Multi-Objective Evolutionary Algorithms for Vehicle Routing Problems

by
Abel Garcia Najera

A thesis submitted to
The University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
The University of Birmingham
November 2010

Abstract

The Vehicle Routing Problem, which main objective is to find the lowest-cost set of routes to deliver goods to customers, has many applications in transportation services. In the past, costs have been mainly associated to the number of routes and the travel distance, however, in real-world problems there exist additional objectives.

Since there is no known exact method to efficiently solve the problem in polynomial time, many heuristic techniques have been considered, among which, evolutionary methods have proved to be suitable for solving the problem. Despite this method being able to provide a set of solutions that represent the trade-offs between multiple objectives, very few studies have concentrated on the optimisation of more than one objective, and even fewer have explicitly considered the diversity of solutions, which is crucial for the good performance of any evolutionary computation technique.

This thesis proposes a novel Multi-Objective Evolutionary Algorithm to solve two variants of the Vehicle Routing Problem, regarding the optimisation of at least two objectives. This approach incorporates a method for measuring the similarity of solutions, which is used to enhance population diversity, and operators that effectively explore and exploit the search space. The algorithm is applied to typical benchmark problems and empirical analyses indicate that it efficiently solves the variants being studied. Moreover, the proposed method has proved to be competitive with recent approaches and outperforms the successful multi-objective optimiser NSGA-II.

To Jerica, my life.

Acknowledgements

When I started this project I did not know what specific direction to follow. Fortunately, there was my supervisor who taught me how to identify promising ways. Thank you very much John for having given me *guided freedom*.

I could not have evaluated my research myself, that is why I want to thank my viva examiners, Prof. Xin Yao and Prof. Graham Kendall, for having taken the time to assess my work.

It is always important to count with the support from people around us, specially from those one loves. Thank you very much Jerica for having left everything behind only to be with me, and for having shared good and bad times, pints of excelent lager and ale, and unpredictable weather. I really appreciate all the things you did.

In the same context, love and support, I want to thank my mum, sisters and brothers, they were aware of my business at all times. Thank you Mum, Beti, Marilú, Héctor and Felipe, you are always in my mind and heart.

When time does not look good, it is healthy to take a break, perhaps to have a pint or two after a football match and then head for a gig or a party. Thank you Gina, Nancy, Agustín, Alhuey, Anaxi and Rodrigo, you made my time when I was incomplete.

This academic adventure would not have been possible without the contribution from the Mexican people and support from the Mexican Council of Science and Technology (CONACYT), from which I was the recipient of the scholarship number 179372/229168. Thank you very much Mexico. Thank you very much CONACYT.

The study contained in this thesis was accepted for presentation in several international workshops and conferences. I would like to thank the School of Computer Science Research Committee for the travel grants I was allocated for attending those events and presenting my work.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The Vehicle Routing Problem (VRP) is one of the most important and widely studied combinatorial optimisation problems, because of the many real-world applications in delivery and transportation logistics [14]. Furthermore, this problem is of great importance in distribution systems, as the transportation process represents between 10% and 20% of the final cost of goods [236]. Additionally, it is estimated that distribution costs account for almost half of the total logistics costs [56].

The objective of the VRP is to obtain the lowest-cost set of routes to deliver demand to customers. But what does *lowest-cost* mean? Since the problem was proposed by Dantzig and Ramser [54] as a generalisation of the Travelling Salesman Problem [123], cost has mostly been associated with the number of routes and the travel distance, but, evidently, there are several other sources of cost, e.g. driver's remuneration and workload imbalance [135].

The VRP has many variants that take into account different constraints, which can be categorised as operational and of precedence. In particular, the *Capacitated* VRP (CVRP) considers vehicles with restricted capacity, and in the VRP *with Time Windows* (VRPTW), in addition to the limited capacity, vehicles must visit customers within specific times. With the introduction of these restrictions, the optimisation of one objective will probably not correspond to the optimisation of all of them. Moreover, the optimisation of one objective could lead to the deterioration of the others. In this scenario, one needs to consider the simultaneous optimisation of all the objectives, and the process needs to provide a range of solutions that represent the trade-offs between the objectives, rather than a single solution.

Optimum solutions for small instances of VRPs can be obtained using exact methods, however, the computation time required increases considerably for larger instances [71]. As a matter of fact, the VRP is NP-hard [159]. These are the main reasons why many published research studies have considered the use of heuristic and metaheuristic methods. Although numerous techniques have been proposed, those studies considering evolutionary computation methods have been among the most suitable approaches for tackling some variants of the VRP [28].

Evolutionary Algorithms (EA) have been successful in many practical situations [254]. They are based on Darwin's theory of evolution by natural selection: A *population* (set) of *individuals* (solutions) is maintained, and the EA *selects*, *recombines*, and *mutates* the *fittest* (best solutions) in the hope of producing *offspring* (new solutions) of increased *fitness* (quality), which may replace the least fit. The evolutionary operations are repeated until the quality of solutions stops increasing, or some fixed number of *generations* (cycles) has been reached.

EAs population-based approach permits them to find multiple optimum solutions in one single run, which is especially useful in the context of multi-objective optimisation, since more than one optimum solution is required [62]. The idea of optimality in multiple objective problems concerns solutions that are as good as all other solutions in each objective function and strictly better than the others in at least one of the objectives [44].

One of the key factors in the success of an EA is the ability to avoid premature convergence, which is the effect of losing population diversity and getting stuck in sub-optimum solutions. This can be achieved with an appropriate trade-off between exploration and exploitation of the search space [81]. Actually, maintaining diversity has been considered as an additional objective to be pursued while solving optimisation problems [232].

The concept of population diversity is not only related to how many distinct solutions there are in the population, information that is easy to access, but also to how different the solutions are between them. In fact, the term is often used without definition and the implicit assumption is the diversity of *genotypes*, or structural diversity [33]. In order to quantify this, one would require the use of specialised tools, which could be utilised to boost population diversity.

Many existing well-known and successful evolutionary approaches do have diversity preservation tools, but their use for solving VRPs would not be appropriate because they require the definition of niche spaces, a task that would be problematic since most good feasible solutions to the VRP reside in a very small region of the number of routes dimension [182].

In most situations, population diversity is inversely proportional to the similarity between the solutions comprised in the population, or directly proportional if the difference or distance between them is considered. To quantify population diversity, one generally must take into account the solution encoding, since different solution representations may require different similarity or distance measures [221]. Furthermore, for some kinds of problems, if similarity or distance are considered in the objective space, one could be led to unreliable measures.

There exist in the literature many similarity and distance measures for specific solution encodings. For example, the *Hamming distance* [125], which measures the minimum number of reversals required to change one binary string into another, is the most commonly used method for binary representations. For real-number encodings, the *Manhattan*, *Euclidean*, or *Chebychev* distances [149, 73] are preferred. However, for problems for which solutions are most suitably represented as a permutation of a set of items, there is no agreed measure and several techniques have been proposed. From these, the *edit distance* [247] is of particular interest to the present study, since it has been suggested as a distance measure between solutions to routing problems [219].

One of the objectives to be pursued with this research is to fulfil the identified gaps and complement previous studies regarding, primarily, population diversity preservation, the simultaneous minimisation of at least two objectives, and a appropriate multi-objective analysis by means of multi-objective performance metrics.

1.1 Motivation

Recently, great attention has been devoted to complex variants of the VRP that are closer to the practical distribution applications that the problems model [14]. In this scenario, a high percentage of the evolutionary proposals for solving the CVRP, VRPTW, and other VRP variants are hybrid approaches, i.e. a combination of an EA with another heuristic or metaheuristic method. Actually, the first questions that arose in the present research were *why those studies were combining an EA with another technique*, and *if the reason is that EAs cannot solve the problem by themselves.*

An exploratory study, revealed that a simple EA suffered from premature lack of population diversity, and consequently got stuck in sub-optimum solutions. Thus new research questions appeared: *Do we really need to combine the EA with another heuristic method to preserve population diversity? Or can we think of another* *strategy which can be incorporated as part of the EA in order to diversify the population? Perhaps we could measure diversity or similarity and use this information somewhere in the algorithm?*

Sörensen [219] stated that the edit distance could be used to quantify the difference between two solutions to routing problems, however we need to know *how difficult the implementation of this method is*, and *what its performance is when compared with other possibilities*. In the same context, *is there an alternative technique, or can it be designed, that performs better that the edit distance?*

On the other hand, the vast majority of the published research studies which utilise heuristics or metaheuristics for solving VRPs takes into account the optimisation of one single objective and only a small percentage considers more than one. Furthermore, from the latter, very few studies present and analyse their results in a proper multi-objective manner. In fact, they are generally compared according to the best results regarding exclusively one of the objectives being optimised, analysis which could be misleading for the optimisation of multiple objectives.

The facts just described, together with the research questions, were the main motivations for studying two of the variants of the VRP, regarding the simultaneous optimisation of multiple objectives, and for designing an EA which incorporates a mechanism to measure solution similarity and uses this information to enhance population diversity.

1.2 Research objectives

The research objectives are set according to, and to fulfil, the gaps detected in previously published studies regarding solutions to the CVRP and VRPTW. In general, these findings are related to: (*i*) the inclusion of heuristic methods as additional stages in EAs for preserving population diversity, (*ii*) the insufficiency of studies considering the optimisation of multiple objectives, and (*iii*) the inappropriate presentation and analysis of the results in multi-criterion studies.

Regarding the first issue, it is crucial to know how diverse the population is in order to increase it or maintain it. Moreover, the information about the similarity between two solutions and between one solution and the rest of the population could be of potential use in order to achieve this goal.

With respect to the other two topics, the flexibility to optimise other objectives, in addition to the the number of routes and travel distance, is necessary. Consequently, proper presentation and analysis of results are expected, which must consider the use of multi-objective performance metrics.

To accomplish what have just been stated, the objective of this research takes into consideration the design and development of an EA for the effective solution of the CVRP and VRPTW regarding the optimisation of multiple objectives. Therefore, the performance of the developed EA must be analysed in a suitable multi-criterion manner by means of using proper multi-objective performance metrics.

Additionally, this algorithm does not have to be combined with any other heuristic method with the purpose of escaping from sub-optimum regions. Instead, if population diversity and solution similarity are known, this information may be used in order to have a suitable exploration and exploitation of the search space.

1.3 Contributions

The study presented here has made the following major contributions.

A measure has been developed to quantify how similar two solutions to the VRP are, and this information, in turn, is used to determine the similarity between one solution and the rest of the population, and the population diversity. The advantages of this measure are: it does not depend on how solutions are represented, it can be used for any variant of the VRP, and it has a linear time complexity.

A mutation process has been designed, which, when incorporated in an Evolutionary Algorithm for solving the VRP, helps to better exploit the search space. This mutation process includes three basic functions, two of which are stochastic, to select routes and customers, and the other is deterministic, to reinsert customers into routes, and a set of three operators which make modifications in the assignment of customers to routes and in the sequence of service within a route.

A multi-objective Evolutionary Algorithm that effectively solves the Capacitated VRP and the VRP with Time Windows, regarding the optimisation of at least two objectives, has been formulated. This algorithm includes the similarity measure and the mutation process mentioned above in order to better explore and exploit the search space, consequently it preserves a higher population diversity. The outcome of this algorithm provides solutions that suitably represent the trade-offs between the objectives.

The solutions from the proposed algorithm to test instances of the CVRP and VRPTW were compared with those from previous studies from a single-objective point of view, showing that, although they are not the overall best, they are comparable to many, in the sense that they have less number of routes, or shorter travel distance, while keeping the other objective equal or within 2% difference. This type of comparison is often misleading however, since the best result for one objective does not necessarily represent the multi-objective performance of an optimiser.

The solutions from the bi-objective and tri-objective optimisations of the VRP with Time Windows were analysed by means of using three multi-objective quality indicators and compared with those from the popular multi-objective optimiser NSGA-II, resulting in that the former has a better performance than the latter in many test problems. In the case of the CVRP, a multi-objective analysis could not be performed, since the objectives in the test instances are not in conflict.

Since many previous VRP studies that regarded the optimisation of multiple criteria reported their performance analysis in a single-objective manner, the research enclosed in this thesis is one of the very few that presents an appropriate multi-objective performance analysis by means of the utilisation of formal evaluation techniques.

1.4 Publications resulting from this thesis

In addition to the contributions above, the following is the list of the publications related to this investigation.

Chapter 4:

A. Garcia-Najera and J. A. Bullinaria. A multi-objective density restricted genetic algorithm for the vehicle routing problem with time windows. In *2008 UK Workshop on Computational Intelligence*, 2008. [Section 4.2]

A. Garcia-Najera and J. A. Bullinaria. Bi-objective optimization for the vehicle routing problem with time windows: Using route similarity to enhance performance. In M. Ehrgott, C. Fonseca, X. Gandibleux, J. K. Hao, and M. Sevaux, editors, *5th International Conference on Evolutionary Multi-Criterion Optimization*, volume 5467 of LNCS, pages 275–289. Springer, 2009, [Sections 4.3.1, 4.3.2.3, and 4.3.2.4]

A. Garcia-Najera. Preserving population diversity for the multi-objective vehicle routing problem with time windows. In Franz Rothlauf editor, *Genetic and Evolutionary Computation Conference 2009*, pages 2689–2692. ACM, 2009. [Sections 4.3.1.2 and 4.3.2.2]

A. Garcia-Najera and J. A. Bullinaria. Comparison of similarity measures for the multi-objective vehicle routing problem with time windows. In Franz Rothlauf editor, *Genetic and Evolutionary Computation Conference 2009*, pages 579–586. ACM, 2009. [Section 4.3.2.1]

Chapter 5:

A. Garcia-Najera and J. A. Bullinaria. An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, 28(1):287-300, 2011. [Sections 5.2 and 5.3]

A. Garcia-Najera and J. A. Bullinaria. Optimizing delivery time in multi-objective vehicle routing problems with time windows. In Robert Schaefer, Carlos Cotta, Joanna Kolodziej, Günter Rudolph editors, *11th International Conference on Parallel Problem Solving from Nature*, volume 6239 part II of LNCS, pages 51–60. Springer, 2010. [Section 5.3]

Additionally, although it is not directly related to the main topic of this thesis, the following is a study at an early stage of this PhD research.

A. Garcia-Najera and J. A. Bullinaria. Extending ACO_R to solve multi-objective problems. In *2007 UK Workshop on Computational Intelligence*, 2007.

1.5 Thesis outline

This thesis is structured as follows. Chapter 2 provides the required background for the present study. It explains what combinatorial optimisation problems are and some methods frequently used for solving them, namely local search, Tabu Search, and Evolutionary Algorithms. Here are described multi-objective combinatorial optimisation problems, and some quality indicators that are used to evaluate optimiser performance. Three widely-known multi-objective Evolutionary Algorithms are presented, highlighting their key stages of processing. Lastly, a series of solution distance measures for combinatorial problems are introduced.

The Vehicle Routing Problem and two of its variants, specifically the Capacitated VRP (CVRP) and the VRP with Time Windows (VRPTW), are formulated in Chapter 3. Additional considerations and constraints are exposed for further variants of the problem. An overview of previous studies that are relevant to this thesis, that is studies that tackled the CVRP and VRPTW, is given, along with their reported results for commonly used benchmark sets. Finally, this chapter discusses multi-objective VRPs and surveys further studies and objectives that have been considered for optimisation.

The developed preliminary approaches to solving VRPs, along with their results and analysis, is presented in Chapter 4. One of the major contributions of this thesis is presented in this chapter, which is the measure designed to quantify the similarity between two solutions to the VRP, as well as the proposed population diversity measure. The algorithm described in this chapter forms the basis of the final proposed Multi-Objective Evolutionary Algorithm for solving the CVRP and VRPTW.

Chapter 5 explains the adjustments made to the algorithm previously introduced in order to effectively solve the CVRP and VRPTW. Here are presented the experimental studies and the analysis of results from the bi-objective and tri-objective optimisation of the concerned problems, and a discussion on the findings.

Finally, the evaluation of the proposed approach at its different stages of development, along with the main contributions of this thesis and potential directions for further research, are presented in Chapter 6.

Chapter 2

Combinatorial optimisation problems

This chapter provides the essential background knowledge to this thesis. Here is described, firstly, what combinatorial optimisation problems are and some common techniques for solving them, specifically local search, Tabu Search and Evolutionary Algorithms. Secondly, an introduction to multi-objective combinatorial optimisation problems is given, as well as to three multi-objective performance metrics, that is coverage, convergence and hypervolume. Then, the key difference between single-objective and multi-objective Evolutionary Algorithms is explained, an three well-known and successful multi-objective Evolutionary Algorithms, namely PAES [144], SPEA2 [261], and NSGA-II [67], are described. Finally, a number of solution distance measures for combinatorial problems are presented.

2.1 What are combinatorial optimisation problems?

An optimisation problem consists in finding the best (optimum) solution to a given instance of the problem. These problems can be grouped into two categories: In the first category one can find those problems with *continuous* variables, where we look for a vector of real numbers. In the second group we find those problems with discrete variables, which are called *combinatorial*, where one typically looks for an object within a finite set, or possibly countable infinite, which is generally a sub-set of the variables, a permutation, or a graph [186].

Formally, an instance of a combinatorial optimisation problem is a pair $(\mathcal{X}, f)$, where $\mathcal{X}$ is an N-dimensional domain, and f is a function that maps $f : \mathcal{X} \to \mathbb{R}$. Without loss of generality, we consider the problem of finding a solution $\boldsymbol{x}^* \in \mathcal{X}$ for which

$$f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}), \qquad \forall\, \boldsymbol{x} \in \mathcal{X} \ . \tag{2.1}$$

Such a solution $\boldsymbol{x}^*$ is called a *globally optimum solution* to the given instance, or, when no confusion arises, simply an *optimum solution* [186]. Furthermore, $f^* = f(\boldsymbol{x}^*)$ denotes the *optimum cost*, and $\mathcal{X}^* = \{\boldsymbol{x} \in \mathcal{X} \mid f(\boldsymbol{x}) = f^*\}$ denotes the *optimum solution set* [2].

Sometimes we face problems for which not all $\boldsymbol{x} \in \mathcal{X}$ are valid solutions, in the sense that they do not satisfy certain restrictions. Thus, all $\boldsymbol{x} \in \mathcal{X}' \subseteq \mathcal{X}$ satisfying such restrictions are the *feasible solutions*, and subset $\mathcal{X}'$ is the *feasible domain*. In this case, the problem in (2.1) must satisfy

$$g_i(\boldsymbol{x}) \leq 0 \ , \tag{2.2}$$

$$h_j(\boldsymbol{x}) = 0 \ , \tag{2.3}$$

where the $g_i(\boldsymbol{x})$ functions in (2.2) and the $h_j(\boldsymbol{x})$ functions in (2.3) are the inequality and equality constraints, respectively, which actually define the feasible domain $\mathcal{X}'$.

We will refer to function f as the *objective function*, to the domain $\mathcal{X}$ as the *solution space*, and to the feasible domain $\mathcal{X}'$ as the *feasible region* or *search space*.

For illustration purposes, Figure 2.1 presents one of the simplest combinatorial problems, though not easy, which is the *knapsack problem* [139]. Here we have a set

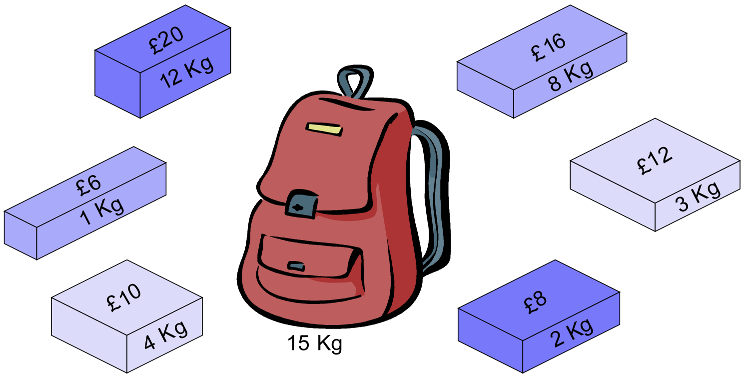


Figure 2.1: Knapsack problem: which items should be packed to maximise the total value while the overall weight is kept under or equal to the backpack limit?

of items, each with a weight and a value, and a backpack which capacity is limited in weight. The problem consists in determining which combination of items (the solution space) should be packed so that the total weight is less than or equal to the backpack limit (inequality constraint) and the total value is as large as possible (the objective function). The Vehicle Routing Problem is a also combinatorial problem, however, in addition to selecting items, we have to define a sequence of them, which perhaps makes the problem even more difficult to solve.

