multi-objective Local Search Procedure

A new solution method called Multi-Start Multi-objective Local Search (MSMLS) is proposed. This algorithm is a neighborhood-based metaheuristic designed to solve multi-objective problems. One of the key aspects of the proposed procedure is its simplicity and robustness. The MSMLS separate two different phases: first, it generates an initial feasible solution and then it applies a multi-objective local search to improve on this solution. The phases are repeated until a stopping criterion is met.

2.1 Phase 1. Solution representation and Construction phase

A compact solution representation that is both flexible and realistic is introduced, which considers the context at hand. Our representation is able to use the direct information obtained from the customers (minimum frequency and maximum capacity) without the need to previously generate a set of feasible service profiles. Within this problem, there are three main decisions that lead to the construction of a solution for a given planning horizon, each of which are represented by using the following variables:

Decision 1. The number of visits to each customer. For all $i \in N$, $freq_i$ is an integer value whose values ranges from the minimum preset frequency to the planning horizon, that is, $\gamma_i \leq freq_i \leq |T|$, which indicates the number of days customer i is visited.

Decision 2. The service profile of each customer (i.e., the actual days when each customer will be visited). For all $i \in N, j = \{1, \ldots, freq_i\}$, t_{ij} is an integer value that ranges from 1 to $|T|$ and indicates the days that customer i is visited. Note that for a given i all t_{ij} values differ.

Decision 3. The number of routes and the order in which each customer will be visited, on each day.

For all $i \in N, j = \{1, \ldots, freq_i\}$, r_{ij} =:

- 1, if customer i is the first one visited on a new route on day t_{ij}

- 0, otherwise

For all $i \in N$ and $t \in T$, O_{it} indicates the position at which node i is visited on day t.

In order to clearly understand the solution representation, an example with $|N| = 5$, $|T| = 3$ is considered, and the values for the rest of the variables are shown in Table 1 and Table 2:

Customer	Frequency	Days	First visit
Customer 1	$freq_1 = 1$	$t_{11} = 1$	$r_{11} = 1$
Customer 2	$freq_2 = 2$	$t_{21} = 1$	$r_{21} = 0$
		$t_{22} = 3$	$r_{22} = 1$
Customer 3	$freq_3 = 3$	$t_{31} = 3$	$r_{31} = 0$
		$t_{32} = 2$	$r_{32} = 0$
		$t_{33} = 1$	$r_{33} = 0$
Customer 4	$freq_4 = 2$	$t_{41} = 2$	$r_{41} = 0$
		$t_{42} = 3$	$r_{42} = 0$
Customer 5	$freq_5 = 1$	$t_{51} = 3$	$r_{51} = 0$

Table 1: Assignation of number of visits and determination of service profiles

O_{it}	Day 1	Day 2	Day 3
Customer 1	4	5	1
Customer 2	3	2	4
Customer 3	2	3	3
Customer 4	1	1	2
Customer 5	5	4	5

Table 2: Ordered list of customers' visits every day

- $freq_1 = 1$, that is, customer 1 must be visited once during the three days of the planning period; $freq_2 = 2$, customer 2 must be visited twice; $freq_3 = 3$, customer 3 must be visited every day; $freq_4 = 2$, customer 4 must be visited on two days; and finally, $freq_5 = 1$, customer 5 must be visited just one day during the three days of the planning period (see Table 1).
- $t_{11} = 1$, that is, customer 1 is served on day 1; $t_{21} = 1, t_{22} = 3$, customer 2 is served on days 1 and 3; $t_{31} = 3, t_{32} = 2, t_{33} = 1$, customer 3 is served on days 3, 2, and 1; $t_{41} = 2, t_{42} = 3$, customer 4 is served on days 2 and 3; and $t_{51} = 3$, customer 5 is served on day 3 (see Table 1).
- $r_{11} = 1, r_{21} = 0, r_{22} = 1, r_{31} = 0, r_{32} = 0, r_{33} = 0, r_{41} = 0, r_{42} = 0, r_{51} = 0$, that is, customer 1 is the first customer visited within a new route on day 1 since $t_{11} = 1$ and $r_{11} = 1$ and customer 2 is the first customer visited within a new route on day 3 since $t_{22} = 3$ and $r_{22} = 1$ (see Table 1).

- $O_{11} = 4, O_{21} = 3, O_{31} = 2, O_{41} = 1, O_{51} = 5$, that is, on day 1 the order in which to visit customers should be $4 - 3 - 2 - 1 - 5$ (see Table 2) but according to Table 1, only customers 1, 2, and 3 must be visited on such a day. Thus, they must be visited in the order $3 - 2 - 1$.
- $O_{12} = 5, O_{22} = 2, O_{32} = 3, O_{42} = 1, O_{52} = 4$, that is, on day 2 the order to in which visit customers should be $5 - 2 - 3 - 1 - 4$ (see Table 2) but according to Table 1, only customers 3 and 4 must be visited on such a day. Thus, the order must be $4 - 3$.
- $O_{13} = 1, O_{23} = 4, O_{33} = 3, O_{43} = 2, O_{53} = 5$, that is, on day 3 the order in which to visit customers should be $1 - 4 - 3 - 2 - 5$ (see Table 2) but according to Table 1, only customers 2, 3, 4, and 5 must be visited on such a day. Thus, the order among them become $4 - 3 - 2 - 5$.

Now, on taking r_{ij} into account, we obtain the following routes:

- Day 1, route 1: 0-3-2-0
- Day 1, route 2: 0-1-0
- Day 2, route 1: 0-3-4-0
- Day 2, there is no route 2
- Day 3, route 1: 0-4-3-0
- Day 3, route 2: 0-2-5-0

Note that for each day, the first route always takes the first customer in the order list, and in the remaining routes, if any, the customer is indicated by variable r_{ij}.

One of the best properties of this representation is that solutions are always feasible, except for two cases:

- when the capacity of a vehicle is full before reaching the next customer with $r_{ij} = 0$. However, this solution can easily be made feasible. If a customer cannot be visited by a given vehicle due to the capacity constraint of the vehicle, we simply, at that point, set $r_{ij} = 1$ and then a new route starts at that customer, that is, the route is split into two separate routes.
- when the demand accumulated over the days between any two successive visits exceeds the capacity of the customer, D_i for all $i \in N$. In this case, the solution cannot be repaired because the algorithm would force a visit to the customer before its accumulated demand overloads.

With this solution representation, we develop an effective construction method to produce feasible solutions with a reasonable computational effort. Our construction method employs two strategies for the generatation of the number of visits during the planning horizon:

1. Set the number of visits to the minimum allowed, in order to generate good solutions for the first objective: the minimization of the total routing cost.

2. Select the number of visits randomly between the minimum allowed and the total days in the planning horizon, in order to generate good solutions for second objective: the maximization of the service frequency.

In both cases, the service profile is designed randomly while observing the customers' capacity constraints, and once those customers to be visited each day have been assigned, with either one of these two procedures, then the routes for each day are initially generated using the nearest-neighbor method. Note that more than one route may be required in those days where the number of visits and/or the accumulated load is greater than the vehicle capacity.