2.2 How to solve combinatorial optimisation problems

A naive approach to solving an instance of a combinatorial optimisation problem would be to simply enumerate all the possible solutions and choose the best according to the evaluation of their objective function. Nonetheless, it will be soon evident that this method is highly expensive in terms of time, due to the large amount of feasible solutions to an instance of practical size [200]. For example, let us consider a combinatorial problem for which all permutations are valid solutions, i.e. $N!$ valid solutions, N being the instance size. Furthermore, let us assume that the current fastest supercomputers[1,2] are available to list all the permutations. These computers perform in the order of 10^{15} operations per second, which means that they would spend around 40 minutes in listing all the permutations for an instance of size 20, 14 hours for an instances of size 21, 13 days for an instance of size 22, 300 days for an instance of size 23, 20 years for an instance of size 24, and, for an instance of size 25, half a millennium! Consequently, it is clear that this method is not an option to solve instances of reasonable sizes.

In the scenario described above, we could use other sort of methods to find *good* solutions to the problem in an efficient manner, although not guaranteeing the best. Such methods are called *heuristics* [172]. There are a number of well-known general purpose heuristics for solving a large variety of combinatorial problems, among which we can mention:

- Local search [1]
- Tabu Search [117]
- Simulated Annealing [142, 241]
- Particle Swarm Optimisation [140]
- Ant Colony Optimisation [76]
- Evolutionary Algorithms [58]

As will be seen later, local search, Tabu Search, and Evolutionary Algorithms, have been the most commonly used methods for solving VRPs. Local search is based on what is perhaps the oldest optimisation method: trial and error. The idea is so simple and natural that is surprising how successful local search has proven on a variety of difficult combinatorial optimisation problems [186]. Tabu Search has

[1]IBM RoadRunner: `http://www.lanl.gov/roadrunner/` .

[2]Cray Jaguar: `http://www.nccs.gov/computing-resources/jaguar/` .

Algorithm 2.1: LocalSearch($\boldsymbol{x}$)

Input: Initial solution $\boldsymbol{x}$
Output: Solution $\boldsymbol{x}^*$ result of the local search
while Improve($\boldsymbol{x}$) $\neq$ **null do**
  $\boldsymbol{x}^* \leftarrow$ Improve($\boldsymbol{x}$)
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^*$
end while
return $\boldsymbol{x}^*$

demonstrated to be an excellent heuristic for solving combinatorial optimisation problems and has achieved practical success in a range of applications [117]. On the other hand, Evolutionary Algorithms have a natural approach for dealing with multi-objective combinatorial optimisation problems, and have been successful in many practical situations as well [255].

2.2.1 Local search

The basic idea behind local search is, as the name suggests, to look for solutions around a given point in the search space. An algorithm of this kind starts with a candidate solution and then, iteratively, moves from this point to a *neighbour* solution where the objective which is being optimised has been improved.

Formally, given an optimisation problem $(\mathcal{X}, f)$, where $\mathcal{X}$ is the feasible domain of solutions and f is the objective function, a *neighbourhood* $\mathcal{N}$ is a function that maps $\mathcal{N} : \mathcal{X} \to 2^{\mathcal{X}}$, which defines, for each solution $\boldsymbol{x} \in \mathcal{X}$, a set $\mathcal{N}(\boldsymbol{x}) \subset \mathcal{X}$ of solutions that are in some sense close to $\boldsymbol{x}$ [2]. Local search explores this neighbourhood for improvement. A solution $\widehat{\boldsymbol{x}} \in \mathcal{X}$ is said to be *locally optimum* with respect to $\mathcal{N}$ if $f(\widehat{\boldsymbol{x}}) \leq f(\boldsymbol{x}),\ \forall\ \boldsymbol{x} \in \mathcal{N}(\widehat{\boldsymbol{x}})$. We denote the set of locally optimum solutions by $\widehat{\mathcal{X}}$. Note that local optimality depends on the neighbourhood function that is used [2]

A generic local search is shown in Algorithm 2.1. It takes a starting candidate solution $\boldsymbol{x}$ and uses function Improve($\boldsymbol{x}$), which returns a solution $\boldsymbol{x}^* \in \mathcal{N}(\boldsymbol{x})$, with $f(\boldsymbol{x}^*) < f(\boldsymbol{x})$, if such $\boldsymbol{x}^*$ exists, **null** otherwise, to search for a better solution in its neighbourhood $\mathcal{N}(\boldsymbol{x})$. The process is repeated over the new neighbourhood as long as a better solution is found [186].

The performance of local search is directly related to how neighbourhoods are defined and explored. Actually, the definition of the neighbourhood is the difference between the approaches that have been proposed. Some of the most commonly used local search heuristics for tackling VRPs, which have been taken from the proposed for solving the Travelling Salesman Problem [123], are *k-opt* [162], *Or-opt* [183], and λ-*interchange* [184].

2.2.1.1 k-opt heuristic

Lin [162] presented the k-opt heuristic, which is a generalisation of the 2-opt[3] of Croes [51] and 3-opt of Bock [24].

The most simple version of the k-opt heuristic, i.e. $k = 2$, works as follows. Let us consider the circuit $\langle v_1, \ldots, v_i, v_{i+1}, \ldots, v_j, v_{j+1}, \ldots, v_N \rangle$ of N vertices. Let two arcs be removed from the circuit, e.g. (v_i, v_{i+1}) and (v_j, v_{j+1}), thus producing the two disconnected paths $\langle\langle v_{j+1}, \ldots, v_N, v_1, \ldots, v_i \rangle\rangle$ and $\langle\langle v_{i+1}, \ldots, v_j \rangle\rangle$. Then, reconnect those paths in the other possible way, that is $\langle v_1, \ldots, v_i, v_j, \ldots, v_{i+1}, v_{j+1}, \ldots, v_N \rangle$. This operation is called *2-exchange*.

For $k = 3$, three arcs are removed, thus producing three disconnected paths. In general, k arcs are removed, originating k disconnected paths, which can be reconnected in different ways to produce another tour. This operation is called *k-exchange*. A tour is *k-optimum* if no k-exchange produces a tour of lower cost [35].

Although the problem of finding a k-optimum tour can be performed in a number of operations polynomial in N, this number is exponential in k and is bounded from below by N^k. Thus only very small values of k can be used in any heuristic [36, 163].

[3] According to Johnson and McGeoch [132], the basic move was suggested by Flood [88].

2.2.1.2 Or-opt heuristic

The Or-opt heuristic, introduced by Or [183], is closely related to the 3-opt heuristic. The basic idea is to relocate a sequence of vertices. This is achieved by replacing three arcs in the original tour by three new arcs without modifying the orientation of the route [26].

Let us illustrate this heuristic with the following example. Consider the circuit $\langle v_1, \ldots, v_i, v_{i+1}, \ldots, v_j, v_{j+1}, \ldots, v_k,\ v_{k+1}, \ldots, v_N \rangle$ and let us remove three arcs from it, e.g. (v_i, v_{i+1}), (v_j, v_{j+1}), and (v_k, v_{k+1}), thus producing the three disconnected paths $\langle v_{k+1}, \ldots, v_N, v_1, \ldots, v_i \rangle$, $\langle v_{i+1}, \ldots, v_j \rangle$, and $\langle v_{j+1}, \ldots, v_k \rangle$. Then, reconnect those paths in the following order $\langle v_1, \ldots, v_i, v_{j+1}, \ldots, v_k, v_{i+1}, \ldots, v_j, v_{k+1}, \ldots, v_N \rangle$. As we can see, the final circuit preserved the sequence order of the original circuit, hence this heuristic is useful when the precedence of the vertices is important.

2.2.1.3 λ-interchange heuristic

Osman [184] proposed the λ-interchange heuristic. This method selects two subsets r_i and r_j of vertices from circuits $\mathcal{R}_i$ and $\mathcal{R}_j$, respectively, satisfying $|r_i| \leq \lambda$ and $|r_j| \leq \lambda$. The operation consists in swapping the vertices in r_i with those from r_j as long as this is feasible.

It can be the case that either r_i or r_j are empty, so this family of operations includes the simply shifting of vertices from one circuit to another. As the number of combinations of choices of r_i and r_j is usually large, this procedure is implemented with $\lambda = 1, 2$, and in the most efficient version of this algorithm the search stops as soon as a solution improvement is discovered [110].

Algorithm 2.2: TabuSearch($\boldsymbol{x}$)

Input: Initial solution $\boldsymbol{x}$
Output: Solution $\boldsymbol{x}^*$ result of the Tabu Search

1: $\boldsymbol{x}^* \leftarrow \boldsymbol{x}$
2: $k \leftarrow 0$
3: $T \leftarrow \emptyset$
4: **repeat**
5:     $k \leftarrow k + 1$
6:     $s_k \leftarrow$ Select($S(\boldsymbol{x}) \setminus T$)   */* s_k is the best move in $S(\boldsymbol{x}) \setminus T$ */*
7:     $\boldsymbol{x} \leftarrow s_k(\boldsymbol{x})$ */* move s_k is applied to $\boldsymbol{x}$ */*
8:     **if** $f(\boldsymbol{x}) < f(\boldsymbol{x}^*)$ **then**
9:         $\boldsymbol{x}^* \leftarrow \boldsymbol{x}$
10:     **end if**
11:     Update(T)
12: **until** $S(\boldsymbol{x}) \setminus T = \emptyset$
13: **return** $\boldsymbol{x}^*$

2.2.2 Tabu Search

Tabu Search (TS) is a guide for certain methods known as *hill climbing* [174], which progress in one direction from their starting point to a local optimum, to continue exploration without becoming confused by an absence of improving moves and without falling back into a local optimum where it was previously stuck. This is accomplished by two of its key elements [115]: that of restricting the search by classifying certain moves as forbidden, i.e. *tabu*, and that of releasing the search by a short term memory function that provides a kind of *strategic forgetting*, which removes records from the tabu list after some iterations.

To have an idea of TS, Algorithm 2.2 presents the general process. Here T represents a list of forbidden moves and $S(\boldsymbol{x})$ is the set of feasible moves that lead from $\boldsymbol{x}$ to a neighbouring solution. Let us emphasise that both T and $S(\boldsymbol{x})$ contain *moves* and not actual solutions.

According to Glover [115, 116], three aspects deserve attention: (i) list T restricts the search, hence the solutions generated depend on the composition of T and the way it is updated (line 11), (ii) local optimality is never referred to, except when a local optimum improves the best solutions previously found, and (*iii*) instead of an improvement move, a *best* move is chosen at each step (line 6).

2.2.3 Evolutionary Algorithms

The first ideas of evolutionary computation date back to the 1950's [60]. In that decade, a number of scientific articles regarding *computer evolutionary processes* were published. For example, the studies of Fraser [94], Friedberg [95], and Friedberg et al. [96] in machine learning refer to the use an evolutionary algorithm for *automatic programming*. In the same time frame, Bremermann [29] presented the first attempts to apply simulated evolution to numerical optimisation problems, and Box [25] developed his *evolutionary operation* for the design and analysis of industrial experiments, to which Satterthwaite [209] introduced randomness and was the basis for the simplex design method of Spendley et al. [222].

By the mid-1960's the bases for what we today identify as the three main forms of Evolutionary Algorithms (EA) were clearly established [60]: Fogel et al. [89] called their method Evolutionary Programming, Holland [126] introduced Genetic Algorithms, and Schwefel [214], Rechenberg [198, 199], and Bienert [23] proposed Evolution Strategies.

EAs are founded on Darwin's theory of evolution, where the *fittest* individuals survive and produce offspring to populate the next generation. In this context, a *population* of *individuals* (also called *chromosomes*) is maintained, with each individual being a problem solution, and *fitness* being an appropriate measure of how good a solution is. EAs have a number of procedures and parameters that must be specified in order to define their operation, and crucial to this is how the offspring are created from the parents. The general process of an EA is presented in Algorithm 2.3 and the specific operations are described below.

Algorithm 2.3: EVOLUTIONARYALGORITHM(P)

Input: Population P of solutions to be evolved
Output: Solutions P^* result of the evolutionary process

$P_0 \leftarrow P$
Evaluate P_0 and assign fitness
$i \leftarrow 0$
repeat
    Select parents from P_i and recombine to produce offspring Q_i
    Mutate Q_i and evaluate
    Combine P_i and Q_i, assign fitness, and select P_{i+1}
    $i \leftarrow i + 1$
until Stop criteria is met
return $P^* \leftarrow P_i$

2.2.3.1 Solution representation

The efficiency and complexity of an EA largely depends upon how solutions to a given problem are represented and how suitable the representation is in the context of the underlying search operators [61]. As the *encoding* of a solution varies from problem to problem, a solution to a particular problem can be represented in a number of ways, some of which lead to a more efficient search.

For the particular case of combinatorial problems, a solution is generally represented as a binary string [10], i.e. a string of 0's and 1's, when only a combination of the variables involved in the problem is necessary, or as a permutation [249], if, in addition to the combination, the sequence of the variables is important.

2.2.3.2 Initial population

It is standard practice for an EA to begin with an initial population chosen randomly with the aim of covering the entire search space. Thus the algorithm starts with a set $P = \{s_1, \ldots, s_{popSize}\}$ of *popSize* randomly generated solutions s_i.

2.2.3.3 Fitness assignment

At each generation of evolution, the objective function is evaluated for every solution s_i in the population P, and for each individual s_i a *fitness* function value $\varphi(s_i)$ is computed, which drives the natural selection process. This fitness value is assigned according to the relative quality of the solutions in the population, which means that the best solutions in the population are assigned a higher fitness.

2.2.3.4 Parent selection

The evolutionary process requires some stochastic function for selecting *parent* individuals from the population, according to their fitness, to undergo *recombination* to create an *offspring*. Fittest individuals should be more likely to be selected, however, low-fitness individuals might also be given a small chance with the aim of not allowing the algorithm to be too greedy [81].

One can find several techniques for parent selection, from which two are the most commonly utilised. The first of them, which belongs to the class of *proportional selection methods* [122], is known as the *Roulette Wheel Selection* [57]. It takes its name from the roulette gambling game, where all individuals $s_i \in P$ are assigned a portion of the wheel accordingly to their relative fitness function value $\varphi(s_i)$. Then, a ball is dropped and selects one element: the ball is more likely to stop on elements assigned a bigger portion of the wheel.

In this case, the probability $p(s_i)$ that individual s_i is selected, is computed as

$$p(s_i) = \frac{\varphi(s_i)}{\Phi} \ , \tag{2.4}$$

where

$$\Phi = \sum_{s_i \,\in\, P} \varphi(s_i) \ . \tag{2.5}$$

Algorithm 2.4: ROULETTEWHEELSELECTION(P)

Input: Population $P = \{s_1, \ldots, s_{popSize}\}$ from which one solution is going to be selected
Output: Selected solution s

$\Phi \leftarrow 0.0$
for all $s_i \in P$ **do**
  $\Phi \leftarrow \Phi + \varphi(s_i)$
end for
$\rho \leftarrow$ RANDOMNUMBER($[0, 1]$)
$sum \leftarrow 0.0$
while $sum < \rho$ **do**
  $s \leftarrow$ SELECTRANDOMINDIVIDUAL(P)
  $sum \leftarrow sum + \dfrac{\varphi(s)}{\Phi}$
end while
return s

The stochastic selection is done according to Algorithm 2.4. First, Φ is calculated and then a random number in the range $[0, 1]$ is stored in ρ. Afterwards, an individual $s \in P$ is randomly selected with function SELECTRANDOMINDIVIDUAL(P), and its proportional fitness is added to *sum*. This random selection continues until *sum* is equal or greater than ρ and the last selected individual is returned.

The second method is the one called *Tournament Selection* [22]. It randomly chooses $Tsize$ (the *tournament size*) individuals from the population and selects the fittest individual from this group to be a parent. Tournaments are often held between pairs of individuals, i.e. $Tsize = 2$, which is called *binary tournament*, although larger tournaments can be used [119].

2.2.3.5 Recombination

Recombination, also called *mating* or *crossover*, is the process of generating one or more offspring from the selected parents, preferably in a manner that maintains and combines the desirable features from both. This operation is carried out with probability γ, otherwise the fittest individual is simply copied into the offspring population.

Similarly to parent selection, there exists a number of recombination methods, however, in contrast, they all depend on the solution encoding in use.

2.2.3.6 Mutation

Once an offspring has been generated, a further stochastic change or *mutation* is applied with probability μ. This operator, together with recombination, have the aim of exploring and exploiting the search space.

There are also many mutation operators, however, likewise the recombination techniques, they all depend on the representation of the solution.

2.2.3.7 Survival selection

The final stage of the evolutionary cycle is the selection of individuals to form the next generation. There are several obvious possibilities: the offspring population, a random selection from the combined parent and offspring populations, or the best individuals from the combined population. In the first two cases, good-quality individuals are likely to be lost, so it is common to consider some degree of *elitism*, i.e. a percentage of the next generation is filled with the best current individuals.

2.2.3.8 Repetition

The whole process of parent selection, offspring generation, i.e. recombination and mutation, and survival selection is repeated for a fixed number *numGen* of generations, or until some stop criteria is met.

2.2.3.9 Parameter setting

Several parameters have been introduced above:

- Solution encoding - *How will solutions be represented?*
- Initial population - *How is it going to be built?*
- Population size - *How many solutions will be considered?*
- Parent selection - *What technique is going to be used?*
- Recombination - *Which method and what is the probability γ of recombination?*
- Mutation - *Which method and what is the probability μ of mutation?*
- Survival selection - *Which individuals will be taken to the next generation?*
- Repetition - *How many generations numGen the EA will run or what will the stop criteria be?*

The values of the parameters involved in an EA, and in any algorithm in general, determine whether the algorithm will find near-optimum solutions, and whether it will find such solutions efficiently [173]. For example, according to De Jong [59], a solutions space with many local optima may require a population size of hundreds to thousands in order to have reasonable chance of finding globally optimum solutions. On the other hand, if reproductive variation is too strong, i.e. high recombination and mutation probabilities, the result is undirected random search.

Two forms of setting parameter values can be distinguished: *tuning* and *control*. Parameter tuning is done by experimenting with different values and selecting the ones that give the best results on the test problems at hand. However, the number of possible parameters and their different values means that this is a very time-consuming activity [82]. Parameter control, on the other hand, forms an alternative, as it amounts to starting a run with initial parameter values that are changed during the run [82]. Parameter setting, in any case, is a field of research by itself [164].

In the experimental studies carried out in this investigation, which will be presented in Chapters 4 and 5, first experiments were performed using a range of parameter values for parameter tuning purposes. Fair parameter optimisation requires separate validation data sets, which were not readily available, hence a full parametric study was not carried out, and thus no comparison nor analysis is presented for the different tested settings. That activity is suggested as part of the future research. Although an exhaustive search was not carried out, parameter values that worked well over all the test cases were identified and used for the final experimentation. Wherever possible, standard parameter values from the literature were used.

2.3 Multi-objective combinatorial optimisation problems

The definition of combinatorial optimisation problems described above considers the optimisation of one objective function $f(\boldsymbol{x})$, but how can we handle a combinatorial problem with not only one, but with F objective functions? For example, consider the knapsack problem previously introduced. How can we solve the problem if, in addition to its value, each item has an associated *risk* of being carried, which is inversely proportional to the value, and the objective is to, additionally to maximise the total value, minimise the total risk?

In this case, $f(\boldsymbol{x})$ becomes the vector $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_F(\boldsymbol{x}))$ of F objective functions, and will map $\boldsymbol{f} : \mathcal{X} \rightarrow \mathbb{R}^F$. Thus, we can re-define (2.1) as the minimisation problem

$$\text{minimise } \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_F(\boldsymbol{x})) \ , \tag{2.6}$$

subject to constraints (2.2) and (2.3). Now, $\boldsymbol{f}$ will be referred to as the *objective space.*

If the objective functions f_k in (2.6) are not in conflict, that is, if the optimisation of one of these objectives will lead to the optimisation of all of them, the optimum

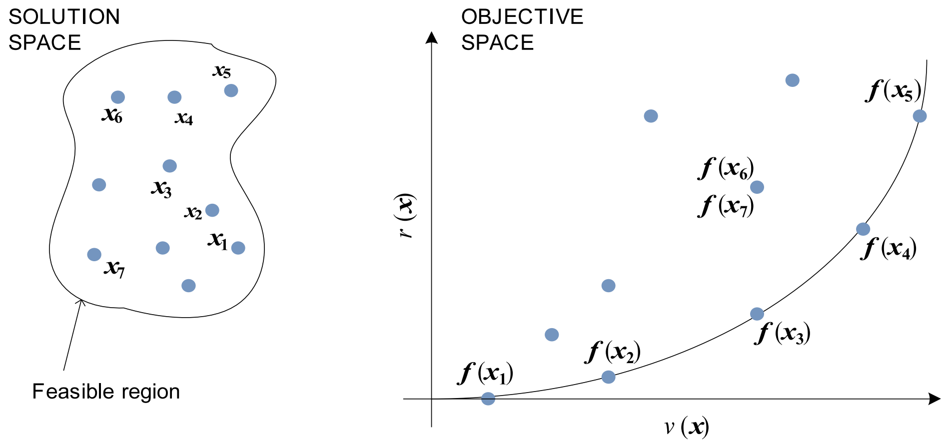


Figure 2.2: Solutions $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5}$ to an example instance of the knapsack problem with two objective functions, $v(\boldsymbol{x})$ and $r(\boldsymbol{x})$, and their corresponding trade-off $\boldsymbol{f}(\boldsymbol{x_1}), \ldots, \boldsymbol{f}(\boldsymbol{x_5})$ between the objectives.

result of the process will be one solution in the search space, or probably more but all of them mapping to one point in the objective space. The most interesting case is when the problem presents conflicting objectives, i.e. the optimisation of one objective will cause the deterioration of the others, where we will have a set of solutions as the outcome of the optimisation process, as shows the example in Figure 2.2. In this example we consider the proposed knapsack problem above, with $\boldsymbol{f}(\boldsymbol{x}) = \big(v(\boldsymbol{x}), r(\boldsymbol{x})\big)$, where $v(\boldsymbol{x})$ is the total value to be maximised and $r(\boldsymbol{x})$ is the total risk to be minimised, and both are in conflict. The optimisation process results in five solutions, $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5}$, which provide the trade-off $\boldsymbol{f}(\boldsymbol{x_1}), \ldots, \boldsymbol{f}(\boldsymbol{x_5})$ between both objectives.

2.3.1 Terminology

The following are some terms used in the multi-objective optimisation context and are graphically represented in Figure 2.3.

Let $\mathcal{X}$ be the N-dimensional domain of solutions to a multi-objective optimisation problem. We say that a solution $\boldsymbol{x} \in \mathcal{X}$ *weakly dominates* (or *covers*) the solution $\boldsymbol{y} \in \mathcal{X}$, written as $\boldsymbol{x} \preceq \boldsymbol{y}$, if $\boldsymbol{x}$ is at least as good as $\boldsymbol{y}$ [44]. In the example shown in

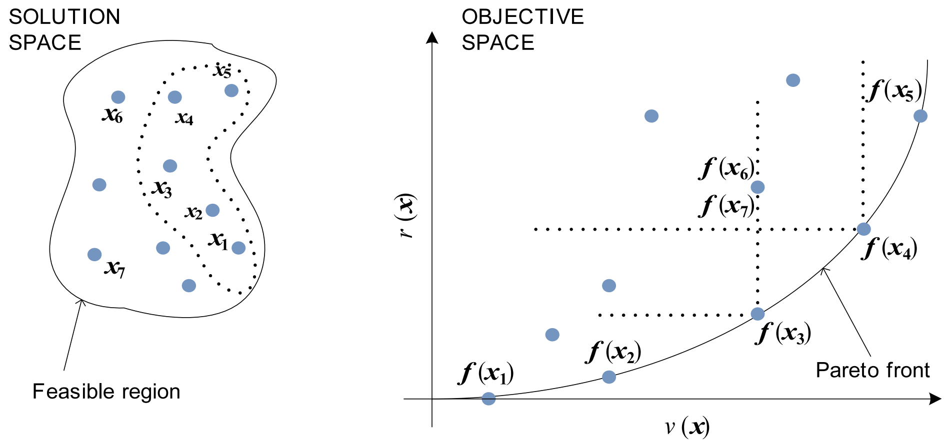


Figure 2.3: Multi-objective concepts: $\boldsymbol{x_3}$, $\boldsymbol{x_4} \preceq \boldsymbol{x_6}$, $\boldsymbol{x_7}$; moreover, $\boldsymbol{x_3}$, $\boldsymbol{x_4} \prec \boldsymbol{x_6}$, $\boldsymbol{x_7}$. $\boldsymbol{x_6} \preceq \boldsymbol{x_7}$ and $\boldsymbol{x_7} \preceq \boldsymbol{x_6}$. $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5}$ are Pareto optimum, since they are non-dominated, thus they belong to the Pareto set. Consequently, $\boldsymbol{f}(\boldsymbol{x_1}), \ldots, \boldsymbol{f}(\boldsymbol{x_5})$ are in the Pareto front.

Figure 2.3, we can see that solutions $\boldsymbol{x_3}$ and $\boldsymbol{x_4}$ weakly dominate solutions $\boldsymbol{x_6}$ and $\boldsymbol{x_7}$, since $v(\boldsymbol{x_3}), v(\boldsymbol{x_4}) \geq v(\boldsymbol{x_6}), v(\boldsymbol{x_7})$, and $r(\boldsymbol{x_3}), r(\boldsymbol{x_4}) \leq r(\boldsymbol{x_6}), r(\boldsymbol{x_7})$. Solutions $\boldsymbol{x_6}$ and $\boldsymbol{x_7}$ weakly dominate each other, because $\boldsymbol{f}(\boldsymbol{x_6}) = \boldsymbol{f}(\boldsymbol{x_7})$.

Solution $\boldsymbol{x}$ *dominates* solution $\boldsymbol{y}$, written as $\boldsymbol{x} \prec \boldsymbol{y}$, if and only if $\boldsymbol{x} \preceq \boldsymbol{y}$ and $\boldsymbol{x}$ is strictly better than $\boldsymbol{y}$ in at least one objective [44]. Solutions $\boldsymbol{x_3}$ and $\boldsymbol{x_4}$ in Figure 2.3 dominate solutions $\boldsymbol{x_6}$ and $\boldsymbol{x_7}$, since $\boldsymbol{x_3}, \boldsymbol{x_4} \preceq \boldsymbol{x_6}, \boldsymbol{x_7}$, and $r(\boldsymbol{x_3}), r(\boldsymbol{x_4}) < r(\boldsymbol{x_6}), r(\boldsymbol{x_7})$.

Consequently, one says that a solution $\boldsymbol{x} \in \mathcal{S} \subseteq \mathcal{X}$ is *non-dominated* with respect to $\mathcal{S}$ if there is no solution $\boldsymbol{y} \in \mathcal{S}$ such that $\boldsymbol{y} \prec \boldsymbol{x}$, like solutions $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5}$ in the example, since there is no solution $\boldsymbol{y} \in \mathcal{S} = \mathcal{X}$ such that $\boldsymbol{y}$ is strictly better than $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5}$ in any of the objectives.

A solution $\boldsymbol{x} \in \mathcal{X}$ is said to be *Pareto optimum* if it is non-dominated with respect to $\mathcal{X}$, and the *Pareto optimum set* is defined as $\mathcal{P}_s = \{\boldsymbol{x} \in \mathcal{X} \mid \boldsymbol{x} \text{ is Pareto optimum}\}$. Solutions $\boldsymbol{x_1}, \ldots \boldsymbol{x_5}$ in Figure 2.3 are Pareto optimum and comprise the Pareto optimum set because they are non-dominated.

Finally, the *Pareto front* is defined as $\mathcal{P}_f = \{\boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^F \mid \boldsymbol{x} \in \mathcal{P}_s\}$. In the example, the vectors in the objective space $\boldsymbol{f}(\boldsymbol{x_1}), \ldots, \boldsymbol{f}(\boldsymbol{x_5})$ comprise the Pareto front, since solutions $\boldsymbol{x_1}, \ldots, \boldsymbol{x_5} \in \mathcal{P}_s$.

2.3.2 Performance metrics

The comparison of multi-objective optimiser performance is not an easy task. In contrast with single-objective problems, where one can straightforwardly compare the best results, or the average of them, from the various methods studied, multi-objective problems have whole sets of solutions to compare with at least two aims: to minimise the distance from the generated solutions, called the *Pareto approximation*, *approximation set*, *non-dominated set*, or *non-dominated solutions*, to the true Pareto front, and to maximise the diversity of them, i.e. the coverage of the Pareto front. For this reason, the definition and use of appropriate performance metrics or quality indicators is crucial. Fortunately, this subject of research has been and continues to be widely studied. Currently, there are many proposed metrics [84, 53, 258, 242, 257, 240, 260, 65] that can be classified into unary, which assign each non-dominated set a number that reflects a certain quality aspect, and binary, which assign a number to a pair of Pareto approximations. The studies of Zitzler et al. [263], Knowles et al. [143], and Zitzler et al. [265], present an excellent review of many quality indicators and provide an in-depth analysis of them.

Formally, let Ω be the set of *all* approximation sets to a given problem. A quality indicator $\mathcal{I}$ is a function $\mathcal{I} : \Omega \to \mathbb{R}$, which assigns each vector of approximation sets a real value. Moreover, we say that $\mathcal{I}$ is *Pareto compliant* if $\forall\, \mathcal{A}, \mathcal{B} \in \Omega : \mathcal{A} \preceq \mathcal{B} \implies \mathcal{I}(\mathcal{A}) \geq \mathcal{I}(\mathcal{B})$ [42]. That is, whenever an approximation set $\mathcal{A}$ is preferable to $\mathcal{B}$ with respect to weak Pareto dominance, the indicator value for $\mathcal{A}$ should be at least as good as that for $\mathcal{B}$. On the other hand, we say that an indicator is *Pareto* *non-compliant* if, for any approximation sets $\mathcal{A}, \mathcal{B} \in \Omega$, it can yield a preference of $\mathcal{A}$ over $\mathcal{B}$, when $\mathcal{B}$ is preferable to $\mathcal{A}$ with respect to weak Pareto dominance [42].

Zitzler et al. [263] showed that there exists no unary quality measure that is able to indicate whether an approximation $\mathcal{A}$ is better than an approximation $\mathcal{B}$, even if a finite combination of unary measures are considered. Most quality measures that have been proposed to indicate that $\mathcal{A}$ is better than $\mathcal{B}$ at best allow to infer that $\mathcal{A}$ is not worse than $\mathcal{B}$, i.e. $\mathcal{A}$ is better than or incomparable to $\mathcal{B}$. Binary quality measures overcome the limitations of unary measures and, if properly designed, are capable of indicating whether $\mathcal{A}$ is better than $\mathcal{B}$.

The multi-objective analyses performed in this research, which are going to be presented in Chapters 4 and 5, have considered the use of three quality indicators. The first, which is binary and was proposed by Zitzler et al. [260], is called *coverage* and indicates, to some extent, how diverse the solutions from two approximation sets are. The second, *convergence* from Deb and Jain [65], is unary and measures the distance from an approximation set to a reference set. Finally, Zitzler and Thiele [258] proposed the unary *hypervolume*, which quantifies the size of the delimited objective space. From these, coverage and hypervolume are Pareto compliant [143, 42]. The following provides formal definitions of these quality indicators.

2.3.2.1 Coverage

This performance metric measures the extent to which one approximation set $\mathcal{B}$ is weakly dominated by another approximation set $\mathcal{A}$. $\mathsf{M_C}(\mathcal{A}, \mathcal{B})$ compares the number of solutions in $\mathcal{B}$ that are weakly dominated by the solutions in $\mathcal{A}$ to the cardinality of $\mathcal{B}$. Formally, this ratio maps the ordered pair $(\mathcal{A}, \mathcal{B})$ to the interval [0,1] as the general coverage metric [260]:

$$\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = \frac{\big|\{\mathbf{y} \in \mathcal{B} : \exists\, \mathbf{x} \in \mathcal{A}, \mathbf{x} \preceq \mathbf{y}\}\big|}{|\mathcal{B}|} \,. \tag{2.7}$$

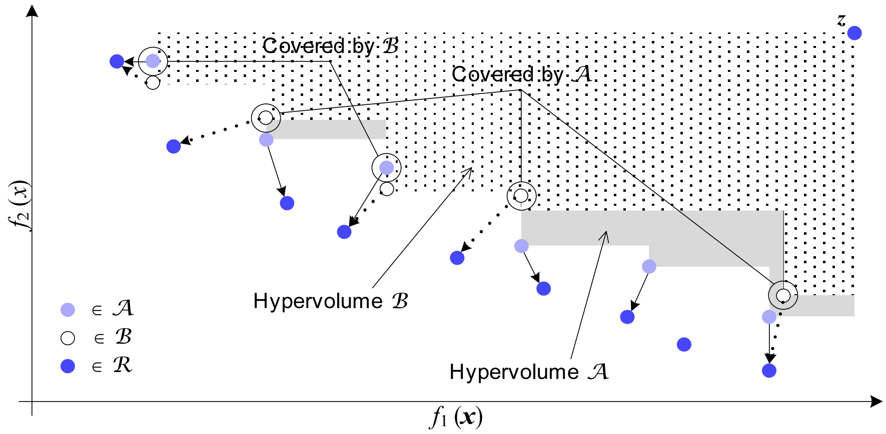


Figure 2.4: Graphical representation of the coverage ($\mathsf{M_C}$), convergence ($\mathsf{M_D}$), and hypervolume ($\mathsf{M_H}$) multi-objective performance metrics: Solutions in set $\mathcal{A}$ have a wider coverage of the solutions in set $\mathcal{B}$ than the opposite case, are closer to the solutions in reference set $\mathcal{R}$, and define a larger hypervolume regarding point $\boldsymbol{z}$. Thus, the algorithm providing the set $\mathcal{A}$ of solutions is better than the one providing set $\mathcal{B}$.

The value $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 1$ means that all solutions in $\mathcal{B}$ are covered by the solutions in $\mathcal{A}$, while $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 0$ indicates the opposite situation, in which none of the solutions in $\mathcal{B}$ are covered by those in $\mathcal{A}$. Note that both $\mathsf{M_C}(\mathcal{A}, \mathcal{B})$ and $\mathsf{M_C}(\mathcal{B}, \mathcal{A})$ have to be considered, since $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 1 - \mathsf{M_C}(\mathcal{B}, \mathcal{A})$ does not necessarily hold.

The idea here is that the method with the best performance is the one which provides solutions with the largest coverage of the solutions from the other method. Figure 2.4 presents an example with points in two sets, the 6 filled circles representing set $\mathcal{A}$, and the 5 open circles representing set $\mathcal{B}$. Three points in set $\mathcal{B}$ are covered by set $\mathcal{A}$, and two points in $\mathcal{A}$ are covered by set $\mathcal{B}$, so $\mathsf{M_C}(\mathcal{A}, \mathcal{B}) = 3/5$, and $\mathsf{M_C}(\mathcal{B}, \mathcal{A}) = 2/6$. Thus the method providing solutions $\mathcal{A}$ is deemed better than that providing solutions $\mathcal{B}$.

2.3.2.2 Convergence

The convergence metric $\mathsf{M_D}(\mathcal{A}, \mathcal{R})$ measures the distance from the approximation set $\mathcal{A}$ to the reference set $\mathcal{R}$ [65]. To define this metric, we need first to calculate the smallest normalised Euclidean distance $d(\boldsymbol{x_i}), \forall\, \boldsymbol{x_i} \in \mathcal{A}$, to $\mathcal{R}$ as

$$d(\boldsymbol{x_i}) = \min_{\boldsymbol{y_j} \,\in\, \mathcal{R}} \sqrt{\sum_{k=1}^{F} \left(\frac{f_k(\boldsymbol{x_i}) - f_k(\boldsymbol{y_j})}{f_k^{\max} - f_k^{\min}} \right)^2} \,, \tag{2.8}$$

where $f_k^{\max}$ and $f_k^{\min}$ are the maximum and minimum function values of the k-th objective function in $\mathcal{R}$. Then the convergence $\mathsf{M}_{\mathsf{D}}(\mathcal{A}, \mathcal{R})$ is defined as

$$\mathsf{M}_{\mathsf{D}}(\mathcal{A}, \mathcal{R}) = \frac{1}{|\mathcal{A}|} \sum_{\boldsymbol{x_i} \in \mathcal{A}} d(\boldsymbol{x_i}) \,, \tag{2.9}$$

i.e. the average normalised distance for all points in $\mathcal{A}$. In the example shown in Figure 2.4, we can see that solutions in set $\mathcal{A}$ are closer to $\mathcal{R}$ than solutions in $\mathcal{B}$. Hence, the method which found $\mathcal{A}$ outperforms that which found $\mathcal{B}$.

2.3.2.3 Hypervolume

The hypervolume metric $\mathsf{M}_{\mathsf{H}}(\mathcal{A}, \boldsymbol{z})$ concerns the size of the objective space defined by the approximation set $\mathcal{A}$ of solutions, which is limited by setting a suitable reference point $\boldsymbol{z}$. The example in Figure 2.4 shows that the six filled circles representing set $\mathcal{A}$ cover the shaded region limited by the reference point $\boldsymbol{z} = (z_1, z_2)$, while the five open circles, which represent set $\mathcal{B}$, cover the dotted region.

For maximisation problems, it is common to take $\boldsymbol{z}$ to be the origin, while for minimisation problems, $\boldsymbol{z}$ is set to exceed the maximal values for each objective. Either way, when using this metric to compare the performance of two or more algorithms, the one providing solutions with the largest delimited hypervolume is regarded to be the best.

Formally, for a two-dimensional objective space $\boldsymbol{f}(\boldsymbol{x}) = \big(f_1(\boldsymbol{x}), f_2(\boldsymbol{x})\big)$, each solution $\boldsymbol{x_i} \in \mathcal{A}$ delimits a rectangle defined by its coordinates $\big(f_1(\boldsymbol{x_i}), f_2(\boldsymbol{x_i})\big)$ and the reference point $\boldsymbol{z} = (z_1, z_2)$, and the size of the union of all such rectangles delimited by the solutions is used as the measure. This concept can be extended to any number of dimensions F to give the general hypervolume metric [258]:

$$\mathsf{M_H}(\mathcal{A}, \boldsymbol{z}) = \lambda\left(\bigcup_{\boldsymbol{x_i}\,\in\,\mathcal{A}} \{[f_1(\boldsymbol{x_i}), z_1] \times \cdots \times [f_F(\boldsymbol{x_i}), z_F]\}\right)\ , \tag{2.10}$$

where $\lambda(\cdot)$ is the standard Lebesgue measure [93].

2.4 Multi-objective Evolutionary Algorithms

As mentioned above, there are generally two aims for multi-objective problems: to minimise the distance from the outcome set of solutions to the Pareto front, and to maximise the diversity of them. In the context of multi-objective Evolutionary Algorithms (EA), the first goal is mainly related to the task of assigning a fitness value to the solutions, while the second concerns how to handle the selection, because it is desirable to avoid identical solutions in the resulting set [264]. These two aspects make the principal difference between single-objective and multi-objective EAs.

The next sections describe generic ideas of how fitness may be assigned in multi-objective scenarios, and how selection can be implemented. Additionally, three widely-known multi-objective EAs are introduced.

2.4.1 Fitness assignment

When dealing with a multi-objective problem, fitness assignment can not be done straightforwardly, due to there being not only one objective function, but at least two of them which have to be taken into account. In general, one can distinguish the fitness assignment approaches below [264].

2.4.1.1 Aggregation

The most intuitive approach to assign fitness in the presence of multiple objective functions is to combine them into a single function. An example of this approach is a sum of weights of the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{F} w_i \; f_i(\boldsymbol{x}) \; , \tag{2.11}$$

where coefficients $w_i \geq 0$ are weighting values representing the relative importance of the F objective functions. It is usually assumed that $\sum_{i=1}^{F} w_i = 1$, however the coefficients are varied during the optimisation process in order to find a set of non-dominated solutions. Although this technique does not require any changes to the basic mechanism of an EA, the major drawbacks of this approach are the uncertainty of the weighting coefficients [39], and that for some kind of problems it can not generate proper members of the Pareto optimum set [55].