2.2 Phase 2. Local Search phase

In order to approximate the efficient set $\hat{E}$, the local search is focused on the Proximate Optimality Principle (POP), which states that efficient solutions are likely to be found close to other efficient solutions, in an adjacent neighborhood. The POP concept, explained in [15] and proposed by [11], is considered as a heuristic counterpart of the so-called Principle of Optimality in dynamic programming. According to [15], the interpretation of POP within multi-objective optimization is that efficient points are connected by a curve inside the efficient set, and thus an effective way to obtain a good approximation of the efficient set is to carry out a series of single-objective linked local searches (*linked* meaning that the last point of one search becomes the initial point of the next search, and *single-objective* meaning that just one of the objective functions is being optimized through the whole search) where each point visited is checked for inclusion in the efficient set $\hat{E}$. The first local search starts from the initial point generated in the Construction Phase and strives to find the optimal solution to the single-objective problem corresponding to f_1. Let x_1 be the last point visited at the end of this search. A local search is then applied again to find the best solution to the single-objective problem corresponding to f_2 using x_1 as the initial solution. Let x_2 be the last point visited at the end of this search. At this point, we again solve the problem with the first objective f_1 using x_2 as the initial solution, in order to finish a cycle around the efficient set.

In order to complete a final approximation of the efficient set $\hat{E}$, a number of single-objective local searches are implemented where compromise functions, with random weights, are employed as the new objective function, as proposed in [15]. These compromise functions strive to find efficient intermediate solutions, that is, solutions that balance the total emissions and the service frequency.

For each of these local searches, we use a set of nine different neighborhood structures, as described below:

$\mathcal{N}_1$: Within a single day, exchange the position of two randomly chosen customers, who are closer than three positions along the same route.

$\mathcal{N}_2$: Exchange the position of two randomly chosen customers on the same day, from the same or from different routes.

$\mathcal{N}_3$: Exchange the position of three randomly chosen customers on the same day, from the same or from different routes.

$\mathcal{N}_4$: Change a randomly chosen day on which a randomly chosen customer is visited.

$\mathcal{N}_5$: Move a set of consecutive randomly chosen customers to another randomly chosen route within a single day.

$\mathcal{N}_6$: Reduce the number of routes on a randomly chosen day by trying to link two different randomly chosen routes.

$\mathcal{N}_7$: Increase the service frequency of a randomly chosen customer while decreasing the frequency of service of another randomly chosen customer. This is achieved by adding the first customer to a randomly chosen route and removing the second customer from that same route.

$\mathcal{N}_8$: Increase the service frequency of a randomly chosen customer by one visit.

$\mathcal{N}_9$: Decrease the service frequency of a randomly chosen customer by one visit.

Each local search uses these neighborhoods, in the order shown, until $3|N|$ consecutive neighborhood solutions are unable to improve the current solution. Thus, instead of limiting the total number of iterations or exploring the whole set of possible neighbors in each case (which may be excessively time-consuming), each neighborhood $\mathcal{N}_j$ is allowed to generate neighbors until $3|N|$ consecutive iterations fail to improve the current solution. In other words, a neighborhood structure is employed on the condition that it improves the current solution within the last $3|N|$ trial solutions, otherwise, the following neighborhood structure, $\mathcal{N}_{j+1}$, is started until the same stopping condition is achieved. Finally, when the last neighborhood structure, $\mathcal{N}_9$, is used, the local search ends.

The pseudo-code of the MSMLS is shown in Algorithm 1.

Algorithm 1 (MSMLS)

```
1: PHASE 1. Construction of initial feasible solutions
2: S ← ∅, Ê ← ∅;
3: S ← Construct;
4: Ê ← Insert&Update(S);
5: PHASE 2. Local Search
6: for all Objective j = 1, 2 do
7:     Initialize: m ← 1 and it ← 1;
8:     repeat
9:         while it < NumIter(N_m) do
10:            local search: S' ← arg min N_m(S);
11:            if f_j(S') < f_j(S) then
12:                Ê ← Insert&Update(S') and m ← 1;
13:                it ← 1;
14:            else
15:                it ← it + 1;
16:            end if
17:        end while
18:        m ← m + 1 and it ← 1;
19:    until m = m_max;
20: end for
21: return Ê
```

As shown in the pseudo-code of Algorithm 1, an initial solution is build using the representation detailed in Section 2.1 (step 3). The solution is included in the set of efficient solutions, $\hat{E}$ (step 4). Then, the local search begins, for each objective function (step 6). The method starts from the first neighborhood (step 7) and iteratively explores the considered neighborhoods until reaching the largest neighborhood (step 19), note that we have considered $m_{max} = 9$ neighborhoods. In each iteration, while the maximum number of iterations is reached, $NumIter(\mathcal{N}_m) = 3|N|$, if the local search finds a better solution with regard to the considered objective function then the solution is checked to be included in the set of efficient solutions and otherwise we continuous with the search as explain in Section 2.2, (step 11-16). The local search ends when reaching the maximum neighborhood, $m_{max} = 9$, without improvement, and returns the set of efficient solutions (step 21).

3 The NSGA-II Algorithm

In order to assess the quality of the proposed algorithm, a competitive Multi-objective Evolutionary algorithm, NSGA-II, is adapted to our problem. NSGA-II, see [7], is an improved version of NSGA (Non-dominated Sorting Genetic Algorithm), which is widely used in the literature as one of the most suitable and robust competitors for various types of multi-objective optimization problems and, has been demonstrated to be highly effective in many situations.

NSGA-II is an elitist evolutionary method based on a non-dominated sorting approach. The solutions are represented in the form of chromosomes. Usually, a chromosome is composed of a bit string of numeric values (binary, integer or floating), whereby each value is called a gene of the chromosome. Within the NSGA-II algorithm, the population of solutions is sorted into a hierarchy of sub-populations based on the ordering of Pareto dominance. Once the population has been sorted and each objective function has been normalized, the *crowding distance* value is assigned to all solutions: the intermediate solutions take the absolute normalized difference in the function values of two adjacent solutions, while the boundary solutions, with the smallest and the largest function value, take the infinite value. The overall crowding-distance value is therefore the sum of values assigned to each objective. This crowding-distance value is employed to select good solutions, and to differentiate between them within the same ranking. Genetic operators (known as crossover and mutation) are therefore employed to recombine and create a new population of solutions. The NSGA-II algorithm repeats the procedure for a number of iterations, which are usually known as generations.

In our case, the NSGA-II algorithm has been adapted to the Bi-PVRP-SC in the following way. First of all, an initial population representing the solutions of the Bi-PVRP-SC is needed, and therefore this initial population of solutions is generated using the procedure explained in Section 2.1. In this way, we are starting from a population with the same quality of feasible solutions, in order to perform a fair comparison. Note that the size of the population is fixed to $|N| \cdot |T|$. As described earlier, the NSGA-II sorts the set of non-dominated solutions and assigns the crowding distance, which enables the elitist set of solutions to be selected. We have fixed the size of this set as half of the initial population. From this set, solutions (parents) are combined in order to create a new population of solutions (offspring) using the crossover and mutation operators.

Concerning the crossover operator, solutions are combined by selecting the service profile from one of the parents and the order in which customers are to be visited from the other parents. This way of combining solutions preserves feasibility in the offspring population. Furthermore, this strategy uses the same information as that in the proposed algorithm of this paper, MSMLS, which therefore again enables an equitable comparison.