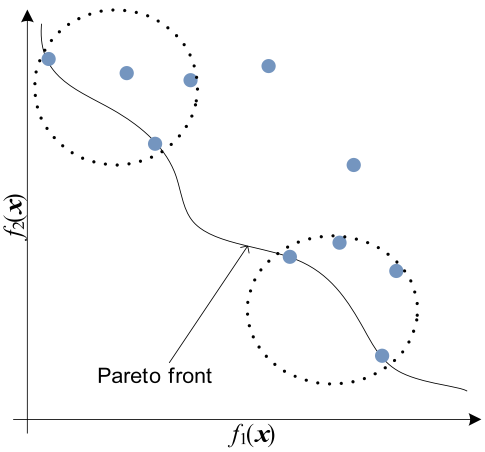
2.4.1.2 Criterion

This approach switches between the objectives during the selection phase. Each time an individual is chosen, potentially a different objective can be used to make this decision [212, 152]. In Figure 2.5(a) we see an example showing that solutions which objectives are contained in the upper-left ellipse have the minimum values for function f_1, while the those in the lower-right ellipse have the minimum values for function f_2. In this case, when one individual has to be selected, it will come from the sets specified by any of the two ellipses, according to the proportion or probability assigned to each objective function.

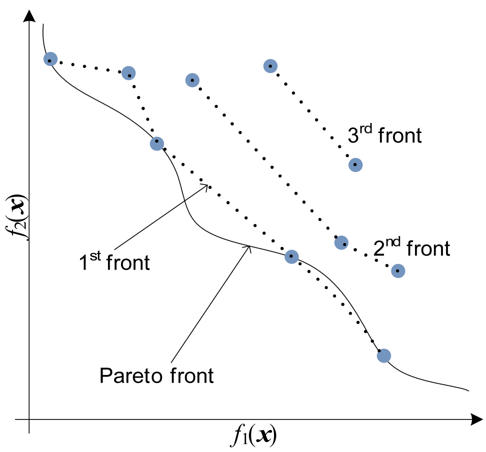
2.4.1.3 Pareto dominance

There are different methods to assign fitness under this approach [264], from which three are described below.

Dominance rank. This method considers the number of solutions in the population P by which an individual is dominated [90]. To find the rank of solution $s_i \in P$, we



(a) Criterion



(b) Dominance depth

Figure 2.5: Criterion and dominance depth methods for fitness assignment in multi-objective Evolutionary Algorithms.

need to compare it with every other $s_j \in P$ and count how many solutions dominate s_i. That is, the rank $r(s_i)$ of solution $s_i \in P$ is

$$r(s_i) = \left| \{ s_j \in P \mid s_j \prec s_i \} \right| . \tag{2.12}$$

Dominance count. This technique takes into consideration the number of solutions dominated by a certain individual [262]. In contrast with dominance rank, this method counts the number of individuals $s_j \in P$ that are dominated by s_i. Specifically, the dominance count $c(s_i)$ of solution $s_i \in P$ is

$$c(s_i) = \left| \{ s_j \in P \mid s_i \prec s_j \} \right| . \tag{2.13}$$

Dominance depth. This approach groups the population into non-dominated *fronts* and their depth indicates the fitness of the individuals belonging to them [223, 67]. This method is represented in Figure 2.5(b). A naive approach to identify the front to which each solution $s_i \in P$ belongs to, is to compare each solution s_i with every other $s_j \in P$ to know if they are non-dominated. However, this operation will only result in the first non-dominated front. If this procedure is repeated, the second non-dominated front will be found. That is, this process has to be executed as many times as needed in order to complete the assignment of solutions to fronts. The worst case is when there are *popSize* fronts and there exists only one solution in each front. In this case, this method requires an overall $O(FpopSize^3)$ comparisons, where F is the number of objective functions.

2.4.2 Diversity preservation

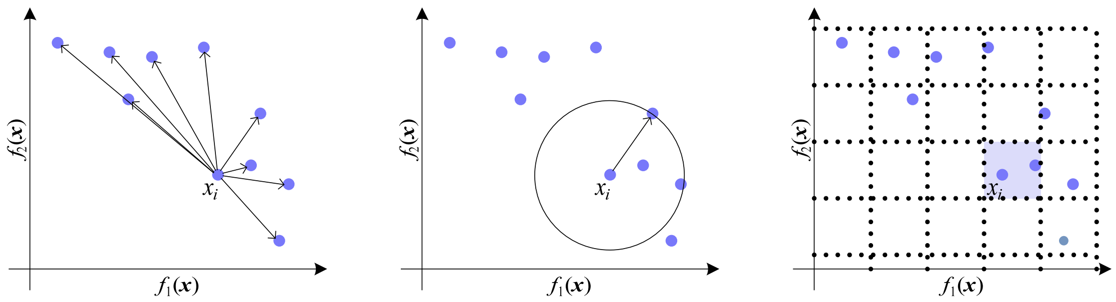
Diversity in the Pareto approximation is important because it is desirable that the solutions contained in this set are different. Density information reflects a good estimation of population diversity and could be used to increase it. This means that the probability of a solution being selected decreases as the density of solutions in its neighbourhood increases. The methods used in multi-objective EAs can be classified according to the categories of the techniques used in statistical density estimation [217, 264].

2.4.2.1 Kernel methods

In this kind of method, the distance in the objective space between each solution and all other in the population is calculated, as shown in Figure 2.6(a) for vector x_i. Then, a *Kernel* function is applied over those values. The density estimate for a solution will be the sum of all evaluations of the Kernel function [67].

2.4.2.2 Nearest neighbour

Methods in this category take into account the distance in the objective space between a given point and its k-th nearest neighbour to estimate density in its neighbourhood [262]. In the example shown in Figure 2.6(b), the x_i's third nearest neighbour is considered.



(a) Kernel (b) Nearest neighbour (c) Histogram

Figure 2.6: Kernel, nearest neighbour, and histogram methods for density estimation which are used to preserve diversity in multi-objective Evolutionary Algorithms.

2.4.2.3 Histogram

Techniques in this category define, as neighbourhoods, grids in the F-dimensional space as shown in Figure 2.6(c). The number of individuals which objectives are in the same grid as those of a given solution is then its density estimate [145, 41]. In the example, we see that x_i is sharing the neighbourhood with another vector.

2.4.3 Pareto Archived Evolution Strategy

One of the most popular multi-objective EAs is the proposed by Knowles and Corne [144], called Pareto Archived Evolution Strategy (PAES), which was designed with two objectives in mind: to use only a mutation operator to move from a current solution to a nearby neighbour, and to treat all non-dominated solutions as having equal value in order to be a true Pareto optimiser.

2.4.3.1 Fitness assignment

PAES is shown in Algorithm 2.5. It comprises three parts [144]: the candidate solution generator, function $\textsc{Mutate}(s)$ in line 3, the candidate solution acceptance

Algorithm 2.5: PAES(s)

Input: Solution s to be evolved
Output: Archive $A = \{s_1^*, \ldots, s_{popSize}^*\}$ result of PAES
1: $A \leftarrow P \cup \{s\}$
2: **repeat**
3: $s' \leftarrow$ MUTATE(s)
4: Evaluate s'
5: **if** $s \prec s'$ **then**
6: Discard s'
7: **else if** $s' \prec s$ **then**
8: $s \leftarrow s'$
9: $A \leftarrow A \cup \{s\}$
10: **else if** $s_i \prec s' : s_i \in A$ **then**
11: Discard s'
12: **else**
13: TEST(s, s', A) */* Determine the new current solution and whether to add s' to A */*
14: **end if**
15: **until** Stop criteria is met
16: **return** A

function in lines 5 to 14, which can be regarded as the fitness evaluator, and the non-dominated solution archive A. PAES maintains a single current solution s and at each iteration produces a single new candidate s' via random mutation. If s' dominates s, s' replaces s and it is added to the archive A. However, if s' is dominated by s or by any member of the archive, it is discarded.

2.4.3.2 Diversity preservation

In the case that solutions s and s' do not dominate each other and none of the solutions in A dominate s', function TEST(s, s', A) (line 13 in Algorithm 2.5) determines the new current solution and whether to add s' to A in the following way. If s' dominates any solution in A, s' is always accepted and archived, and the now dominated solutions in A are removed. If s' is non-dominated with respect to A, the histogram method for density estimation reviewed in Section 2.4.2.3 is used, where s' is accepted and/or archived based on the degree of crowding in its grid location [144].

Algorithm 2.6: SPEA2(P)

Input: Population $P = \{s_1, \ldots, s_{popSize}\}$ to be evolved
Output: Archive $A = \{s_1^*, \ldots, s_{popSize}^*\}$ result of SPEA2

$A \leftarrow \emptyset$
repeat
  Compute fitness of each individual in P and A
  $A \leftarrow \{s_i \in P \cup A \mid s_i$ is non-dominated with respect to $P \cup A\}$
  if $|A| > maxSize$ **then**
    TRUNCATE(A)
  else if $|A| < maxSize$ **then**
    Fill A with dominated individuals in P
  end if
  $P \leftarrow$ MAKENEWPOPULATION(A)
until Stop criteria is met
return A

2.4.4 Strength Pareto Evolutionary Algorithm 2

Another commonly used multi-objective EA is the Strength Pareto Evolutionary Algorithm 2 (SPEA2) proposed by Zitzler et al. [261]. The main characteristics of SPEA2 are [259]: it keeps an archive of all non-dominated solutions found during the evolutionary process and the offspring population is generated from the archived solutions. SPEA2 is presented in Algorithm 2.6 [42] and described below.

2.4.4.1 Fitness assignment

SPEA2 utilises the dominance count method reviewed in Section 2.4.1.3 to define the *strength* of each individual, considering both population and archive. The *raw fitness* of an individual is determined by the sum of the strengths of its dominator solutions. The fitness of an individual takes into account its raw fitness and, additionally, density information in order to discriminate between individuals having identical raw fitness. In this case, SPEA2 utilises an adaptation of the k-th nearest neighbour method reviewed in Section 2.4.2.2.

2.4.4.2 Diversity preservation

Line 6 in Algorithm 2.6 indicates the use of function TRUNCATE(A) when the size of the archive has been exceeded. This function removes one individual from the archive iteratively until the size is rectified. The individual to be removed is selected according to the minimum distance to another individual. If there are several individuals with minimum distance the tie is broken by considering the second minimum distance and so forth [261].

2.4.5 Non-dominated Sorting Genetic Algorithm II

Among the most successful multi-objective EAs is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) of Deb et al. [67]. In contrast to PAES and SPEA2, NSGA-II does not use an archive for storing the non-dominated solutions since it is elitist, that is, it preserves the best solutions from the combined parent and offspring populations. Fitness assignment and diversity preservation are explained below.

2.4.5.1 Fitness assignment

First, the dominance depth criterion reviewed in Section 2.4.1.3, which is also called *non-dominated sort* [118], is used to assign fitness to individuals. Deb et al. designed a faster algorithm to perform this sort in $O(FpopSize^2)$, where F is the number of objective functions, which is shown in Algorithm 2.7. The idea of this algorithm is, first, to find the overall non-dominated solutions, assign them to the first front, and record, for each non-dominated solution, the solutions they dominate. This is done in the cycle between lines 1 and 15, where all other dominated solutions are assigned their dominance rank. In the second main cycle between lines 17 and 30, solutions in the current front $\mathcal{F}_i$ are scanned and the rank of the solutions they dominate is

Algorithm 2.7: FASTNONDOMINATEDSORT(P)

Input: Population $P = \{s_1, \dots, s_{popSize}\}$ to be sorted
Output: Population P sorted according to non-domination

for all $s_i \in P$ **do**
  $Q_i \leftarrow \emptyset$ /* *Set of solutions dominated by* s_i */
  $c_i \leftarrow 0$ /* *Dominance count* */
  for all $s_j \in P$ **do**
    if $s_i \prec s_j$ **then**
      $Q_i \leftarrow Q_i \cup \{s_j\}$
    else if $s_j \prec s_i$ **then**
      $c_i \leftarrow s_i + 1$
    end if
  end for
  if $c_i = 0$ **then**
    $\varphi(s_i) \leftarrow 1$ /* s_i *belongs to the first non-dominated front* */
    $\mathcal{F}_1 \leftarrow \mathcal{F}_1 \cup \{s_i\}$
  end if
end for
$p \leftarrow 1$
while $\mathcal{F}_i \neq \emptyset$ **do**
  $R \leftarrow \emptyset$ /* *Solutions in the next front* */
  for all $s_i \in \mathcal{F}_p$ **do**
    for all $s_j \in Q_i$ **do**
      $c_j \leftarrow c_j - 1$
      if $c_j = 0$ **then**
        $\varphi j \leftarrow p$ /* s_j *belongs to the next front* */
        $R \leftarrow R \cup \{s_j\}$
      end if
    end for
  end for
  $p \leftarrow p + 1$
  $\mathcal{F}_p \leftarrow R$
end while

decreased by one. If there is a solution which rank is now zero, it belongs to the next front.

2.4.5.2 Diversity preservation

Fitness is the criteria used for parent selection, where fittest individuals are chosen for recombination, and is also used for selecting individuals to be taken to the next generation. Here, those solutions belonging to the fittest fronts are considered to

Algorithm 2.8: ASSIGNCROWDINGDISTANCE(P)

Input: Population $P = \{s_1, \ldots, s_M\}$ which crowding distance is going to be computed
Output: Population P with crowding distance assigned

for all $s_i \in P$ **do**
  $crowd(s_i) \leftarrow 0$
end for
for all objective functions f_j **do**
  $L \leftarrow \text{SORT}(P, f_j)$ /* *Sorts in a list the solutions according to* f_j */
  $crowd(L[1]) \leftarrow \infty$ /* *Boundary solutions are always selected* */
  $crowd(L[M]) \leftarrow \infty$
  for $k \leftarrow 2$ **to** $M - 1$ **do**
    $crowd(L[k]) \leftarrow crowd(L[k]) + \dfrac{f_j(L[k+1]) - f_j(L[k-1])}{f_j^{max} - f_j^{m}in}$
  end for
end for

be the next parent population. If the population size is exceeded in the last chosen front, density of solutions is estimated by using the kernel method reviewed in Section 2.4.2.1. Deb et al. introduced the *crowding distance*, which is a measure to quantify the average distance of one solution to its two nearest neighbours in the same front. Algorithm 2.8 shows this method, which is $O(FpopSize \log popSize)$ worst-case time complexity.

Thus, solutions belonging to that last selected front are sorted according to their crowding distance, and those individuals located in a not so crowded space are preferred. The overall NSGA-II is presented in Algorithm 2.9.

2.4.6 Comparison of multi-objective Evolutionary Algorithms

As was stated earlier, there are many multi-objective evolutionary approaches in the literature, some of them being more successful than others. The objective of this research does not contemplate an investigation of these algorithms. However, there are many surveys and comparative studies [65, 157, 141, 68, 40, 147, 146, 143, 230, 4, 64, 66] which analyse their performance, by means of the quality indicators presented in Section 2.3.2, and on which we can rely.

Algorithm 2.9: NSGA-II(P)

Input: Population $P = \{s_1, \ldots, s_{popSize}\}$ to be evolved
Output: Population $P^* = \{s_1^*, \ldots, s_{popSize}^*\}$ result of NSGA-II

$P_1 \leftarrow P$
$\mathcal{F} \leftarrow \textsc{FastNonDominatedSort}(P_1)$
$i \leftarrow 1$
repeat
  $Q_i \leftarrow \textsc{MakeNewPopulation}(P_i)$
  $R_i \leftarrow P_i \cup Q_i$
  $\mathcal{F} \leftarrow \textsc{FastNonDominatedSort}(R_i)$
  $P_{i+1} \leftarrow \emptyset$
  $k \leftarrow 1$
  while $|P_{i+1}| + |\mathcal{F}_k| \leq popSize$ **do**
    $\textsc{AssignCrowdingDistance}(\mathcal{F}_k)$
    $P_{i+1} \leftarrow P_{i+1} \cup \mathcal{F}_k$
    $k \leftarrow k + 1$
  end while
  $L \leftarrow \textsc{Sort}(\mathcal{F}_k, crowd)$
  $P_{i+1} \leftarrow P_{i+1} \cup \{L[1..(popSize - |P_{i+1}|)]\}$
  $i \leftarrow i + 1$
until Stop criteria is met
return $P^* \leftarrow P_i$

The evolutionary optimisers previously introduced have been considered in some of those studies, from which we can highlight the following findings and conclusions. In general, SPEA2 and NSGA-II perform equally well on convergence and diversity maintenance [143, 146, 4, 66], however the Truncate() operator of SPEA2 is more computationally expensive than the AssignCrowdingDistance() function of NSGA-II [65, 141]. Moreover, NSGA-II converges much faster near to the true Pareto optimum set than SPEA2 [68]. On the other hand, for a specific combinatorial problem, PAES faced problems in maintaining a consistent performance, as evident from the relatively large variance of the quality indicators [230].

Overall, according to Coello Coello [40], due to its clever mechanisms, NSGA-II's performance is so good, that it has become very popular in the last few years, establishing itself as a landmark against other multi-objective EAs. Consequently, we believe that NSGA-II is perhaps the chosen baseline algorithm with which to compare. Therefore, the performance of the developed multi-objective EA is going to be compared against that of NSGA-II.

2.5 Solution distance measures for combinatorial problems

Maintaining population diversity is crucial for EAs, in that their success depends on the avoidance of premature convergence and on the balance of the trade-off between exploration and exploitation of the search space [81]. For multi-objective EAs, it is also important to maintain diversity in order for the approximation set to contain solutions that represent the full Pareto front, rather than just a small portion of it. With the purpose of preserving population diversity, one can take into consideration the information provided by distance or similarity measures.

Different solution representations require different distance (or similarity) measures. For example, the *Hamming distance* [125] is the most common measure for binary representations, and for representations using a vector of real numbers, a variation of the *Minkowski-r-distance* [150] (e.g. Manhattan, Euclidean, and Chebychev distances) can be employed [221]. For problems which solutions are more suitable represented as a permutation, many methods have been proposed, like the *exact match distance* and *deviation distance* [205], the *R-permutation distance* [166] and the *edit distance* [247]. These distance measures for permutation-based representations are described below.

2.5.1 Exact match distance

The exact match distance [205] is similar to a Hamming measure acting on two strings (permutations) with a higher-order (than binary) alphabet. This distance function is relevant to permutation problems where the absolute position of a character in a string is important. When comparing two strings s and t of size l, a contribution to the overall distance between s and t is made for character position i, if $s(i) \neq t(i)$.

In this measure, the only quantity of interest is the number of exact character matches between two strings. The exact match distance $d_{\text{match}}(s,t)$ can be defined in terms of the exact match function $m(s,t,i)$ [205]:

$$d_{\text{match}}(s,t) = l - \sum_{i=1}^{l} m(s,t,i) \tag{2.14}$$

where

$$m(s,t,i) = \begin{cases} 1 & \text{if } s(i) = t(i) \ , \\ \\ 0 & \text{otherwise} \ . \end{cases} \tag{2.15}$$

2.5.2 Deviation distance

The deviation distance [205] considers absolute character position as an important problem-domain property, and the amount of positional deviation between matching characters is used in the calculation of distance measure. This distance is relevant for problems where, for two strings s and t of size l, the degree of positional deviation of a character x between s and t has some problem-specific importance.

The deviation distance $d_{\text{deviation}}(s,t)$ is based on the positional perturbation of one character in string s to its matching position in string t and is defined as the sum of the absolute value of the displacement of character $s(i)$ [205]:

$$d_{\text{deviation}}(s,t) = \sum_{i=1}^{l} \frac{|i-j|}{l-1}, \qquad s(i) = t(j) \ . \tag{2.16}$$

2.5.3 R-permutation distance

Martí et al. [166] proposed the R-permutation distance for problems where the relative position of the elements is more important than the absolute position. The R-permutation distance is applicable to problems where, for two strings s and t of size l, it is relevant that the contiguous characters $s(i)$ and $s(i+1)$ are also adjacent in string t.

The R-permutation distance $d_{\text{permutation}}(s,t)$ is defined as the number of times $s(i+1)$ does not immediately follow $s(i)$ in t [244, 220]:

$$d_{\text{permutation}}(s,t) = \sum_{i=1}^{l} r(i) \ , \tag{2.17}$$

where

$$r(i) = \begin{cases} 1 & \text{if } \exists\, j : s(i) = t(j) \text{ and } s(i+1) = t(j+1) \ , \\ 0 & \text{otherwise} \ . \end{cases} \tag{2.18}$$

2.5.4 Edit distance

The edit distance is based on *Levenshtein distance* [160], which was first introduced in the field of error correcting codes for dealing with binary strings. The Levenshtein distance between two binary strings is the minimal number of edit operations required to transform one of the strings into the other. These edit operations are defined as [219]: (*i*) reversal, $0 \to 1$ or $1 \to 0$, (*ii*) deletion, $0 \to \Lambda$ or $1 \to \Lambda$, and (*iii*) insertion, $\Lambda \to 0$ or $\Lambda \to 1$, where Λ is the null-character, specifying the absence of a character, i.e. $|\Lambda| = 0$.

Wagner and Fischer [247] extended the work of Levenshtein, first, by considering that strings are composed of any finite alphabet, and second, by widening the reversal operation, which became substitution when one character is converted into another [220]. They also provided a dynamic programming algorithm to calculate the edit distance, which time complexity is $O(N^2)$, N being the length of the strings. A commonly used bottom-up dynamic programming algorithm, based on that of Wagner and Fischer, is presented in Algorithm 2.10, which is available from many world wide web sites[4].

[4]For example, Wikipedia: `http://en.wikipedia.org/wiki/Levenshtein_distance`.

Algorithm 2.10: EDITDISTANCE(s, t)

Input: Strings $s, |s| = M$, and $t, |t| = N$
Output: Edit distance between s and t

/* *d is a table with $M+1$ rows and $N+1$ columns* */
for $i \leftarrow 1$ to $M+1$ **do**
  $d[i,1] \leftarrow i-1$ /* *deletion* */
end for
for $j \leftarrow 1$ to $N+1$ **do**
  $d[1,j] \leftarrow j-1$ /* *insertion* */
end for
for $j \leftarrow 2$ to $N+1$ **do**
  for $i \leftarrow 2$ to $M+1$ **do**
    if $s[i] = t[j]$ **then**
      $d[i,j] \leftarrow d[i-1,j-1]$
    else
      $d[i,j] \leftarrow \min\,(d[i-1,j]+1,$ /* *deletion* */
        $d[i,j-1]+1,$ /* *insertion* */
        $d[i-1,j-1]+1)$ /* *substitution* */
    end if
  end for
end for
return $d[M+1, N+1]$

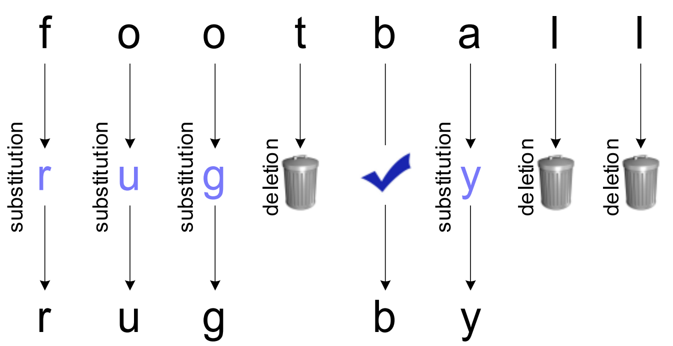


Figure 2.7: Edit distance between words **football** and **rugby**: Letters **foo** are substituted by **rug**. Letter **t** is deleted, while **b** is kept in its place. Letter **a** is substituted by **y**, and finally letters **ll** are deleted. In total, four substitutions and three deletion operations were performed, hence, the edit distance is 7.

An example of the computation of the edit distance is presented in Figure 2.7, which shows the edit operations to transform the string **football** into **rugby**. Strings are analysed from left to right, thus characters **f**, **o**, and **o** are substituted by **r**, **u**, and **g**, respectively. Then, character **t** is deleted, while **b** remains in its place. Character **a** is substituted by **y**. Finally, the last characters **l** and **l** are deleted. In total, four substitutions and three deletion operations were performed, hence, the edit distance between these two strings is 7.

2.5.5 Distance measures for solutions to Vehicle Routing Problems

A solution to any variant of the Vehicle Routing Problem can be regarded as a set of routes, and a route as a set of customers. Moreover, a route must list the customers ordered according to service precedence. Hence, if we are interested in measuring the distance (or similarity) between two solutions to the VRP, we need to compare the number of routes in the solutions, the customers served in each route, and the customer service sequences.

Having this in mind, the exact match distance and the deviation distance would not be an appropriate measure, since they are related to the absolute position of the characters in the strings. In contrast, the R-permutation distance and the edit distance provide information regarding the relative position of the characters. However, solutions to the VRP can be encoded in many different ways, e.g. by changing the order of the routes in the solution or by reversing some routes. A distance measure for solutions to the VRP should be able to take this into account and recognise the fact that the distance between two solutions encoded in different ways should be zero [220]. In this case, the edit distance could potentially provide a more reliable information.

In fact, Sörensen [219] proposed to use the edit distance to quantify the distance between two solutions to the VRP in the following way. Let $\mathcal{R}$ and $\mathcal{Q}$ be solutions to the VRP. Moreover, let $\mathcal{R}$ be split into parts $\{r_1, \ldots, r_K\}$, and $\mathcal{Q}$ into parts $\{q_1, \ldots, q_L\}$, which actually represent the designed routes in each solution. Then, the edit distance $\delta_{\text{edit}}(\mathcal{R}, \mathcal{Q})$ between solutions $\mathcal{R}$ and $\mathcal{Q}$ is given by the *assignment* *problem* [32]

$$\delta_{\text{edit}}(\mathcal{P}, \mathcal{Q}) = \min \sum_{i=1}^{K} \sum_{j=1}^{L} \textsc{EditDistance}(r_i, q_j)\ x_{ij}\ , \tag{2.19}$$

subject to

$$x_{ij} \in \{0, 1\}, \qquad \forall\, i, j\ , \tag{2.20}$$

$$\sum_j x_{ij} = 1, \qquad \forall\, i\ , \tag{2.21}$$

$$\sum_i x_{ij} = 1, \qquad \forall\, j\ , \tag{2.22}$$

where $x_{ij} = 1$ if r_i is assigned to q_j, 0 otherwise. One of the most widely used techniques to solve assignment problems is the *Hungarian method* [151], for which the most efficient known algorithm is $O(\max(K^2L, KL^2))$, or, if $K \approx L$, $O(K^3)$.

2.6 Summary

This chapter described what combinatorial optimisation problems are, and presented three heuristic methods that are commonly used to solve this kind of problems, e.g. Vehicle Routing Problems, which are local search, Tabu Search, and Evolutionary Algorithms. Multi-objective combinatorial optimisation problems were introduced, along with three quality indicators which evaluate algorithm performance regarding multiple criteria. A description of how Evolutionary Algorithms can handle problems with multiple objectives was given. Finally, a number of solution distance measures for combinatorial problems were explained.

A combinatorial optimisation problem consists in finding the best solution to an instance of a given problem. They consider discrete variables and one typically looks for a solution in the form of a sub-set, permutation, or graph. The problem of interest in this thesis, the Vehicle Routing Problem, is a combinatorial problem.

Small instances of combinatorial optimisation problems can be solved by enumerating all possible solutions and selecting the one that have the best objective function value. Unfortunately, this is not a practical method for solving instances of realistic sizes, in which case one can consider heuristic methods for solving the problem. There are in the literature many problem-specific and general heuristic methods. Local search, Tabu Search, and Evolutionary Algorithms (EA) are among the general heuristics that have been considered for solving a number or variants of the Vehicle Routing Problem. An EA consists in a cyclic process which recombines and mutates a population of solutions. Each solution is assigned a fitness value according to its relative quality to rest of the population, and is the fitness value which drives the natural selection process.

If the combinatorial optimisation problem considers only one objective function, it is clear that the best solution will be that providing the minimum (or maximum) value of the objective function. However, if more objective functions are considered, the problem consists in finding a set of solutions which provide suitable trade-offs between the objectives.

Performance comparison of multi-objective optimiser can not be made as straightforwardly as in the single-objective case, since comparing the best results, or the average of them, could be deceptive when optimising multiple objectives. Instead, the closeness to the optimum solutions and the diversity of the obtained solutions must be considered. There are many multi-objective quality indicators that have been developed to measure these criteria. In particular, the convergence metric measures the distance from a set of solutions to a reference set, the coverage metric evaluates to some extent the diversity of the solutions, and the hypervolume metric quantifies the size of the delimited objective space.

EAs can be extended for tackling multi-objective problems, the difference lies in how fitness is assigned to solutions and the survival of individuals. While in the single-objective case fitness is assigned to an individual directly (or indirectly) proportional to its objective function value, in the multiple criteria case it may be assigned by counting how many solutions it dominates, how many solutions dominate it, or by grouping the population into non-dominated fronts and each front is assigned a fitness value. On the other hand, the survival of individuals is generally performed by considering the density of individuals in a given region, for which statistical density estimation techniques may be used.

Three of the most popular multi-objective EAs are the Pareto Archived Evolution Strategy (PAES) [144], the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [262] and the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [67]. PAES considers one single solution which is iteratively mutated for finding non-dominated solutions, which are archived depending on their quality and on the crowding of the region where they are located in the objective space. SPEA2 also considers an external population to archive the non-dominated solutions found during the evolutionary process. It assigns fitness to solutions according to their Pareto dominance and to the density of solutions. NSGA-II uses the non-dominated sort, which groups individuals into fronts for fitness assignment, and computes the crowding distance of the solutions in each front for estimating density. The metrics mentioned above have been used to compare these evolutionary approaches. Results suggest that NSGA-II is one of the best algorithms with which to compare new proposals.

The success of any EA depends on several factors, e.g. crossover and mutation operators. As will be shown later, preserving and maintaining population diversity help the evolutionary operators in achieving a wider exploration and exploitation of the search space. In this respect, one can measure how different or similar the solutions in the population are and use this information to preserve diversity. There are many distance and similarity measures, but all of them depend on the solution encoding. Particularly, for a permutation-based representation, which is the most logical encoding for solutions to the Vehicle Routing Problem, the edit distance measures how different two solutions are by counting the number of replacement, insertion, and deletion operations required to transform one string into another.

Chapter 3

Vehicle Routing Problems

The Vehicle Routing Problem (VRP) is one of the most studied among the combinatorial optimisation problems, due to both its practical relevance to many real-world applications in transportation and distribution logistics [14], and its considerable difficulty [238]. The VRP generalises [234] the well-known *Travelling Salesman Problem* [123] and is also related to the *Bin Packing Problem* [43]. Its main objective is to obtain the lowest-cost set of routes for the distribution of goods, between depots and customers, by means of utilising a fleet of vehicles, where *lowest-cost* can mean many things. Since the problem was proposed by Dantzig and Ramser [54], cost has been associated with the fleet size and the total travel distance, though there are several other sources of cost, such as the delivery time, workload imbalance, waiting time and makespan, among many others [135].

This problem has, additionally to the several objectives to be optimised, a number of constraints to be considered, which actually call for the many variants of the VRP that have been introduced in the literature during the years. In fact, some efforts have been made to classify and model these variants [70, 72]. Common restrictions lie in the vehicles capacity, the maximum distance vehicles may travel, the arrival time of vehicles to customer locations, and customers service precedence, to name only a few.

Small instances of Vehicle Routing Problems can be solved to optimality by means of using exact methods [107, 233, 234, 238, 13, 15, 138], however, the computation time required increases considerably for larger instances [71]. Actually, it has been demonstrated to belong to the NP-hard class [159], which means that there is no known efficient method for optimally solving the problem in polynomial time. This is the main reason why many published studies have considered the use of heuristic methods.

Current heuristics for solving the VRP can be grouped into the following categories [28]: (*i*) construction heuristics, (*ii*) improvement heuristics, and (*iii*) metaheuristics. Construction heuristics are algorithms aiming at designing initial solutions, building a route for each vehicle using decision functions for the selection of the customer to be inserted in the route and the insertion position within the route.

Most of the recently published heuristics use a two-phase approach: Firstly, a construction heuristic is used to generate a feasible initial solution. Secondly, an iterative improvement heuristic is applied to the initial solution. These route improvement methods iteratively modify the current solution by performing local searches.

To escape from local optimum, the improvement procedure can be embedded in a metaheuristic framework. In general, a metaheuristic is guided by intelligent search strategies to avoid getting trapped in sub-optimum regions of the search space.

The surveys by Laporte et al. [156], Laporte and Semet [154], Gendreau et al. [109], Cordeau et al. [46], Bräysy et al. [28], Bräysy and Gendreau [26, 27], and Cordeau et al. [49], and the more recent by Baldacci et al. [13, 15], Marinakis and Migdalas [165], Potvin [190, 191], Eksioglu et al. [83], Laporte [153], and Gendreau and Tarantilis [105], provide a complete list of studies concentrating on the use of a number of construction and improvement heuristics, and metaheuristics for solving several variants of the VRP. Additionally, the novel survey by Jozefowiez et al. [135] examines a number of multi-objective studies related to the solution of classical and practical VRPs.

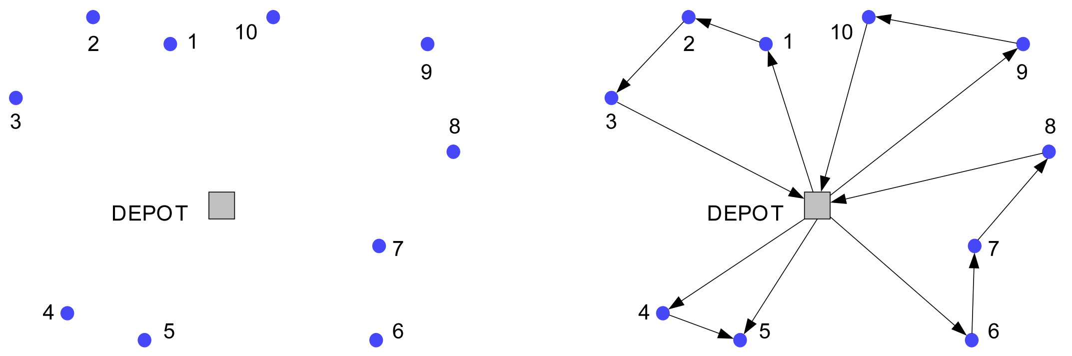
This chapter describes the notation and the problem in detail for two variants of the VRP, namely the Capacitated VRP (CVRP) and the VRP with Time Windows (VRPTW), which are the principal problems studied in this thesis. These problems have been considered for study due to: (*i*) being the most basic VRPs, they are perhaps the best starting point for researchers who are willing to immerse themselves in the problem, (*ii*) there are plenty of studies regarding their solution that can be deemed as the baseline, and (*iii*) specifically for VRPTW, there are benchmark instances that can be use for actual multi-objective optimisation. Other VRPs commonly found in the literature have also been considered for description. Additionally, a review of three classical construction heuristics is given, which is followed by an overview of some studies that have tackled the problems of interest. Finally, VRPs regarding multiple objectives, along with a literature survey, are introduced.

3.1 Capacitated VRP

The most elemental variant of the VRP is the Capacitated Vehicle Routing Problem (CVRP), which takes into account a homogeneous fleet of vehicles with restricted capacity as an approximation to real transportation problems.

The required information to define an instance of the CVRP is the following [236]:

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N\}$ of $N + 1$ vertices, representing the geographical location of the depot and customers.



(a) Depot and customers locations (b) Designed routes

Figure 3.1: An example instance of the CVRP and a potential solution to it.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N\}$. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i) and has a demand of goods $q_i > 0$.

Depot The special vertex v_0, located at (x_0, y_0), with $q_0 = 0$, is the depot, from where customers are serviced and a fleet of vehicles is based.

Vehicles There is a homogeneous fleet of vehicles available to deliver goods to customers, departing from and arriving at the depot, which have capacity $Q \geq \max\{q_i : i = 1, \ldots, N\}$.

The information above allows the definition of the travel distance d_{ij} between vertices v_i and v_j, which is often considered to be the Euclidean distance, i.e.

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}\ . \tag{3.1}$$

The problem consists in designing a lowest-cost set $\mathcal{R} = \{r_1, \ldots, r_K\}$ of K routes, such that each route begins and ends at the depot, and each customer is serviced by exactly one vehicle. Thus, each vehicle is assigned a set of customers that it has to supply. Figure 3.1 shows an example instance of the CVRP with ten customers and one potential solution to it comprising of four such routes.

If $r_k = \langle u(1,k), \ldots, u(N_k,k) \rangle$ is defined as the sequence of N_k customers supplied in the k-th route, where $u(i,k)$ is the i-th customer to be visited in route r_k, then $\mathcal{V}'_k = \{u(1,k), \ldots, u(N_k,k)\}$ is the set of customers serviced in route r_k. Note that the depot does not appear explicitly in this notation, however, it has to be taken into account before the first and after the last customers when computing costs, i.e. $u(0,k) = u(N_k+1,k) = v_0$. Then

$$d(r_k) = \sum_{i=0}^{N_k} d_{u(i,k)\ u(i+1,k)} \tag{3.2}$$

is the travel distance associated with route r_k, and

$$q(r_k) = \sum_{i=1}^{N_k} q_{u(i,k)} \tag{3.3}$$

is the total demand of the customers serviced in route r_k.

Having defined the CVRP, there are at least two obvious objective functions to concentrate on minimising, namely the number of routes (or vehicles)

$$f_1(\mathcal{R}) = |\mathcal{R}| = K , \tag{3.4}$$

and the total travel distance

$$f_2(\mathcal{R}) = \sum_{r_k \in \mathcal{R}} d(r_k) = \sum_{r_k \in \mathcal{R}} \sum_{i=0}^{N_k} d_{u(i,k)\ u(i+1,k)} , \tag{3.5}$$

subject to the route demand $q(r_k)$, associated with customers serviced in route r_k, which must not exceed the vehicle capacity Q, i.e.

$$q(r_k) = \sum_{i=1}^{N_k} q_{u(i,k)} \leq Q, \quad \forall\, r_k \in \mathcal{R} . \tag{3.6}$$

In addition to capacity, route r_k might also be restricted to certain maximum length D, i.e.

$$d(r_k) = \sum_{i=0}^{N_k} d_{u(i,k)\ u(i+1,k)} \leq D, \quad \forall\, r_k \in \mathcal{R} . \tag{3.7}$$

The proposed evolutionary approach, which will be presented in Chapter 5, was tested on instances of this problem, considering the simultaneous minimisation of the number of routes, $f_1(\mathcal{R})$ in (3.4), and the total travel distance, $f_2(\mathcal{R})$ in (3.5), subject to the capacity constraint (3.6) and the maximum route length constraint (3.7), and giving the same priority to both objectives.

3.1.1 Benchmark sets of the Capacitated VRP

There exist several benchmark sets of the CVRP in the literature, two of which, namely that of Christofides et al. [37] and that of Rochat and Taillard [204], are among the most used. These instances were designed to be solved as single-objective problems, considering the minimisation of the number of routes, and for a given number of routes, the minimisation of the travel distance. These data sets are publicly available from several world wide web sites[1].

3.1.1.1 Christofides et al.'s benchmark set

One of the typical public benchmark sets of the CVRP is the one of Christofides et al. [37]. It consists of 14 instances of sizes varying from 50 to 199. Table 3.1 presents the characteristics and the best-known results for this set. Columns in this table describe, respectively from left to right, the instance name, number of customers, geographical distribution of customers, vehicle capacity, maximum length of any route, and stop time at customer location. The last three columns correspond to the number of routes (R), travel distance (D), and author who found the best-known result. In instances vrpnc1 to vrpnc10 the customers are randomly distributed, whereas in problems vrpnc11 to vrpnc14 customers are located in clusters. For illustrative purposes, Figure 3.2 shows the geographical location of the customers for two instances in this set.

[1]For example, The VRP Web: `http://neo.lcc.uma.es/radi-aeb/WebVRP/`.

Instance	N	Location	Q	Max. Length	Stop Time	Best-known R	Best-known D	Best-known Author
vrpnc1	50	Random	160	∞	0	5	524.61	T
vrpnc2	75	Random	140	∞	0	10	835.26	T
vrpnc3	100	Random	200	∞	0	8	826.14	T
vrpnc4	150	Random	200	∞	0	12	1028.42	T
vrpnc5	199	Random	200	∞	0	17	1291.29	MB
vrpnc6	50	Random	160	200	10	6	555.43	T
vrpnc7	75	Random	140	160	10	11	909.68	T
vrpnc8	100	Random	200	230	10	9	865.94	T
vrpnc9	150	Random	200	200	10	14	1162.55	T
vrpnc10	199	Random	200	200	10	18	1395.85	RT
vrpnc11	120	Clustered	200	∞	0	7	1042.11	T
vrpnc12	100	Clustered	200	∞	0	10	819.56	T
vrpnc13	120	Clustered	200	720	50	11	1541.14	T
vrpnc14	100	Clustered	200	1040	90	11	866.37	O

Author: MB: Mester and Bräysy [170] RT: Rochat and Taillard [204] T: Taillard [225]

Table 3.1: Characteristics and best-known results to instances in the Christofides et al.'s benchmark set.