Hence, our mutation operator is applied to each solution with the aim of exploring new solutions. This mutation operation therefore consists of applying movements using one of the nine randomly chosen neighborhoods structures, described in Subsection 2.2. This operator is applied $|N|$ consecutive times to each objective function.

When the new population is created, the process is repeated a fixed number of generations equal to the number of customers, $|N|$. In other words, this version of NSGA-II uses the same solution generator and the same solution representation as our method, and the mutation operator uses the same local search structure. This enables a fair comparison to be implemented within the two methods, NSGA-II and MSMLS.

4 Computational Results

In this section, the proposed algorithm, MSMLS, and the NSGA-II algorithm are tested on a variety of PVRP instances from the literature. Specifically, a set of 32 instances are considered. Instances p1-p10 were proposed by [8] for the VRP and adapted to the PVRP by [6]. Instance p11 was proposed by [16], instances p12 and p13 are taken from [17] and instances p14-p32 were proposed by [5]. A description of the instances can be found in Table 3. The first column indicates the instance name, the second column is the number of customers without considering the depot, the third column contains the number of vehicles, the fourth column specifies the number of days in the planning horizon, the fifth column specifies the capacity of the vehicles, and the remaining columns give the frequency information: F_{min} and F_{max} indicate the minimum and the maximum number of total visits required by all customers, and F_i is the number of customers that must be visited at least i times. For example, in instance p02, 50 customers must be served or visited using 3 vehicles per day during a planning horizon of 5 days. Each vehicle has a limited capacity of 160 units. Each customer has a minimum number of visits assigned. In total, considering the 50 customers, if the value is preset as the minimum 104 visits that must be performed by the 2 vehicles during the 5 days, and if all customers are visited every day, then a maximum of 250 visits would be required. Furthermore, there are 17 customers that need a minimum of 1 visit within the 5 days, 26 customers that require a minimum 2 visits within the 5 days, and 7 customers that impose 5 visits within the 5 days.

The experiments were conducted on a MacBook Pro 2.7 GHz Intel Core i5 and 16GN 1867 MHz DDR3, and the algorithms are implemented using C++ and executed with Xcode version 7.3.

In order to measure the quality of the solutions obtained, the indicators induced in the PISA project (http://www.tik.ee.ethz.ch/pisa) are shown since they are effective discriminators in the multi-objective scenario. Each indicator is based on different preference information, and therefore all the indicators together provide more information than simply using one single indicator. The following indicators have thus been computed:

- The number of efficient solutions found. The higher the number of efficient solutions there are in the approximation of the Pareto front, the better.
- The coverage, $C(A, B)$, calculates the proportion of solutions of algorithm B that are weakly dominated by the efficient solutions found by algorithm A, see [19]. If the coverage value is 1, then all solutions in B are weakly dominated by A. However, if the coverage is zero, no solution from B is weakly dominated by A. Consequently, $C(A, B)$ is not necessarily equal to $1 - C(B, A)$.
- The hypervolume calculates the volume in the objective space covered by the approximation of the Pareto front. The larger the volume, the better the approximation. For more details, refer to [20].
- The ϵ-indicator measures the smallest distance needed to transform every point obtained in the approximation of the Pareto front to a point that is non-dominated by a reference to the Pareto front or to the optimal front if known. The smaller the value, the better the approximation. See [18] for details.
- The $R2$ indicator is used to assess the quality of the approximation of the Pareto front set by means of utility functions that represent the preferences of the decision maker. In our case, the utility function represents the weighted Tchebychev function. A lower R2 value indicates a better approximation of the set. We refer the reader [13] for more details.
- The CPU time required to obtain the approximation of the Pareto front. A shorter computational time is preferable.

Table 4 and Table 5 show the quality of the efficient sets obtained by each algorithm as measured by the indicators previously introduced. In both tables, the first column contains the problem label. The second and third columns of Table 4 show the number of efficient points obtained by the MSMLS algorithm and the NSGA-II, respectively. The fourth column shows the proportion of points in the estimated efficient frontier of the NSGA-II that are weakly dominated by the efficient frontier of the MSMLS, and the fifth column contains the proportion of points in the estimated efficient frontier of the MSMLS that are weakly dominated by the efficient frontier of the NSGA-II. The sixth and seventh columns show the hypervolume of the MSMLS and the NSGA-II, respectively. The second and third columns of Table 5 calculate the ϵ-indicator of the MSMLS and the NSGA-II, respectively. Forth and fifth columns give the $R2$ indicator of each algorithm. Finally, the last two columns indicate the computational time (in seconds) spent executing each algorithm.

Table 4 shows that, in most of the cases, the proposed algorithm (MSMLS) obtains more efficient points than does the NSGA-II (except in instances p01, p04, p16, and p17, whose number of points are similar to those of NSGA-II). If the focus is placed on the coverage metric, in all cases C(MSMLS,NSGA-II)>C(NSGA-II,MSMLS) except in instances p07, p18, and p20, and hence the coverage metric gives clear performance advantage to MSMLS over NSGA-II. The hypervolume also gives the advantage to MSMLS over NSGA-II except in instances p07, p18, and p20. Finally, if we continuous with Table 5, the ϵ-indicator and the $R2$ indicator validate the conjecture that MSMLS outperforms NSGA-II, since MSMLS procedures have smaller R2 values than does NSGA-II in all cases. In terms of computational effort, it is worth mentioning that the CPU time for the easiest problems is similar for the two procedures and that when the complexity of the problem increases, NSGA-II is more time-consuming.

On the other hand, it is interesting to analyze the MSMLS and NSGA-II solutions in greater detail. To this end, Table 6 and Table 7 show several measures to analyze the trade-off between the total emissions and the service frequency in each of the two algorithms. In Table 6, the first column contains the problem label, the second and third columns show the extreme point of the efficient set for the MSMLS and NSGA-II, that is, the best value for the total emissions, and the fourth and fifth columns show the other extreme point of the efficient set for the MSMLS and NSGA-II, respectively, that is, the best value for the service frequency. In Table 7, the first column contains the problem label, the second and third columns compute the increment obtained in the total emissions and the forth and fifth columns calculate the increment obtained in the service frequency. Finally, the two last columns compute the average increment incurred by the total emissions each time that the frequency is incremented in a solution.

In the light of the results of Table 6, the efficient extreme points obtained by the MSMLS are always better than those obtained by the NSGA-II with regard to the total emissions (see column 2 and 3). Furthermore, observing the problem 25, we can also affirm that the efficient extreme point found by the MSMLS, (2524.49, 120), also dominates the efficient point (2733.96, 120) found by the NSGA-II. A similar behavior is observed in problem 29. However, in the remaining problems we cannot perform such a clear comparison between the efficient extreme points obtained by each algorithm. If we focus now on columns four and five where the efficient extreme points obtained by MSMLS and NSGA-II with regard to the frequency service are included, we can affirm that the MSMLS finds better efficient extreme points than the NSGA-II, in the majority of the problems except for problems 07, 16, 18, 19, 20 and 23. Nevertheless, all these efficient extreme solutions are very similar for both algorithms.

If we focus in Table 7 to analyze both algorithms, it is necessary to consider in detail each instance since each one has different characteristics, as shown in Table 3. For instance, given instance p01, it can be observed that an increment in the service frequency of 100% (double that of the minimum preset frequency), implies an increment of 72.91% in the total emissions by using the MSMLS algorithm. However, if instance p01 is solved by using the NSGA-II algorithm, an increment in the service frequency of 58.73%, implies an increment of 53.64% in the total emissions. Examining the last two columns, it can be observed that the average increment of the total emissions per frequency is lower for the MSMLS algorithm than for the NSGA-II algorithm (1.17% and 1.99%, respectively). Furthermore, we can observe that column containing the emission increment and the frequency increment for the MSMLS (columns 2 and 4) are always greater than for the NSGA-II (columns 3 and 5). That means that the MSMLS always gets larger increases than the NSGA-II when solving every problem and that indicates us that the MSMLS will converge better. This again supports the conjecture that MSMLS outperforms NSGA-II.

This section concludes with comments on Figures 1, 2, 3, and 4. These show the trade-off of the total emissions and the service frequency associated to the estimated efficient frontier obtained by MSMLS and NSGA-II. Figure 1 shows instances p01 to p08, Figure 2 represents instances p09 to p16, Figure 3 plots instances p17 to p24, and Figure 4 depicts instances p24 to p32. Here it is possible to visually detect the non-dominated solutions, although remains impossible to speak about their number or other characteristics. Clearly, if we consider Figure p05, p14, or p15, then the MSMLS approximates the Pareto front better than the NSGA-II since solutions through MSMLS dominate those of NSGA-II. However, the illustrations of instances p18 or p20 display a performance in contrast to the previous one, that is, it reveals that, in those two instances, the NSGA-II finds solutions that dominate solutions of MSMLS. Of course, there are instances in which the behavior is also similar, such as instance p17.

5 Conclusions and Future Research

The Bi-objective Periodic Vehicle Routing Problem with Service Choice has been introduced and analyzed. The problem considers two objectives: the minimization of the total emissions (the objective from the industrial enterprises' perspective, since most governments impose production suspensions on industrial enterprises) and the maximization of the service frequency to improve the quality of service (the objective from the customer's perspective). Both objectives are clearly in conflict since an increase in service (i.e., frequency of visits) typically results in an increase in total emissions given that the amount of greenhouse gas emitted by the transportation is going to be higher.

One solution of the problem consists of a schedule to visit customers during the planning horizon. We propose representing the requirements of a customer as a minimum number of visits and a capacity. This representation is very flexible and allows the modeling of a variety of practical problems, including that of waste collection. Due to the complexity of the problem, a heuristic procedure is proposed. Specifically, a multi-start multi-objective local search is developed, whose output is an approximation of the efficient frontier that can be used to trade off total emissions and service frequency.

Our computational experiments show the performance advantage of using our procedure over an adaptation of NSGA-II. The quality indicators indicate that the MSMLS outperforms the NSGA-II not only in

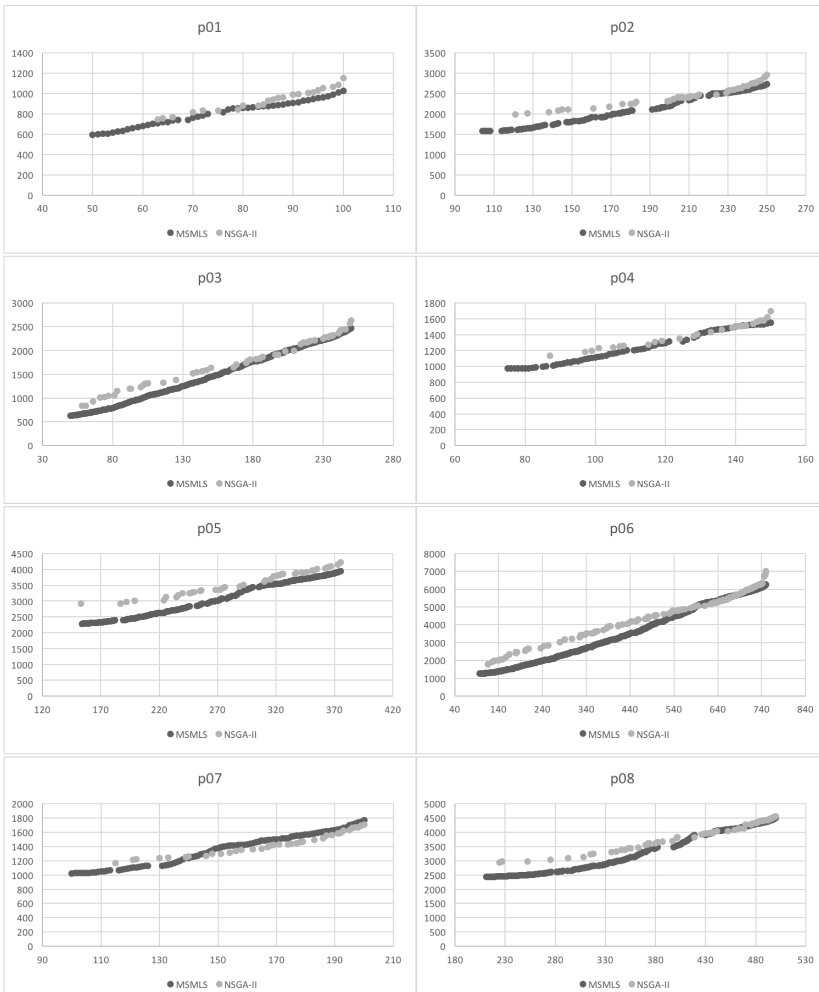

Figure 1: Trade-off between total emissions and service frequency (MSMLS and NSGA-II)

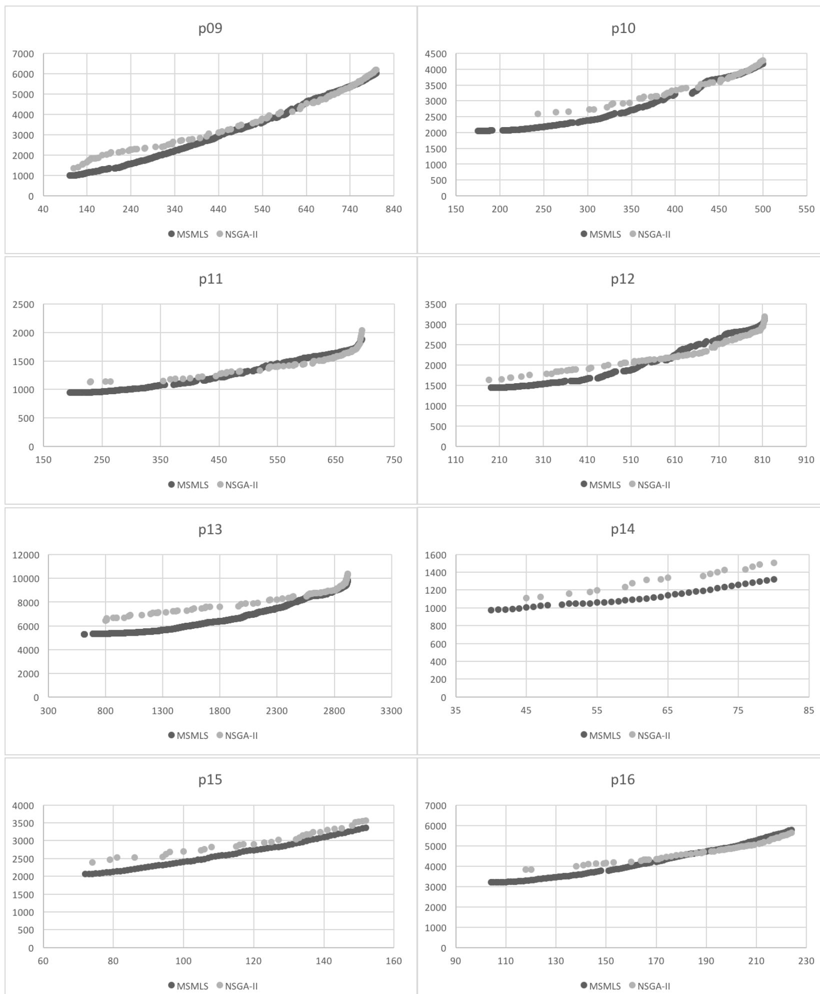

Figure 2: Trade-off between total emissions and service frequency (MSMLS and NSGA-II)