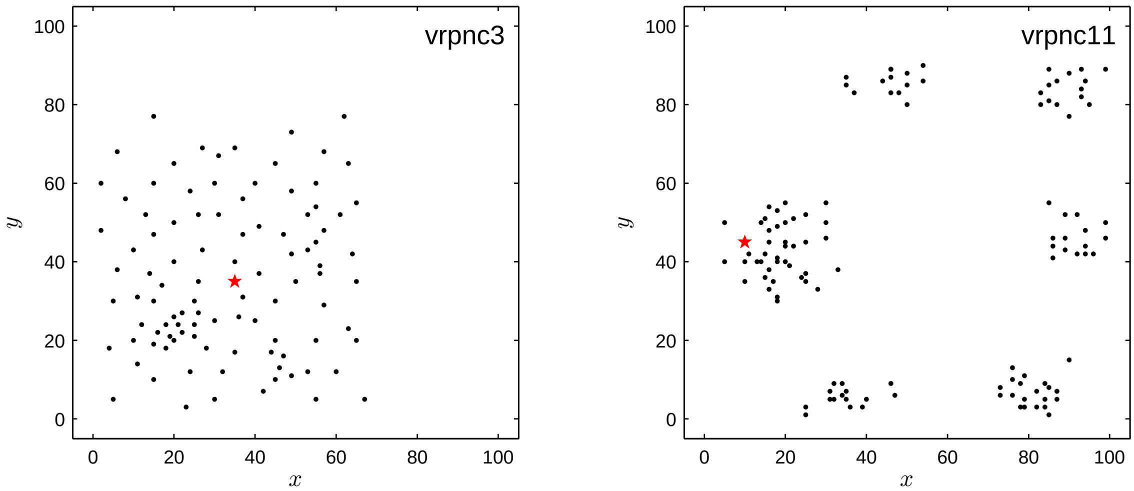


Figure 3.2: Geographical location of customers in instances vrpnc3 and vrpnc11 from the Christofides et al.'s benchmark set. Location with a star represents the depot.

Instances vrpnc6 to vrpnc10, vrpnc13 and vrpnc14, consider a maximum route length constraint, feature that could suggest the service of less customers in each route, thus forcing the use of more vehicles. The geographical location of customers and the vehicle capacity for the same instances are as in vrpnc1 to vrpnc5, vrpnc11 and vrpnc12, respectively.

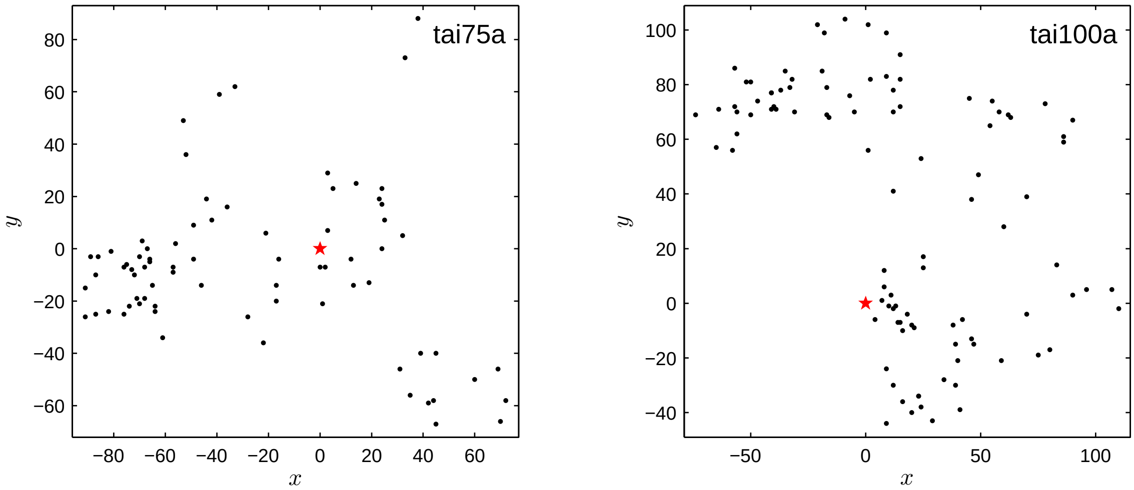


Figure 3.3: Geographical location of customers in instances tai75a and tai100a from the Rochat and Taillard's benchmark set. Location with a star represents the depot.

3.1.1.2 Rochat and Taillard's benchmark set

Another commonly used CVRP benchmark set is the one of Rochat and Taillard [204], who generated a set of test instances where customers are spread in several clusters, and the number of such clusters and their compactness are quite variable. Figure 3.3 shows how customers are geographically located in two of these instances. The demands ordered by the customers are exponentially distributed, hence one customer may use almost the entire capacity of one vehicle. They generated four problems of each size $N = 75, 100, 150$. The characteristics and best-known results for these instances are shown in Table 3.2, which format is similar to that of Table 3.1

3.1.2 Classical construction heuristics

In the literature we can find several heuristics that have been proposed for constructing solutions to the CVRP. Since many of the approaches that are going to be reviewed later refer to these heuristics, this section presents a description of two of the most commonly used classical methods.

Instance	N	Location	Q	Best-known		
				R	D	Author
tai75a	50	Clustered	1445	10	1618.36	T
tai75b	75	Clustered	1679	10	1344.62	T
tai75c	100	Clustered	1122	9	1291.01	T
tai75d	150	Clustered	1699	9	1365.42	T
tai100a	199	Clustered	1409	11	1291.29	GTA
tai100b	120	Clustered	1842	11	1940.61	MB
tai100c	50	Clustered	2043	11	1406.2	GTA
tai100d	75	Clustered	1297	11	1580.46	NB
tai150a	100	Clustered	1544	15	3055.23	T
tai150b	150	Clustered	1918	14	2656.47	GTA
tai150c	199	Clustered	2021	15	2341.84	T
tai150d	120	Clustered	1874	14	2645.39	T

Author: GTA: Gambardella et al. [97] NB: Nagata and Bräysy [179] T: Taillard [225]
MB: Mester and Bräysy [170]

Table 3.2: Characteristics and best-known results to instances in the Rochat and Taillard's benchmark set.

3.1.2.1 Clarke and Wright's savings heuristic

Clarke and Wright [38] proposed the *savings algorithm*, which is perhaps the most widely-known heuristic for the CVRP. It defines, for two customers v_i and v_j, the saving s_{ij} with respect to the depot v_0 as

$$s_{ij} = d_{i0} + d_{0j} - d_{ij} \ , \tag{3.8}$$

that is, the saving in cost of merging the routes $\langle v_0, \ldots, v_i, v_0 \rangle$ and $\langle v_0, v_j, \ldots, v_0 \rangle$ into a single route $\langle v_0, \ldots, v_i, v_j, \ldots, v_0 \rangle$.

The algorithm works as follows [154]. First, compute the savings $s_{ij}, \forall\ v_i, v_j \in \mathcal{V}'$, $v_i \neq v_j$. Then, create N routes $\langle v_0, v_i, v_0 \rangle, \forall\ v_i \in \mathcal{V}'$, and sort the savings in non-increasing order. Consider in turn each route $\langle v_0, v_i, \ldots, v_j, v_0 \rangle$. Determine the first saving s_{ki} or s_{jl} that can be feasibly used to merge the current route with another route ending with the arc (v_k, v_0) or starting with the arc (v_0, v_l). Implement the merge and repeat this operation to the current route. If no feasible merge exists, consider the next route and reapply the same operations. Stop when no route merge is feasible.

There is a parallel version of this algorithm which, after sorting the savings, starts from the top of the savings list and execute the following [154]. Given a saving s_{ij}, determine whether there exist two routes, one starting with the arc (v_0, v_j) and the other ending with the arc (v_i, v_0), that can be feasibly merged. If these routes exist, combine them by deleting the arcs (v_0, v_j) and (v_i, v_0) and introducing the arc (v_i, v_j).

This algorithm scores very high on simplicity and speed. It contains no parameters and is easy to code. However, the lack of flexibility is probably not a good feature of this algorithm, in the sense that, while additional constraints can, in principle, be incorporated, this usually results in a sharp deterioration in solution quality. This can be explained by the fact that the algorithm is based on a greedy principle and contains no mechanism to undo early unsatisfactory route merges [48].

3.1.2.2 Gillet and Miller's sweep algorithm

Gillett and Miller [113] popularised the *sweep algorithm*[2], which initially forms feasible clusters by rotating a *ray* centred at the depot and then a route is obtained for each cluster by solving a TSP.

A simple implementation of this method is as follows [156]. Assume each vertex v_i is represented by its polar coordinates (θ_i, ρ_i), where θ_i is the angle and ρ_i is the ray length. Assign a value $\theta_i^* = 0$ to an arbitrary vertex v_i^* and compute the remaining angles centred at 0 from the initial ray $(0, v_i^*)$. Sort the vertices in increasing order of their θ_i. Choose an unused vehicle k and, starting from the unserviced vertex having the smallest angle, assign vertices to vehicle k as long as its capacity or the maximal route length is not exceeded. If there are vertices that have not been

[2]Laporte et al. [156] suggest its origins are alluded to Wren [252] and Wren and Holliday [253].

routed, choose the next available vehicle. Finally, optimise each route separately by solving the corresponding TSP.

This algorithm scores high on simplicity, but does not seem to be superior to the savings algorithm both in terms of accuracy and speed. It is also rather inflexible, since the greedy nature of the sweep mechanism makes it difficult to accommodate additional constraints and the fact that the algorithm assumes a planar structure severely limits its applicability. In particular, the algorithm is not well suited to instances defined in an urban setting with a grid street layout [48].

A natural extension of the sweep algorithm is to generate several routes, called *petals*, and make a final selection by solving a *set partitioning problem* [12]. Some examples of this approach are the 1-*petal algorithm* of Foster and Ryan [91] and Ryan et al. [208], and the 2-*petal heuristic* of Renaud et al. [201] where, in addition to single routes, double vehicle routes are also generated. These extensions provide both accuracy and speed gains with respect to the sweep algorithm [48].

3.1.3 Overview of metaheuristic approaches

The Capacitated VRP has been subject of extensive research, perhaps it is the most studied VRP [165]. In order to set a baseline for the present investigation, the purpose of this section is to survey some of the approaches that have proved to be good solvers of this variant of the problem.

This review starts with some Tabu Search (TS) approaches. Gendreau et al. [106] proposed their *TABUROUTE*, which main characteristics are the heuristics *GENI* and *US*. The first, GENI, is a generalised insertion routine, in which a customer is removed from its route and it may be inserted only into a route containing one of its closest neighbours, and every insertion is executed simultaneously with a local re-optimisation of the current tour. US is a post-optimisation procedure that successively removes and reinserts every customer using GENI. The GENIUS heuristic is used in both to construct initial solutions and to search the neighbourhood.

Taillard [225] proposed a parallel TS which partitions the problem into several clusters and solves them separately. Initial solutions are obtained by the use of the λ-interchange local search on a solution with N routes, being N the number of customers. The same local search heuristic is used to define the neighbourhood. This method executes, every 200 iterations, the algorithm of Volgenant and Jonker [245] for exactly solving an instance of the TSP. This TS was enhanced by Rochat and Taillard [204], who proposed a method that considers an *adaptive memory*: a pool of routes taken from the best solutions visited during the search. Its purpose is to provide new starting solutions to the TS through a probabilistic selection and combination of routes extracted from the memory. The probability of selecting a particular route depends on the quality of the solution to which the route belongs. After the TS executes, a post-optimisation process is applied, specifically a set partitioning problem [12] is solved, using the routes in the adaptive memory to return the best possible solution.

Toth and Vigo [239] presented the *Granular* Tabu Search (GTS). This approach restricts the search to granular neighbourhoods, which only consider moves generated by the arcs in a reduced set. This reduced set only contains *short* arcs, being an arc short if its cost is not greater than the *granularity threshold* value. The algorithm is initialised with the solution obtained by the savings heuristic. GTS uses a multiple granular neighbourhood based on the local search heuristics 2-opt, 3-opt, and Or-opt. Visiting infeasible solutions is allowed during the search, however, their costs are modified by adding penalties to the routing cost.

Evolutionary approaches have also been considered for tackling the CVRP. Baker and Ayechew [11], for example, proposed a Genetic Algorithm (GA) that initialises the population with random and structured solutions. Two methods were used to generate structured solutions, namely the sweep heuristic and the cluster-first, route-second algorithm of Fisher and Jaikumar [87]. The GA considers the 2-point crossover [57] and a mutation which selects two customers and exchanges their position. They also applied the two local search heuristics 2-opt and λ-interchange after a number of generations.

Prins [195] proposed a hybrid GA which executes a local search as the mutation operator. This GA uses a permutation-based representation and includes the *Split* function to retrieve the best solution from the encoding. The algorithm starts with an initial population formed by three solutions obtained after applying the savings and sweep heuristics, and the Mole and Jameson [175] heuristic. The rest of the population are random permutations. Parents are selected with the binary tournament method and recombined using the order crossover [181] to produce two children, one of them randomly chosen to be mutated. The mutation is a local search procedure that tests nine specific simple moves for every pair of vertices. Each iteration of the local search is stopped at the first improving move. The solution is then modified and the process repeated until no further saving can be found. The new solution replaces a mediocre individual drawn above the median.

Morgan and Mumford [176] proposed a hybrid GA which includes a *perturbation method* to randomly and lightly move all customer coordinates and use the resulting positions to produce a set of routes using the savings heuristic. This method is used to generate an initial population. They used three different crossover operators, namely the single-point and 2-point crossover [57], and uniform crossover [3, 224]. The mutation procedure randomly selects a number of customers and applies the perturbation method to those customers.

The *Active Guided Evolution Strategy* (AGES) of Mester and Bräysy [170] combines a *Guided Local Search* (GLS) [246] with an (1+1)-Evolution Strategy (ES). Initial solutions are created with a modified version of the savings heuristic. The iterative improvement phase starts with the GLS based on the two local search heuristics λ-interchange and 2-opt, and on the relocate heuristic of Savelsbergh [210], considering a reduced neighbourhood. After no improvement, the ES is applied to remove a selected set of customers and reinsert them at optimum cost. Mester et al. [171] modified this method: they considered an *Adaptive Variable Neighbourhood*, which is created by dividing the problem in smaller geographical regions, and a number of routes within each region are grouped in order to apply the ES to them.

Alba and Dorronsoro [5, 6] implemented a *cellular* GA (cGA), in which the population is structured in a specified topology and the genetic operations may only take place in a small neighbourhood of each individual. cGA uses the edge recombination crossover of Whitley et al. [250] and includes three mutation operators that insert, swap and invert customers, which are applied with equal probability. Additionally, a local search step is considered after mutation, which implements the 2-opt and λ-interchange heuristics.

Nagata [178] and Nagata and Bräysy [179] proposed to use a modified version of the *edge assembly crossover* (EAX) [180], originally introduced in a GA for solving the TSP. They extended EAX by handling vehicle capacity constraint violations with a penalty function method, based on 2-opt and λ-interchange local search, that modifies solutions in order to be feasible. Initial solutions are created from routes with one single customer, which are applied the modification method mentioned above and finally a further local search is executed. Afterwards, each time an offspring is generated by means of EAX, the modification and local search procedures are applied.

Prins [196] presented a hybrid method GRASP $\times$ ELS which combines a *Greedy Randomised Adaptive Search Procedure* (GRASP) [86] and an *Evolutionary Local Search* (ELS) [251]. GRASP can be thought as a multi-start local search, in which each initial solution is generated using one greedy randomised heuristic and improved by local search. ELS uses a population of only one individual and there is no need for recombination. GRASP is the main loop of this approach, which is used to provide the nested ELS with distinct initial solutions and can be viewed as a diversification mechanism. Initial solutions are built with the savings heuristic, which are then submitted to the 3-opt local search. The mutation in ELS consists in swapping two distinct customers randomly selected and, to adapt the mutation level, a number of successive swaps are executed. Finally, the 2-opt, Or-opt and λ-interchange local search techniques are applied.

Another metaheuristic that has been used for solving VRPs is Ant Colony Optimisation (ACO) [76], which simulates the behaviour of ants in their search for food sources. In the approach of Bullnheimer et al. [31], ants successively choose customers to visit until each customer has been visited. Whenever the choice of another customer would lead to an infeasible solution, for reasons of vehicle capacity or total route length, the ant returns to the depot and starts a new tour. For the selection of customers that have not yet been served, the savings criterion is taken into account. Additionally, this algorithm introduces candidate customer lists, which, for every customer v_i, considers a restricted set of locations to be visited immediately after v_i. Finally, the 2-opt local search heuristic is applied after each iteration.

To conclude the CVRP survey, Pisinger and Ropke [189] designed an *Adaptive Large Neighbourhood Search* (ALNS), which removes between 30% and 40% of the customers from their routes, and reallocates them to other vehicles. They considered seven different removal and two different insertion heuristic methods: the heuristics to be used are chosen stochastically according to the weights the methods are assigned. These weights are dynamically adjusted according to their efficiency during the process. The initial solutions are generated with the regret-2 heuristic of Potvin and Rousseau [193].

3.1.4 Results from previous studies

The surveyed studies tested their approaches on either the Christofides et al.'s or the Rochat and Taillard's benchmark sets, or on both. The difference between their best obtained results and the best-known is shown in Table 3.3, for each instance of the Christofides et al.'s benchmark set, and in Table 3.4, for each instance of the the Rochat and Taillard's benchmark set. Some of these approaches were also tested on VRPTW benchmark instances and their results will be shown later in Section 3.2.4. Pisinger and Ropke [189] tested their ALNS on the Christofides et al.'s data set, however, they did not provide detailed results, but only mention that they are 0.11% above the best-known results, considering the best result after 10 repetitons, and 0.31% above considering the average.

These tables present, for each instance (row) and author (column), the per cent difference with the best-known result. In Table 3.3 we observe that the algorithm of Mester and Bräysy [170] (MB) found the best-known solutions for seven out of the 14 instances, while that of Nagata and Bräysy [179] (NB) found the best solutions for 13 of them. Solutions from the methods of Taillard [225] (T), Prins [195] (P04), and Prins [196] (P09) are less than 0.1% above the best-known. In contrast, the ACO of Bullnheimer et al. [30] (BHS) reports the highest average difference (4.43%).

On the other hand, in Table 3.4 we see that the ACO of Gambardella et al. [97] (GTA), which approach will be presented in Section 3.2.3, achieved the current best-known results for three of the 12 instances, though they did not report the results for the remaining test problems. Morgan and Mumford [176] (MM), Mester and

Instance	Best	T	GHL	BHS	TV	P04	BA	MM	MB	NB	P09
vrpnc1	524.61	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vrpnc2	835.26	0.00	0.01	4.23	0.40	0.00	0.43	0.40	0.00	0.00	0.00
vrpnc3	826.14	0.00	0.00	6.45	0.29	0.00	0.40	0.62	0.00	0.00	0.00
vrpnc4	1028.42	0.00	0.26	11.57	0.47	0.20	0.62	1.78	0.00	0.00	0.10
vrpnc5	1291.29	0.58	1.55	14.10	2.09	0.39	2.83	2.15	0.00	0.01	0.22
vrpnc6	555.43	0.00	0.00	1.35	0.00	0.00	0.00			0.00	0.00
vrpnc7	909.68	0.00	0.00	4.23	1.21	0.00	0.00			0.00	0.00
vrpnc8	865.94	0.00	0.00	2.34	0.41	0.00	0.20			0.00	0.00
vrpnc9	1162.55	0.00	0.03	3.39	0.91	0.00	0.32			0.00	0.00
vrpnc10	1395.85	0.15	0.64	7.80	2.86	0.49	2.11			0.00	0.40
vrpnc11	1042.11	0.00	0.00	2.91	0.07	0.00	0.46	0.00	0.00	0.00	0.00
vrpnc12	819.56	0.00	0.00	0.05	0.00	0.00	0.00		0.00	0.00	0.00
vrpnc13	1541.14	0.00	0.31	3.20	0.28	0.11	0.34			0.00	0.28
vrpnc14	866.35	0.00	0.00	0.41	0.00	0.00	0.09			0.00	0.00
Average		0.05	0.20	4.43	0.64	0.09	0.56	0.83	0.00	0.00	0.07
St. Dev.		0.16	0.43	4.24	0.87	0.16	0.85	0.92	0.00	0.00	0.13

Author: BA: Baker and Ayechew [11]; MM: Morgan and Mumford [176]; P09: Prins [196]; BHS: Bullnheimer et al. [31]; NB: Nagata and Bräysy [179]; T: Taillard [225]; GHL: Gendreau et al. [106]; P04: Prins [195]; TV: Toth and Vigo [239]; MB: Mester and Bräysy [170]

Table 3.3: Best results from previous studies, and their percentage difference with the best-known results, for the Christofides et al.'s benchmark set.

Instance	Best	RT	GTA	MB	MM	AD	NB
tai75a	1618.36			0.00	0.00	0.00	0.00
tai75b	1344.62			0.00	0.00		0.00
tai75c	1291.01			0.00	0.00	0.00	0.00
tai75d	1365.42			0.00	0.00	0.00	0.00
tai100a	2041.34	0.32	0.00	0.00	0.65	0.32	0.00
tai100b	1939.90	0.04		0.00	0.02	0.02	0.00
tai100c	1406.20	0.09	0.00	0.00	0.00	0.39	0.00
tai100d	1580.46	0.05	0.05	0.05	0.69	0.24	0.00
tai150a	3055.23	0.51		0.00		0.04	0.00
tai150b	2656.47	2.90	0.00	2.68		2.87	2.66
tai150c	2341.84	0.96		0.05		0.95	0.72
tai150d	2645.39	0.67		0.00		0.35	0.00
Average		0.69	0.01	0.23	0.17	0.47	0.28
St. Dev.		0.95	0.02	0.77	0.31	0.85	0.78

Author: AD: Alba and Dorronsoro [5]; MB: Mester and Bräysy [170]; NB: Nagata and Bräysy [179]; GTA: Gambardella et al. [97]; MM: Morgan and Mumford [176]; RT: Rochat and Taillard [204]

Table 3.4: Best results from previous studies, and their percentage difference with the best-known results, for the Rochat and Taillard's benchmark set.

Bräysy [170] (MB), and Nagata and Bräysy [179] (NB) obtained solutions which are, respectively, 0.16%, 0.23%, and 0.28% above the best-known results, while Rochat and Taillard [204] (RT) and Alba and Dorronsoro [5] (AD) found solutions with the highest results, 0.69% and 0.47% higher, respectively.

The multi-objective EA developed during this research, which will be presented in Chapter 5, was tested on these instances, thus it is going to be compared with the reviewed studies in order to know where the algorithm establishes among them.

3.2 VRP with Time Windows

The VRP with Time Windows (VRPTW) is a variant of the problem that has, additionally to the capacity constraint, restrictions on the service times. In this problem, vehicles have to arrive within the customers time windows to carry out the deliveries. Consequently, the following modified definition is required [46]:

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N\}$ of $N+1$ vertices, representing the geographical location of the depot and customers.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N\}$. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i), has a demand of goods $q_i > 0$, has a time window $[b_i, e_i]$ during which it has to be supplied and requires a service time s_i to unload goods.

Depot The special vertex v_0, located at (x_0, y_0), with $q_0 = 0$, time window $\big[0, e_0 \geq \max\, \big\{e_i + s_i + d_{i0} : i \in \{1, \ldots, N\}\big\}\big]$ and $s_0 = 0$, is the depot, from where customers are serviced and a fleet of vehicles is based.

Vehicles There is a homogeneous fleet of vehicles available to deliver goods to customers, departing from and arriving at the depot, which have capacity $Q \geq \max\{q_i : i = 1, \ldots, N\}$.

In this problem, in addition to the cost related to fuel cost, i.e. the travel distance d_{ij} defined in (3.1), there is a cost related to driver's remuneration, that is the delivery time t_{ij}. For the standard benchmark instances to be considered for this problem, it is common to assume unit velocity and direct travel, hence the times and distances are simply taken to be equal, i.e.

$$t_{ij} = d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \ . \tag{3.9}$$

For real-world problems, however, the distances d_{ij} are unlikely to be Euclidean and the travel times t_{ij} are unlikely to be simply related to the distances. The following will take care to accommodate those possibilities.

Time plays an important role in this problem, as it is not possible to supply a customer before or after its time window. Let $a(i,k)$ denote the arrival time of vehicle k at the i-th customer location and $l(i,k)$ be the time it leaves, and have vehicle k depart from the depot at time 0, i.e. $l(0,k) = 0$. The arrival time at the i-th customer in route r_k is then

$$a(i,k) = l(i-1,k) + t_{u(i-1,k)\,u(i,k)} \ . \tag{3.10}$$

If the vehicle arrives early at the i-th customer location, it will have to wait until the beginning of the i-th customer time window before the unload of goods can start. Hence, the departure time $l(i,k)$ will be the maximum of the arrival time $a(i,k)$ and window opening time $b_{u(i,k)}$, plus the unloading time $s_{u(i,k)}$. Consequently, the waiting time $w(i,k)$ associated with serving the i-th customer in route r_k will be

$$w(i,k) = \begin{cases} 0 & \text{if } a(i,k) \geq b_{u(i,k)} \ , \\ b_{u(i,k)} - a(i,k) & \text{otherwise} \ . \end{cases} \tag{3.11}$$

Arriving after the end of the customer time window is simply not allowed. Thus, the time vehicle k leaves the i-th customer can be written as

$$l(i,k) = a(i,k) + w(i,k) + s_{u(i,k)} \ , \tag{3.12}$$

and the total time required to complete route r_k is

$$t(r_k) = \sum_{i\,=\,0}^{N_k} \left(t_{u(i,k)\ u(i+1,k)} + w(i+1,k) + s_{u(i+1,k)} \right) \ , \tag{3.13}$$

i.e. the arrival time $a(N_k+1,k)$ of vehicle k back at the depot.

Hence, in addition to the number of routes, $f_1(\mathcal{R})$ in (3.4), and the travel distance, $f_2(\mathcal{R})$ in (3.5), a further objective function is identified, which is the total delivery time, i.e.

$$f_3(\mathcal{R}) = \sum_{r_k \in \mathcal{R}} t(r_k) = \sum_{r_k \in \mathcal{R}} \sum_{i\,=\,0}^{N_k} \left(t_{u(i,k)\ u(i+1,k)} + w(i+1,k) + s_{u(i+1,k)} \right) \ , \tag{3.14}$$

and, additionally to the capacity restriction defined in (3.6), this problem considers a supplementary constraint, which is that vehicle k must arrive at the i-th customer location no later than its time window ends, that is

$$a(i,k) \leq e_{u(i,k)}, \quad i = 1, \ldots, N_k \quad \forall\, r_k \in \mathcal{R} \ . \tag{3.15}$$

The Evolutionary Algorithms developed in this research, which will be described in Chapters 4 and 5, were tested on instances of this problem, considering the simultaneous minimisation of the number of routes, $f_1(\mathcal{R})$ in (3.4), the total travel distance, $f_2(\mathcal{R})$ in (3.5), and the total delivery time, $f_3(\mathcal{R})$ in (3.14), considering all objective functions f_i equally important, subject to the capacity constraint (3.6) and the time window constraint (3.15).

3.2.1 Solomon's benchmark set of the VRP with Time Windows

One of the standard benchmark sets, and actually most widely-used, for the VRPTW is that of Solomon [218][3], which includes 56 instances of each size $N = 25, 50, 100$.

[3]Available from Solomon's web site: `http://w.cba.neu.edu/~msolomon/problems.htm`.

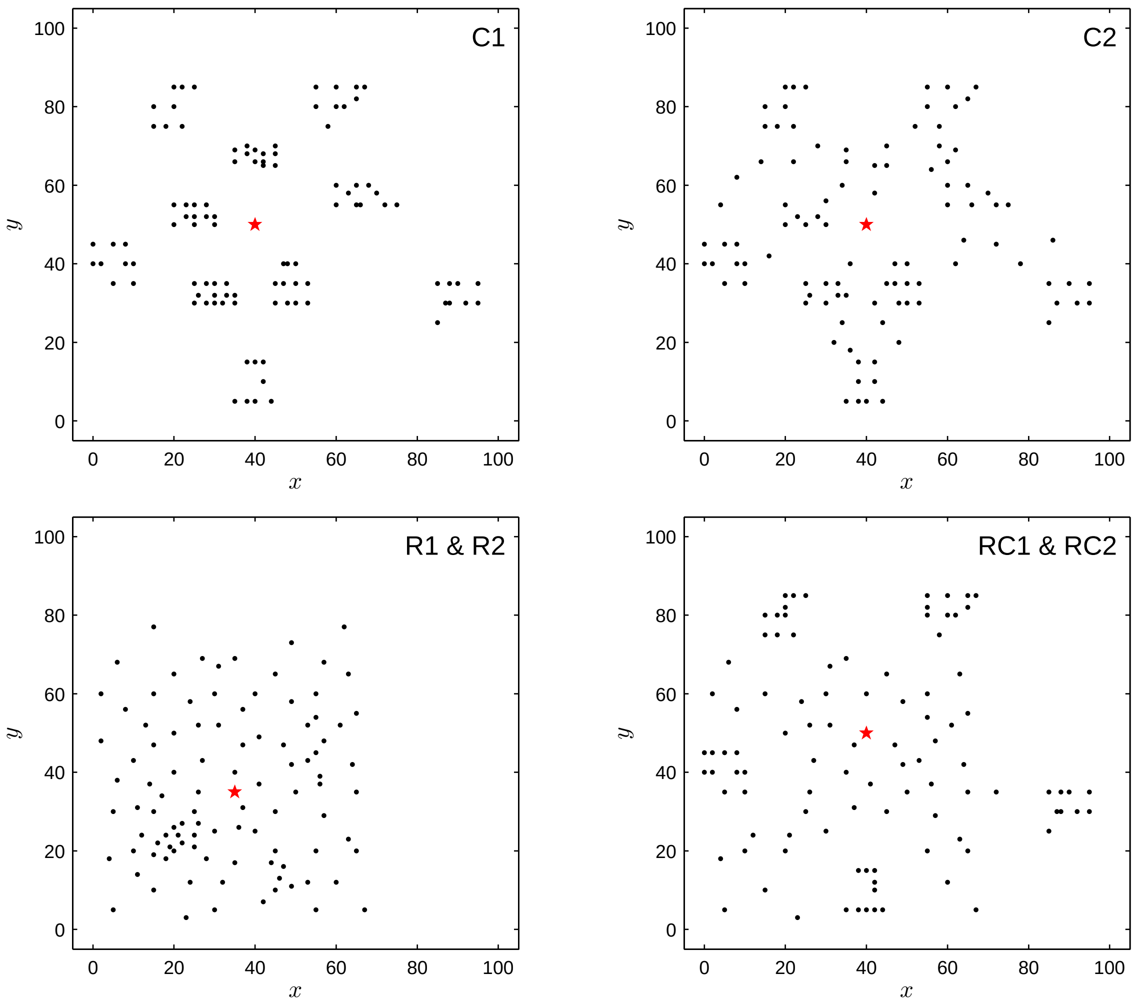


Figure 3.4: Geographical location of customers for each instance category in the Solomon's benchmark set. Location with a star represents the depot.

These instances are categorised as sets C1 (9 instances) and C2 (8 instances), where customers are located in geographical clusters, R1 (12 instances) and R2 (11 instances), where customers are randomly distributed, and RC1 (8 instances) and RC2 (8 instances), which have a mix of random locations and clusters. Figure 3.4 shows the geographical location of customers for each instance category.

Moreover, instances in sets C1, R1 and RC1 have a short scheduling horizon, i.e. the time constraint acts as a capacity constraint, which allows only a few customers to be serviced by the same vehicle. In contrast, instances in sets C2, R2 and RC2 have a long scheduling horizon, which, coupled with large vehicle capacities, permits

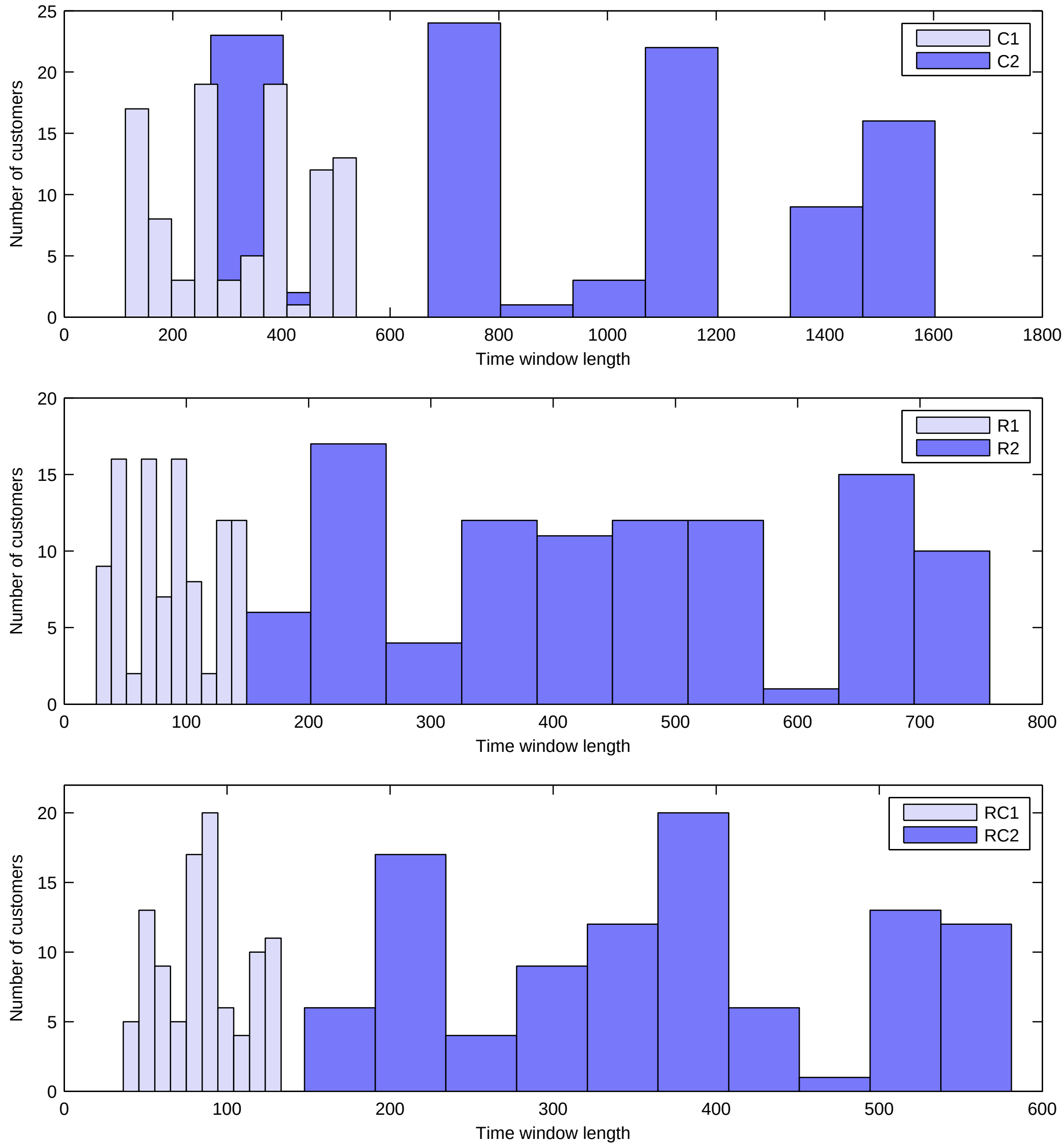


Figure 3.5: Histograms showing the number of customers and their corresponding average time window length: Customers in categories C1, R1, and RC1 have to be scheduled for service in a short period of time. On the contrary, customers in sets C2, R2, and RC2 can be scheduled to be visited in a longer period of time.

many customers to be serviced by the same vehicle [218]. The instances scheduling horizon is presented as histograms in Figure 3.5. These plots show the number of customers and their corresponding average time window length over the instances in each set category. For example, in the top plot, corresponding to the categories C1 and C2, we see that nearly 20 customers in the category C1 (lighter bars) have a time window length of approximately 300 units and that of another 20 customers is of 400 units. Nearly 13 customers have the largest time window, which length is approximately 500 units. On the other hand, for instances in category C2 (darker bars), we see that the majority of the customers have a time window which length is of approximately 700 units and longer. This means that customers in category C1 have to be scheduled for service in a short period of time and they only have a small fraction of this time to be appropriately programmed for their visit. On the contrary, customers in category C2 are scheduled to be visited in a longer period of time and the length of their time windows is also longer. For categories R1 and R2, and RC1 and RC2 we observe a similar pattern.

Due to customers have longer time windows, there are more possibilities to schedule a customer in categories C2, R2, and RC2 than in categories C1, R1, and RC1. Consequently, by the nature of the instance data, the former categories have a wider feasible region than the latter. Categories C1 and C2 are special cases since, due to the fact that customers are located in geographical clusters, the probability of finding good solutions comprising routes servicing customers from different clusters is low, although they are feasible. In fact, this aspect will reduce the region of the search space where the best solutions are and limit the degree to which the diversity enhancement mechanisms can provide improved performance.

Although the main objective of this problem is to minimise the number of routes and then, for a given number of routes, to minimise the travel distance, many studies have focused exclusively in the latter, leading to obtain solutions with an increased number of routes. For this reason, the comparison of results to these instances could be misleading if this is not taken into consideration. Table 3.5 shows the best-known result for each individual instance, regarding both the number of routes (min R) and the travel distance (min D) as the main objective to be minimised.

Short scheduling horizon							Long scheduling horizon						
Inst.	min R			min D			Inst.	min R			min D		
	R	D	Aut.	R	D	Aut.		R	D	Aut.	R	D	Aut.
C101	10	828.94	RT	10	828.94	RT	C201	3	591.56	RT	3	591.56	RT
C102	10	828.94	RT	10	828.94	RT	C202	3	591.56	RT	3	591.56	RT
C103	10	828.06	RT	10	828.06	RT	C203	3	591.17	RT	3	591.17	RT
C104	10	824.78	RT	10	824.78	RT	C204	3	590.60	RT	3	590.6	RT
C105	10	828.94	RT	10	828.94	RT	C205	3	588.88	RT	3	588.88	RT
C106	10	828.94	RT	10	828.94	RT	C206	3	588.49	RT	3	588.49	RT
C107	10	828.94	RT	10	828.94	RT	C207	3	588.29	RT	3	588.29	RT
C108	10	828.94	RT	10	828.94	RT	C208	3	588.32	RT	3	588.32	RT
C109	10	828.94	RT	10	828.94	RT							
R101	18	1613.59	TCL	18	1613.59	TCL	R201	4	1252.37	HG	9	1144.48	AMT
R102	17	1486.12	RT	18	1454.68	TCL	R202	3	1191.70	RGP	8	1034.35	JM
R103	13	1292.68	LL	14	1213.62	RT	R203	3	939.54	M	6	874.87	JM
R104	9	1007.24	M	10	974.24	TCL	R204	2	825.52	BVH	4	736.52	JM
R105	14	1377.11	RT	15	1360.78	JM	R205	3	994.42	RGP	5	954.16	ORH
R106	12	1251.98	M	13	1240.47	JM	R206	3	906.14	SSS	5	879.89	JM
R107	10	1104.66	S97	11	1073.34	JM	R207	2	837.20	BC	4	799.86	JM
R108	9	960.88	BBB	10	947.55	JM	R208	2	726.75	M	4	705.45	JM
R109	11	1194.73	HG	13	1151.84	JM	R209	3	909.16	H	5	859.39	JM
R110	10	1118.59	M	12	1072.41	JM	R210	3	939.34	M	5	910.7	JM
R111	10	1096.72	RGP	12	1053.5	JM	R211	2	892.71	BVH	4	755.96	JM
R112	9	982.14	GTA	10	953.63	RT							
RC101	14	1696.94	TBG	15	1623.58	RT	RC201	4	1406.91	M	6	1134.91	TCL
RC102	12	1554.75	TBG	14	1461.23	JM	RC202	3	1365.65	RTI	8	1095.64	JM
RC103	11	1261.67	S98	11	1261.67	S98	RC203	3	1049.62	CC	5	928.51	JM
RC104	10	1135.48	CLM	10	1135.48	CLM	RC204	3	798.41	M	4	786.38	JM
RC105	13	1629.44	BBB	16	1518.58	JM	RC205	4	1297.19	M	7	1157.55	JM
RC106	11	1424.73	BBB	13	1371.69	TCL	RC206	3	1146.32	H	7	1054.61	JM
RC107	11	1222.16	TCL	12	1212.83	JM	RC207	3	1061.14	BVH	6	966.08	JM
RC108	10	1139.82	TBG	11	1117.53	JM	RC208	3	828.14	IKM	4	779.31	JM

Author: AMT: Alvarenga et al. [7]
BBB: Berger et al. [20]
BC: Le Bouthillier and Crainic [158]
BVH: Bent and Van Hentenryck [17]
CC: Czech and Czarnas [52]
CLM: Cordeau et al. [47]
GTA: Gambardella et al. [97]
H: Homberger [127]
HG: Homberger and Gehring [128]
IKM: Ibaraki et al. [130]
JM: Jung and Moon [137]
LL: Li and Lim [161]
M: Mester [169]
RGP: Rousseau et al. [207]
RT: Rochat and Taillard [204]
RTI: Repoussis et al. [203]
ORH: Ombuki et al. [182]
S97: Shaw [215]
S98: Shaw [216]
SSSD: Schrimpf et al. [213]
TBG: Taillard et al. [226]
TCL: Tan et al. [228]

Table 3.5: Best-known results, regarding the solutions with the smallest number of routes and with the shortest travel distance, for the Solomon's benchmark set.