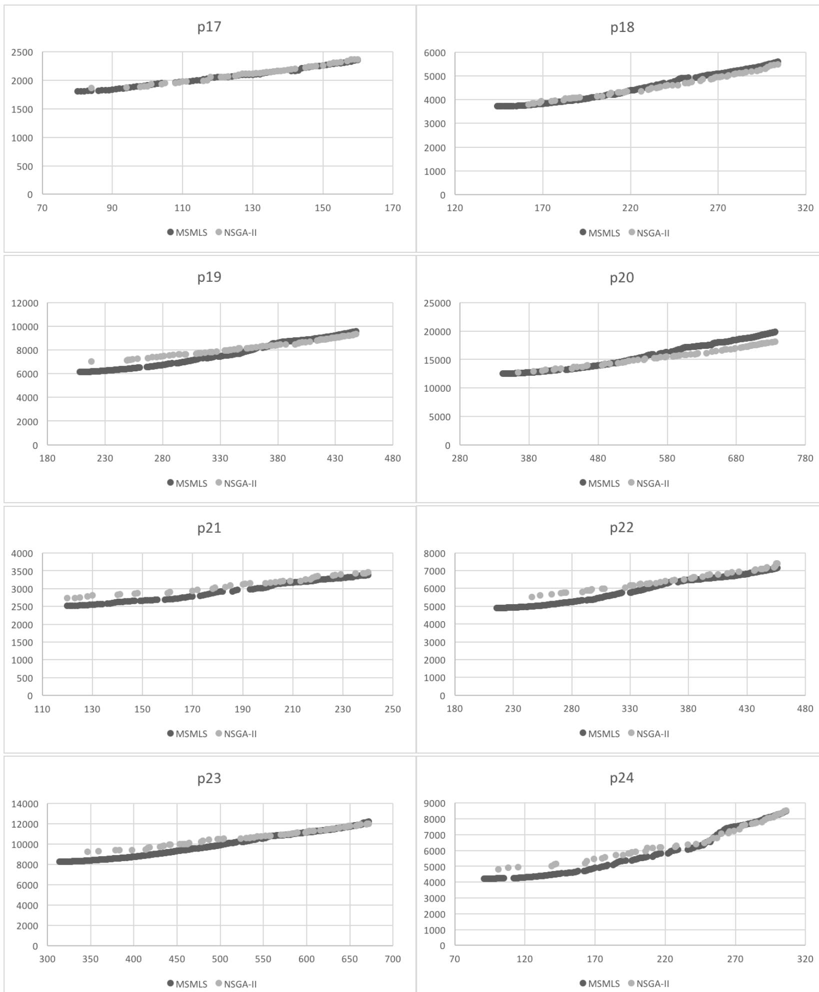

Figure 3: Trade-off between total emissions and service frequency (MSMLS and NSGA-II)

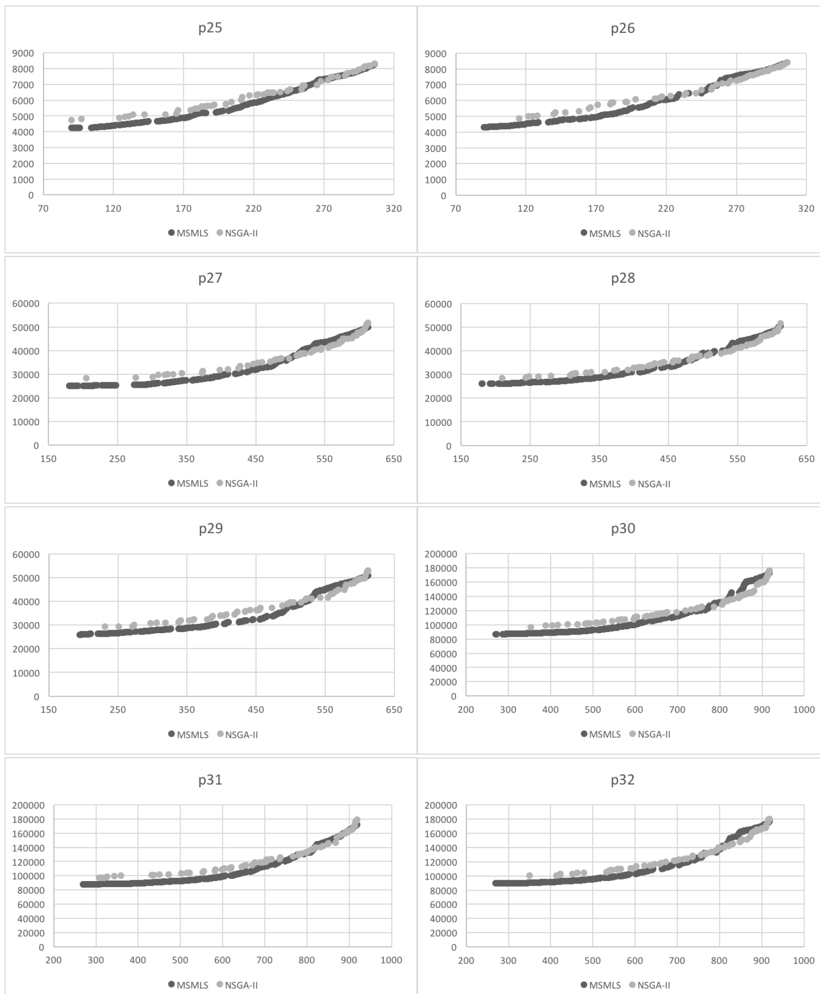

Figure 4: Trade-off between total emissions and service frequency (MSMLS and NSGA-II)

average but also in most instances. Specifically, in average, the proportion of points in the estimated efficient frontier of the NSGA-II that are weakly dominated by the efficient frontier of the MSMLS is larger than the proportion of points in the estimated efficient frontier of the MSMLS that are weakly dominated by the efficient frontier of the NSGA-II, the volume in the objective space covered by the approximation of the Pareto front obtained by the MSMLS is also greater then by the NSGA-II. Furthermore, the ϵ-indicator and the $R2$ indicators prove that the approximation of the Pareto front is better using the MSMLS. Finally, we will highlight the MSMLS finds a greater number of efficient solutions in shorter computational times (almost five times faster in average) than the NSGA-II. Our results establish the first benchmarks for this problem, which can be used in future developments and improvements.

As a follow-up to this research, we are planning to apply the proposed procedure to a waste collection problem in a city of southern Spain.

Acknowledgements

Alfredo G. Hernández-Díaz and Ana D. López-Sánchez acknowledge support from the Spanish Ministry of Economía, Industria y Competitividad through Project ECO2016-76567-C4-1-R.

References

[1] Claudia Archetti, Elena Fernández, and Diana L. Huerta-Mu noz. The flexible periodic vehicle routing problem. *Computers & Operations Research*, 85:58 – 70, 2017.

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

[3] Olli Braysy, Geir Hasle, and Wout Dullaert. A multi-start local search algorithm for the vehicle routing problem with time windows. *European Journal of Operational Research*, 159(3):586–605, 2004.

[4] Ann Melissa Campbell and Jill Hardin Wilson. Forty years of periodic vehicle routing. *Networks*, 63(1):2–15, 2014.

[5] I-Ming Chao, Bruce L. Golden, and Edward Wasil. An improved heuristic for the period vehicle routing problem. *Networks*, 26(1):25–44, 1995.

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

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[8] Samuel Eilon, C. Watson-Gandy, Nicos Christofides, and Richard de Neufville. Distribution management-mathematical modelling and practical analysis. *Systems, Man and Cybernetics, IEEE Transactions on*, 21:589 – 589, 12 1974.

[9] Peter Francis, Karen Smilowitz, and Michal Tzur. The period vehicle routing problem with service choice. *Transportation Science*, 40(4):439–454, November 2006.

[10] M. Gaudioso and G. Paletta. A heuristic for the periodic vehicle routing problem. *Transportation Science*, 26(2):86–92, 1992.

[11] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[12] Jos Rubn Gómez, Joaqun Pacheco, and Hernn Gonzalo-Orden. A tabu search method for a bi-objective urban waste collection problem. *Computer-Aided Civil and Infrastructure Engineering*, 30(1):36–53, 2015.

[13] Michael Pilegaard Hansen and Andrzej Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set, 1998.

[14] Rafael Martí, Ricardo Aceves, Maria Teresa León, Jose M. Moreno-Vega, and Abraham Duarte. *Intelligent Multi-Start Methods*, pages 221–243. Springer International Publishing, 2019.

[15] Julián Molina, Manuel Laguna, Rafael Marti, and Rafael Caballero. Sspmo: A scatter tabu search procedure for non-linear multiobjective optimization. *INFORMS Journal on Computing*, 19(1):91–100, 2007.

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

[17] Robert A. Russell and Dave Gribbin. A multiphase approach to the period routing problem. *Networks*, 21(7):747–765, 1991.

[18] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.

[19] Eckart Zitzler. Evolutionary algorithms for multiobjective optimization: Methods and applications, 1999.

[20] Eckart Zitzler and Lothar Thiele. *Multiobjective optimization using evolutionary algorithms — A comparative case study*, pages 292–301. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

Instance	Customers	Vehicles	Days	Capacity	F_{min}	F_{max}	F_1	F_2	F_3	F_4	F_5	F_6
p01	50	3	2	160	50	100	50					
p02	50	3	5	160	104	250	17	26			7	
p03	50	1	5	160	50	250	50					
p04	75	2	5	140	75	150	75					
p05	75	6	5	140	153	375	30	34			11	
p06	75	1	10	140	75	750	75					
p07	100	4	2	200	75	200	100					
p08	100	5	5	200	156	500	40	46			14	
p09	100	1	8	200	75	800	100					
p10	100	4	5	200	132	500	40	46	14			
p11	139	4	5	235	192	695	103	22	12	1	1	
p12	163	3	5	140	185	815	148	8	7			
p13	417	9	7	2000	457	2919	337	40				
p14	20	2	4	20	40	80	8	8		4		
p15	38	2	4	30	72	152	16	16		6		
p16	56	2	4	40	104	224	24	24		8		
p17	40	4	4	20	80	160	16	16		8		
p18	76	4	4	30	144	304	32	32		12		
p19	112	4	4	40	208	448	48	48		16		
p20	184	4	4	60	336	736	80	80		24		
p21	60	6	4	20	120	240	24	24		12		
p22	114	6	4	30	216	456	48	48		18		
p23	168	6	4	40	312	672	72	72		24		
p24	51	3	6	20	90	306	36	9				6
p25	51	3	6	20	90	306	36	9				6
p26	51	3	6	20	90	306	36	9				6
p27	102	6	6	20	180	612	72	19				12
p28	102	6	6	20	180	612	72	19				12
p29	102	6	6	20	180	612	72	19				12
p30	153	9	6	20	279	918	108	27				18
p31	153	9	6	20	270	918	108	27				18
p32	153	9	6	20	270	918	108	27				18
Average	102.5	4.4	5.15	143.3	157.5	540.1						

Table 3: Instance Description

	Number Eff.		Coverage		Hypervolume	
	MSMLS	NSGA-II	C(MSMLS,NSGA-II)	C(NSGA-II, MSMLS)	MSMLS	NSGA-II
p01	48	52	0.9615	0.0625	0.6285	0.5468
p02	112	111	0.9910	0.0357	0.6620	0.1189
p03	194	138	0.9565	0.0567	0.5821	0.5099
p04	69	97	0.9583	0.0870	0.6790	0.6036
p05	190	177	1.0000	0.0000	0.6557	0.4818
p06	573	395	0.8937	0.1483	0.6194	0.5341
p07	94	81	0.2222	0.6064	0.5751	0.5848
p08	239	211	0.9479	0.1297	0.6197	0.4895
p09	605	379	0.8486	0.1983	0.5935	0.5409
p10	264	225	0.8667	0.1364	0.6640	0.5561
p11	395	306	0.4052	0.3924	0.7238	0.6892
p12	481	371	0.4690	0.3389	0.6735	0.6219
p13	1603	1110	1.0000	0.0000	0.7336	0.5445
p14	40	32	1.0000	0.0000	0.7378	0.4544
p15	81	72	1.0000	0.0000	0.6386	0.4779
p16	118	122	0.3934	0.3220	0.6132	0.5294
p17	72	95	0.8000	0.2361	0.6593	0.6412
p18	150	146	0.3219	0.5400	0.6422	0.6607
p19	218	214	0.5421	0.3349	0.5723	0.4726
p20	367	316	0.3703	0.5940	0.5950	0.6665
p21	108	105	1.0000	0.0000	0.6861	0.5586
p22	222	202	1.0000	0.0000	0.6227	0.5036
p23	333	258	0.9380	0.0571	0.5822	0.4638
p24	178	135	0.7333	0.2360	0.6566	0.5756
p25	187	143	0.9371	0.0909	0.6749	0.5987
p26	179	169	0.6095	0.2905	0.6247	0.5578
p27	333	279	0.5412	0.2973	0.7426	0.6801
p28	313	310	0.6097	0.2684	0.7208	0.6788
p29	311	264	0.7462	0.2283	0.7364	0.6676
p30	464	419	0.6301	0.1983	0.7617	0.7123
p31	521	439	0.8633	0.1420	0.7819	0.7075
p32	447	408	0.6275	0.1834	0.7635	0.7081
Average	**297.2**	243.2	**0.7558**	0.1941	**0.6632**	0.5668

Table 4: Indicators to measure the performance between MSMLS and NSGA-II

	ϵ-indicator		R2		CPU time	
	MSMLS	NSGA-II	MSMLS	NSGA-II	MSMLS	NSGA-II
p01	< 0.0001	0.2773	< 0.0001	0.2004	36.3	20.1
p02	< 0.0001	0.2834	< 0.0001	0.1411	71.6	119.6
p03	< 0.0001	0.1019	< 0.0001	0.0552	208.0	136.9
p04	< 0.0001	0.2007	< 0.0001	0.1350	187.6	84.0
p05	< 0.0001	0.2904	< 0.0001	0.1399	208.3	625.3
p06	< 0.0001	0.0892	< 0.0001	0.0434	889.8	2088.1
p07	< 0.0001	0.1962	< 0.0001	0.1298	111.5	298.5
p08	< 0.0001	0.2337	< 0.0001	0.1145	371.0	2247.5
p09	< 0.0001	0.0710	< 0.0001	0.0336	1864.5	6265.0
p10	< 0.0001	0.2453	< 0.0001	0.1469	470.0	1704.3
p11	< 0.0001	0.1653	< 0.0001	0.0921	2607.7	10779.0
p12	< 0.0001	0.1111	< 0.0001	0.0505	3846.0	41567.2
p13	< 0.0001	0.2226	< 0.0001	0.1132	43963.6	67157.0
p14	< 0.0001	0.2502	< 0.0001	0.1447	10.3	7.1
p15	< 0.0001	0.2123	< 0.0001	0.1051	46.3	102.2
p16	< 0.0001	0.2433	< 0.0001	0.1227	129.6	450.7
p17	< 0.0001	0.0789	< 0.0001	0.0487	40.0	136.2
p18	< 0.0001	0.0500	< 0.0001	0.0327	108.0	1879.6
p19	< 0.0001	0.2627	< 0.0001	0.1319	239.2	6761.8
p20	< 0.0001	0.0677	< 0.0001	0.0352	3660.5	59323.8
p21	< 0.0001	0.1714	< 0.0001	0.0827	83.9	597.9
p22	< 0.0001	0.2279	< 0.0001	0.1295	1230.5	7514.7
p23	< 0.0001	0.2539	< 0.0001	0.1219	5969.6	32290.9
p24	< 0.0001	0.1334	< 0.0001	0.0675	168.0	745.5
p25	< 0.0001	0.1093	< 0.0001	0.0521	178.7	682.4
p26	< 0.0001	0.1337	< 0.0001	0.0834	207.6	682.4
p27	< 0.0001	0.1143	< 0.0001	0.0572	1568.3	10778.8
p28	< 0.0001	0.0973	< 0.0001	0.0619	1568.3	10614.6
p29	< 0.0001	0.1156	< 0.0001	0.0744	1330.3	10315.8
p30	< 0.0001	0.1204	< 0.0001	0.0821	2223.5	10315.8
p31	< 0.0001	0.0900	< 0.0001	0.0503	3944.1	49801.2
p32	< 0.0001	0.1094	< 0.0001	0.0590	2554.0	59062.6
Average	**¡0.0001**	0.1666	**¡0.0001**	0.0918	**2503.0**	12348.6

Table 5: Indicators to measure the performance between MSMLS and NSGA-II

	MSMLS (Emissions*, Frequency)	NSGA-II (Emissions*, Frequency)	MSMLS (Emissions, Frequency*)	NSGA-I (Emissions, Freq
p01	(593.22, 50)	(746.93, 63)	(1025.72, 100)	(1147.54, 1
p02	(1580.20, 104)	(1981.89, 121)	(2721.64, 250)	(2950.27, 2
p03	(629.46, 50)	(833.89, 58)	(2470.59, 250)	(2629.85,25
p04	(970.39, 75)	(1136.91, 87)	(1553.45, 150)	(1699.21, 1
p05	(2283.05, 154)	(2915.60, 153)	(3951.74, 375)	(4223.07, 3
p06	(1262.19, 97)	(1787.62, 116)	(6257.51, 750)	(6993.18, 7
p07	(1023.77, 100)	(1169.61, 115)	(1767.18, 200)	(1708.70, 2
p08	(2440.79, 211)	(2941.43, 224)	(4498.65, 500)	(4549.07, 5
p09	(991.60, 100)	(1365.28, 109)	(6036.37, 800)	(6202.95, 8
p10	(2054.07, 175)	(2597.34, 243)	(4178.05, 500)	(4266.33, 5
p11	(942.67, 195)	(1132.99, 230)	(1883.14, 695)	(2032.91, 6
p12	(1440.28, 190)	(1637.19, 185)	(3134.39, 815)	(3185.58, 8
p13	(5304.15, 612)	(6455.21, 800)	(9815.12, 2919)	(10361.60, 2
p14	(976.55, 40)	(1109.06, 45)	(1319.06, 80)	(1506.19, 8
p15	(2069.01,72)	(2393.44, 74)	(3360.18, 152)	(3569.22, 1
p16	(3207.40, 104)	(3830.31, 118)	(5768.05, 224)	(5644.70, 2
p17	(1808.48, 80)	(1864.36, 84)	(2359.38, 160)	(2372.65, 1
p18	(3724.75, 144)	(3800.73, 162)	(5608.91, 304)	(5478.29, 3
p19	(6143.98, 208)	(7043.12, 218)	(9566.83, 448)	(9318.74, 4
p20	(12544.50, 342)	(12688.30, 364)	(19862.40, 736)	(18164.10, 7
p21	(2524.49, 120)	(2733.96, 120)	(3385.82, 240)	(3464.22, 2
p22	(4900.17, 216)	(5507.38, 246)	(7152.73, 456)	(7415.22, 4
p23	(8261.21, 314)	(9257.23, 346)	(12183.40, 672)	(12008.10, 6
p24	(4220.16, 91)	(4804.41, 101)	(8479.17, 306)	(8542.15, 3
p25	(4247.58, 90)	(4734.39, 90)	(8268.39, 306)	(8334.01, 3
p26	(4317.24, 90)	(4862.32, 115)	(8390.96, 306)	(8393.96, 3
p27	(25142.50, 180)	(28316.00, 205)	(49965.60, 612)	(51849.80, 6
p28	(25975.90, 180)	(28497.30, 209)	(50447.70, 612)	(51457.90, 6
p29	(25894.90, 195)	(29268.00, 231)	(50988.60, 612)	(52973.10, 6
p30	(87039.20, 270)	(96383.90, 353)	(173538.00, 918)	(176275.00, 9
p31	(88136.60, 270)	(96853.30, 307)	(172466.00, 918)	(178676.00, 9
p32	(89570.50, 270)	(100201.00, 350)	(177120.00, 918)	(180029.00, 9
Average	(13194.40, 168.41)	(14714.08, 195.06)	(25610.15, 540.13)	(26169.46, 54

Table 6: Trade-off analysis (extreme values of the objective functions)

	MSMLS Emission	NSGA-II Increment	MSMLS Frequency	NSGA-II Increment	MSMLS Emissions/Frequency	NSGA-II
p01	72.91%	53.64%	100.00%	58.73%	1.17%	1.99%
p02	72.23%	48.86%	140.38%	106.61%	0.49%	1.00%
p03	292.50%	215.37%	400.00%	331.03%	0.71%	2.31%
p04	60.09%	49.46%	100.00%	72.41%	0.70%	1.63%
p05	73.09%	44.84%	143.51%	145.10%	0.29%	0.96%
p06	395.77%	291.20%	673.20%	546.55%	0.28%	1.10%
p07	72.61%	46.09%	100.00%	73.91%	0.59%	1.06%
p08	84.31%	54.66%	136.97%	123.21%	0.26%	0.77%
p09	508.75%	354.34%	700.00%	633.94%	0.30%	1.36%
p10	103.40%	64.26%	185.71%	105.76%	0.27%	0.85%
p11	99.77%	79.43%	256.41%	202.17%	0.18%	0.80%
p12	117.62%	94.58%	328.95%	340.54%	0.16%	0.73%
p13	85.05%	60.52%	376.96%	264.88%	0.04%	0.50%
p14	35.07%	35.81%	100.00%	77.78%	0.77%	1.82%
p15	62.41%	49.13%	111.11%	105.41%	0.61%	1.26%
p16	79.84%	47.37%	115.38%	89.83%	0.50%	0.71%
p17	30.46%	27.26%	100.00%	90.48%	0.38%	0.50%
p18	50.58%	44.14%	111.11%	87.65%	0.28%	0.61%
p19	55.71%	32.31%	115.38%	105.50%	0.20%	0.38%
p20	58.34%	43.16%	115.20%	102.20%	0.13%	0.48%
p21	34.12%	26.71%	100.00%	100.00%	0.27%	0.63%
p22	45.97%	34.64%	111.11%	85.37%	0.17%	0.62%
p23	47.48%	29.72%	114.01%	94.22%	0.12%	0.38%
p24	100.92%	77.80%	236.26%	202.97%	0.40%	1.16%
p25	94.66%	76.03%	240.00%	240.00%	0.36%	1.10%
p26	94.36%	72.63%	240.00%	166.09%	0.37%	0.95%
p27	98.73%	83.11%	240.00%	198.54%	0.21%	1.05%
p28	94.21%	80.57%	240.00%	192.82%	0.21%	0.78%
p29	96.91%	80.99%	213.85%	164.94%	0.22%	1.00%
p30	99.38%	82.89%	240.00%	160.06%	0.15%	0.76%
p31	95.68%	84.48%	240.00%	199.02%	0.13%	0.83%
p32	97.74%	79.67%	240.00%	162.29%	0.15%	0.83%
Average	**106.58%**	79.55%	**214.55%**	175.94%	**0.35%**	0.97%

Table 7: Increment of emissions and frequency

	Min emissions	Min freq	Max emissions	Max freq	Emission Increment	Freq Increment
p01	593.22	50	1025.72	100	72.91%	100.00%
p02	1580.20	104	2721.64	250	72.23%	140.38%
p03	629.46	50	2470.59	250	292.50%	400.00%
p04	970.39	75	1553.45	150	60.09%	100.00%
p05	2283.05	154	3951.74	375	73.09%	143.51%
p06	1262.19	97	6257.51	750	395.77%	673.20%
p07	1023.77	100	1767.18	200	72.61%	100.00%
p08	2440.79	211	4498.65	500	84.31%	136.97%
p09	991.60	100	6036.37	800	508.75%	700.00%
p10	2054.07	175	4178.05	500	103.40%	185.71%
p11	942.67	195	1883.14	695	99.77%	256.41%
p12	1440.28	190	3134.39	815	117.62%	328.95%
p13	5304.15	612	9815.12	2919	85.05%	376.96%
p14	976.55	40	1319.06	80	35.07%	100.00%
p15	2069.01	72	3360.18	152	62.41%	111.11%
p16	3207.40	104	5768.05	224	79.84%	115.38%
p17	1808.48	80	2359.38	160	30.46%	100.00%
p18	3724.75	144	5608.91	304	50.58%	111.11%
p19	6143.98	208	9566.83	448	55.71%	115.38%
p20	12544.50	342	19862.40	736	58.34%	115.20%
p21	2524.49	120	3385.82	240	34.12%	100.00%
p22	4900.17	216	7152.73	456	45.97%	111.11%
p23	8261.21	314	12183.40	672	47.48%	114.01%
p24	4220.16	91	8479.17	306	100.92%	236.26%
p25	4247.58	90	8268.39	306	94.66%	240.00%
p26	4317.24	90	8390.96	306	94.36%	240.00%
p27	25142.50	180	49965.60	612	98.73%	240.00%
p28	25975.90	180	50447.70	612	94.21%	240.00%
p29	25894.90	195	50988.60	612	96.91%	213.85%
p30	87039.20	270	173538.00	918	99.38%	240.00%
p31	88136.60	270	172466.00	918	95.68%	240.00%
p32	89570.50	270	177120.00	918	97.74%	240.00%
Average	13194.40	168.41	25610.15	540.13	106.58%	214.55%

Table 8: Trade-off analysis of the MSMLS algorithm

	Min emissions	Min freq	Max emissions	Max freq	Emissions Increment	Freq Increment
p01	746.93	63	1147.54	100	53.64%	58.73%
p02	1981.89	121	2950.27	250	48.86%	106.61%
p03	833.89	58	2629.85	250	215.37%	331.03%
p04	1136.91	87	1699.21	150	49.46%	72.41%
p05	2915.60	153	4223.07	375	44.84%	145.10%
p06	1787.62	116	6993.18	750	291.20%	546.55%
p07	1169.61	115	1708.70	200	46.09%	73.91%
p08	2941.43	224	4549.07	500	54.66%	123.21%
p09	1365.28	109	6202.95	800	354.34%	633.94%
p10	2597.34	243	4266.33	500	64.26%	105.76%
p11	1132.99	230	2032.91	695	79.43%	202.17%
p12	1637.19	185	3185.58	815	94.58%	340.54%
p13	6455.21	800	10361.60	2919	60.52%	264.88%
p14	1109.06	45	1506.19	80	35.81%	77.78%
p15	2393.44	74	3569.22	152	49.13%	105.41%
p16	3830.31	118	5644.70	224	47.37%	89.83%
p17	1864.36	84	2372.65	160	27.26%	90.48%
p18	3800.73	162	5478.29	304	44.14%	87.65%
p19	7043.12	218	9318.74	448	32.31%	105.50%
p20	12688.30	364	18164.10	736	43.16%	102.20%
p21	2733.96	120	3464.22	240	26.71%	100.00%
p22	5507.38	246	7415.22	456	34.64%	85.37%
p23	9257.23	346	12008.10	672	29.72%	94.22%
p24	4804.41	101	8542.15	306	77.80%	202.97%
p25	4734.39	90	8334.01	306	76.03%	240.00%
p26	4862.32	115	8393.96	306	72.63%	166.09%
p27	28316.00	205	51849.80	612	83.11%	198.54%
p28	28497.30	209	51457.90	612	80.57%	192.82%
p29	29268.00	231	52973.10	612	80.99%	164.94%
p30	96383.90	353	176275.00	918	82.89%	160.06%
p31	96853.30	307	178676.00	918	84.48%	199.02%
p32	100201.00	350	180029.00	918	79.67%	162.29%
Average	14714.08	195.06	26169.46	540.13	79.55%	175.94%

Table 9: Trade-off analysis of the NSGA-II algorithm