Analysing the figures shown in this table, we realise that, with the exception of all instances in categories C1 and C2, and instances R101, RC103, and RC104, all other instances have two different solutions: one with the shortest number of routes and the other with the shortest travel distance. Therefore, it is clear that these instances have at least two solutions, thus this benchmark set, specially instances in categories R1, R2, RC1, and RC2, might potentially be considered for multi-objective optimisation.

3.2.2 Solomon's I1 insertion heuristic

Solomon [218] proposed several construction heuristics: two savings heuristics, a time-oriented, nearest-neighbour heuristic, three insertion heuristics, and a time-oriented sweep heuristic. Among the insertion heuristics, the *I1* is considered to be the most successful [27].

A route is first initialised with a *seed* customer, which is selected by finding either the geographically farthest unrouted customer in relation to the depot or the unrouted customer with the lowest allowed starting time for service. The method uses two criteria, $c_l(v_i, v_j, v_{i+1})$ and $c_2(v_i, v_j, v_{i+1})$, at every iteration to insert a new customer v_j into the current partial route, between two adjacent customers v_i and v_{i+1}.

Let $r_k = \langle v_1, \ldots, v_{N_k} \rangle$ be the current route. For each unserviced customer, we first compute its best feasible insertion place in the emerging route as

$$c_1\big(v_i(v_j), v_j, v_{i+1}(v_j)\big) = \min_{p\,=\,1,\ldots,N_k+1} c_l(v_{p-1}, v_j, v_p) \ . \tag{3.16}$$

It is important to mention that inserting v_j between v_{p-1} and v_p could potentially alter the times to begin service at all customers after v_{p-1}, that is $\langle\langle v_p, \ldots, v_{N_k} \rangle\rangle$. Actually, Dullaert and Bräysy [80] argue that, if customer v_j is inserted between v_0 and v_1 in the partially constructed route, the additional time needed is underestimated. This can cause the selection of sub-optimum insertion places for unrouted customers. Thus, a route with a relatively small number of customers can have a larger schedule time than necessary.

Next, the best customer v_j^* to be inserted in the route is selected as the one for which

$$c_2\big(v_i(v_j^*), v_j^*, v_{i+1}(v_j^*)\big) = \max_{v_j \in \mathcal{U}} \big\{c_2\big(v_i(v_j), v_j, v_{i+1}(v_j)\big)\big\} \ , \tag{3.17}$$

where set $\mathcal{U}$ contains all unserviced customers which insertion will not produce an infeasible solution.

Customer v_j^* is then inserted in the route between $v_i(v_j^*)$ and $v_{i+1}(v_j^*)$. When no more customers with feasible insertions can be found, the method starts a new route, unless it has already routed all customers.

The best feasible insertion place for an unrouted customer is the one that minimises the distance and time required to visit the customer, criterion $c_l(v_i, v_j, v_{i+1})$, and maximises the benefit derived from servicing a customer on the partial route being constructed rather than on a direct route, criterion $c_2(v_i, v_j, v_{i+1})$.

3.2.3 Overview of metaheuristic approaches

A significant percentage of the research on VRP has been focused on the variant with Time Windows. This section provides a review of some studies that have achieved important results and are of concern to this thesis.

Many Tabu Search (TS) methods have been also applied to this problem. For instance, the TS of Potvin et al. [194] was inspired by the approach of Gendreau et al. [106] for solving the CVRP. However, solution feasibility is always maintained because, due to the time window constraint, it is quite difficult to get back to a feasible solution after an infeasible move. This TS is initialised with the solution produced by the I1 heuristic. Afterwards, the 2-opt and Or-opt local search are alternated to define neighbourhoods and perform moves. However, the entire neighbourhoods are not explored, instead they are reduced by selecting a subset of customers at each iteration and by considering only the exchanges that link the selected customers with customers that are close in distance.

Taillard et al. [226] based their TS on the strategy of Rochat and Taillard [204] for solving the CVRP, in the sense that it uses an adaptive memory containing routes from good solutions and performs the GENIUS heuristic of Gendreau et al. [106] as a post-optimisation procedure. The adaptive memory is initialised with routes from solutions obtained with the I1 heuristic. This approach defines the *CROSS exchange* local search, which is used to build the neighbourhood. Dynamic diversification is incorporated by penalising CROSS exchanges that are frequently performed during the search. The customers within each individual route are reordered, by means of the I1 heuristic, when an overall best solution is found.

Cordeau et al. [47] proposed a method in which initial solutions are constructed by means of a kind of sweep heuristic, i.e. according to the angle the customers make with the depot. This algorithm allows violation of constraints, though they are penalised in the cost function which weighting coefficients are adjusted dynamically. The best feasible solution identified during the search is finally post-optimised by applying an adaptation of the GENIUS heuristic of Gendreau et al. [106].

As well as for CVRP, a number of evolutionary and hybrid approaches have been proposed to tackle VRPTW, the main difference between them being the recombination and mutation operators, and the additional strategies incorporated in the algorithms. For example, Potvin and Bengio [192] developed the *GENEtic ROUting System* (GENEROUS). They proposed the recombination operators *sequence-based crossover* (SBX) and *route-based crossover* (RBX), and the mutation operators *one-level exchange*, *two-level exchange*, and one based on the Or-opt local search.

Something different was proposed by Jung and Moon [137], who encoded a solution in a two-dimensional image and introduced the *natural crossover*, which partitions the image by drawing one or more curves or geometric figures. Routes from both parents that have not been broken with this partition are copied into the offspring and a repair mechanism is applied. The mutation calls three local search heuristics in sequence, namely Or-opt, and the crossover and relocation of Savelsbergh [210].

Zhu [256] presented a Genetic Algorithm that adapts the recombination and mutation rates to the population dynamics, maintaining population diversity at user-defined levels, thus preventing premature convergence. The initial population is created with a feasible solution from the I1 heuristic and some of its 2-opt neighbours, and with randomly generated solutions. Two recombination operators are considered, namely partially matched crossover [120] and order crossover [181], each of them with the same probability of being applied. They introduced the three mutation operators *one-step route reduction*, *one-step cost reduction*, and *gene relocation*, from which one is applied.

The hybrid approach of Berger et al. [20] considers two populations Pop_1 and Pop_2, primarily formed of non-feasible solution individuals, which are evolving concurrently, each with its own objective function. Pop_1 contains at least one feasible solution and is used to minimise the total travel distance, while Pop_2 focuses on minimising constraint violation. They introduced the two recombination operators *insertion-based crossover* and *insertion within route-based crossover*, and a suite of five mutation operators, namely *large neighbourhood search-based*, *edge exchange*, *repair solution*, *reinsert shortest route*, and *reorder customers*. Later, Berger and Barkaoui [19] proposed a parallel version of this approach: the master component controls the execution of the algorithm, coordinates evolutionary operators and handles parent selection, while the slave element concurrently executes reproduction and mutation operators.

Le Bouthillier and Crainic [158] proposed a cooperative parallel metaheuristic that implements two TS algorithms, that is the one by Gendreau et al. [106] and that of Cordeau et al. [47], and two Evolutionary Algorithms (EA), which difference lies in the recombination operator used, either order crossover [181] or the edge recombination crossover of Whitley et al. [250]. To generate initial solutions and help diversify the pool, four simple construction heuristics from Bentley [18] were adapted: *least successor*, *double-ended nearest neighbour*, *multiple fragments*, and *shortest arcs hybridizing*. Each of these metaheuristics works independently from each other and the best solutions found are collected in the solution warehouse, which is divided into two sub-populations: *in-training* and *adult*. All solutions received from the independent processes are placed in the in-training part. Then, one of the post-optimisation local search procedures 2-opt, 3-opt, and Or-opt, is applied and the resulting solution is moved to the adult sub-population.

Homberger and Gehring [129] proposed a two-phase metaheuristic that combines a $(\mu + \lambda)$-Evolution Strategy (ES), where μ denotes the parent population size and λ the offspring population. and a subsequent TS. It starts with an initial population P of μ different feasible solutions generated using a modified version of the savings heuristic. Then, the ES is executed by selecting a random solution from P which is submitted to one of the local search methods 2-opt, Or-opt, and λ-interchange. The best solution found in this stage is submitted to the TS, which considers restricted neighbourhoods of the same local search heuristics used in the first stage. The authors also designed large scale instances, ranging from 200 to 1000 customers.

Alvarenga et al. [7] introduced a Genetic Algorithm and a set partitioning formulation of the problem. The authors proposed the *stochastic PFIH*, which is a stochastic version of the I1 heuristic, for the construction of initial solutions. The recombination randomly chooses a route from each parent in turns and, after all feasible routes have been inserted in the offspring, the insertion of remaining customers are tested in existing routes. If some customers continue to be unrouted, new routes are created. In this step, the stochastic PFIH is again applied. A total of eight different specialised operators were used in the mutation phase, namely *random customer migration*, *bringing the best customer*, *re-insertion using stochastic PFIH*, *similar customer exchange*, *customer exchange with positive gain*, *merge two routes*, *reinserting customer*, and *route partitioning*.

Repoussis et al. [203] designed a EA which uses the *Greedy Randomised Construction*, a modified variant of the probabilistic parallel construction heuristic of Kontoravdis and Bard [148], for generating initial solutions. This EA considers a multi-parent recombination operator, which goal is to deterministically select and extract promising solution arcs from each parent individual. The proposed mutation operator follows the ruin-and-recreate principle, in which a selected set of customers is removed from the current solution and they are reinserted back to the partially ruined solution in a probabilistic-heuristic fashion. This EA considers two additional TS-based processes at every generation after mutation: one to reduced the number of routes and the other to minimise the travel distance.

Other studies have considered different metaheuristics, e.g. Gambardella et al. [97] designed the *Multiple Ant Colony System* (MACS), which considers two objective functions: the minimisation of the number of routes, and the minimisation of the total travel time, where the former takes precedence over the latter. In MACS both objectives are optimised simultaneously by coordinating the activities of two colonies. The goal of the first, *VEI*, is to try to decrease the number of vehicles used, while the second colony, *TIME*, optimises the feasible solutions found by VEI. Furthermore, TIME implements a local search procedure similar to the CROSS exchange of Taillard et al. [226] to improve the quality of feasible solutions.

Finally, *Alternate K-exchange Reduction* (AKRed), which is a two-phase approximation algorithm, was proposed by Cordone and Wolfler Calvo [50]. It builds a set of initial solutions using the heuristics introduced by Solomon [218], improves them through a local search procedure and returns the best solution found. AKRed's local search procedure combines k-opt moves, specifically 2-opt and 3-opt moves between routes and inside a single route, and an ad hoc procedure, called *Reduction*, to reduce the number of vehicles. Both take advantage of using a global best strategy and are nested in such a way that their interaction is as effective as possible. To escape from sub-optimum solutions, the algorithm applies the same local search procedure with a slightly different objective function, whose main component is still the number of vehicles, whereas the secondary component is the route duration, instead of the travel distance.

3.2.4 Results from previous studies

The vast majority of the reviewed studies have tested their approaches on the Solomon's benchmark set. Due to this data set contains 56 instances, it would be impractical to show, for all of them, the best result found by each individual approach. Additionally, not all of the studies made available the detail of their results, instead, the best results have been traditionally presented averaged over the instances in each of the C1, C2, R1, R2, RC1, and RC2 categories. Furthermore, as was stated earlier (Section 3.2.1), these benchmark instances have been solved considering either the number of routes (R) or the travel distance (D) as the main objective to be optimised, thus the comparison of studies which prioritised different objectives could result in large differences in the objectives.

For the reasons above, results from previous studies are presented in two ways. The first considers the difference between their best results and the best-known results, regarding both the solution with the smallest number of routes and the solution

Author		C1	C2	R1	R2	RC1	RC2	Total
Rochat and Taillard [204]		0.00	0.00	3.79	7.58	3.07	4.17	3.33
		0.00	0.00	0.08	1.80	-0.28	0.38	0.38
Taillard et al. [226]		0.00	0.00	3.15	4.55	0.00	4.17	2.16
		0.00	0.00	0.05	3.74	0.43	0.46	0.87
Cordeau et al. [47]		0.00	0.00	2.31	0.00	0.00	0.00	0.50
		0.00	0.00	0.13	2.61	0.47	1.29	0.79
Cordone and Wolfler Calvo [50]		0.00	0.00	6.31	9.09	7.67	4.17	4.83
		0.69	0.32	3.05	5.18	2.28	2.47	2.50
Jung and Moon [137]		0.00	0.00	12.73	96.21	12.55	91.67	36.52
		0.00	0.00	-2.43	-7.01	-2.55	-9.57	-3.63
Le Bouthillier and Crainic [158]		0.00	0.00	2.31	0.00	0.00	0.00	0.50
		0.00	0.00	0.05	1.57	0.23	1.15	0.51
Alvarenga et al. [7]		0.00	0.00	12.73	102.27	11.41	100.00	38.73
		0.00	0.00	-2.10	-4.59	-2.70	-8.45	-2.94
Repoussis et al. [203]		0.00	0.00	0.46	0.00	0.00	0.00	0.10
		0.00	0.00	0.24	0.70	0.10	0.04	0.21

Table 3.6: Average difference, grouped by instance category, between results from previous studies and the best-known results, considering the solutions with the smallest number of routes to the Solomon's benchmark set.

with the shortest travel distance presented in Table 3.5. For each author and test instance, the solution with the smallest number of routes and that with the shortest travel distance were taken, when available, and the per cent difference in both objectives between these solutions and the best-known was computed. Then, the average of this difference over the instances in each category was calculated. Table 3.6 shows, for each author (row) and instance category (column), the percentage difference in the number of routes (upper figure) and in the travel distance (lower figure) regarding solutions with the smallest number of routes. The last column presents the total average difference for all 56 instances. A negative difference corresponds to an improvement in the best-known result. Table 3.7 presents the corresponding average per cent difference considering the solutions with the shortest travel distance.

In Table 3.6 we can observe that the approach of Repoussis et al. [203] found solutions with the smallest difference in the number of routes (0.1%) between their

Author	C1	C2	R1	R2	RC1	RC2	Total
Rochat and Taillard [204]	0.00	0.00	-6.15	-44.60	-6.38	-40.42	-16.76
	0.00	0.00	2.85	9.70	2.82	12.70	4.73
Taillard et al. [226]	0.00	0.00	-6.75	-45.73	-9.06	-40.42	-17.50
	0.00	0.00	2.85	11.84	3.55	12.73	5.26
Cordeau et al. [47]	0.00	0.00	-7.44	-48.01	-9.06	-41.98	-18.32
	0.00	0.00	2.93	10.59	3.59	14.00	5.22
Cordone and Wolfler Calvo [50]	0.00	0.00	-3.98	-43.46	-2.17	-40.42	-15.47
	0.69	0.32	5.92	13.20	5.37	14.98	6.93
Jung and Moon [137]	0.00	0.00	1.76	0.00	1.97	6.25	1.55
	0.00	0.00	0.28	0.02	0.41	1.44	0.33
Le Bouthillier and Crainic [158]	0.00	0.00	-7.44	-48.01	-9.06	-41.98	-18.32
	0.00	0.00	2.84	9.48	3.34	13.85	4.93
Alvarenga et al. [7]	0.00	0.00	1.76	4.62	0.83	11.46	3.04
	0.00	0.00	0.61	2.63	0.26	2.69	1.07
Repoussis et al. [203]	0.00	0.00	-9.11	-48.01	-9.06	-41.98	-18.67
	0.00	0.00	3.05	8.51	3.20	12.57	4.58

Table 3.7: Average difference, grouped by instance category, between results from previous studies and the best-known results, considering the solutions with the shortest travel distance to the Solomon's benchmark set.

results and the best-known, followed by the approach of Cordeau et al. [47] and that of Le Bouthillier and Crainic [158], which found solutions with a difference of 0.5%, however, the latter achieved solutions with a smaller difference in travel distance (0.51%), while the study of Jung and Moon [137] obtained solutions with the highest difference (36.52%).

On the other hand, regarding solutions with the shortest travel distance, in Table 3.7 we see that the approach of Jung and Moon [137], which obtained the largest difference in the number of routes, achieved the smallest difference between their results and the best-known (0.33%), followed by that of Alvarenga et al. [7] which achieved solutions with a difference of 1.07%, while the largest difference was due to Cordone and Wolfler Calvo [50] (6.93%). It is clear from both tables that all approaches found the best-known solutions for categories C1 and C2, since the difference in both objectives is zero, except for that of Cordone and Wolfler Calvo [50].

Author	C1	C2	R1	R2	RC1	RC2	Total
RT	10.00	3.00	12.58	3.09	12.38	3.62	415.00
	828.45	590.32	1197.42	954.36	1369.48	1139.79	57231.05
PB	10.00	3.00	12.60	3.00	12.10	3.40	
	838.00	589.90	1296.80	1117.70	1446.20	1360.60	
PKG	10.00	3.00	12.50	3.10	12.60	3.40	
	850.20	594.60	1294.50	1154.40	1456.30	1404.80	
TBG	10.00	3.00	12.17	2.82	11.50	3.38	410.00
	828.38	589.86	1209.35	980.27	1389.22	1117.44	57521.79
GTA	10.00	3.00	12.00	2.73	11.63	3.25	
	828.38	589.86	1217.73	967.75	1382.42	1129.19	
CLM	10.00	3.00	12.08	2.73	11.50	3.25	407.00
	828.38	589.86	1210.14	969.57	1389.78	1134.52	57555.63
CW	10.00	3.00	12.50	2.91	12.38	3.38	422.00
	834.05	591.78	1241.89	995.39	1408.87	1139.70	58481.21
JM	10.00	3.00	13.25	5.36	13.00	6.25	486.00
	828.38	589.86	1179.95	878.41	1343.64	1004.20	54779.02
BBB	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.48	589.93	1121.10	975.73	1389.89	1159.37	57952.00

Author: BBB: Berger et al. [20] GTA: Gambardella et al. [97] PKG: Potvin and Bengio [192]
CLM: Cordeau et al. [47] JM: Jung and Moon [137] RT: Rochat and Taillard [204]
CW: Cordone and Wolfler Calvo [50] PB: Potvin et al. [194] TBG: Taillard et al. [226]

Table 3.8: Average best results from previous studies, grouped by instance category, for the Solomon's benchmark set.

The second way of presenting previous results is the traditional style commonly found in the literature. Tables 3.8 and 3.9 show, for each author (row) and instance category (column), the average number of routes (upper figure) and the average travel distance (lower figure). Additionally, the last column presents, when available, the total number of routes and total travel distance for all 56 instances. We can observe again that many approaches found the best-known solutions for instances in categories C1 and C2, in which customers are clustered. The approach of Homberger and Gehring [129] (HG) obtained solutions with the smallest average number of routes for categories R1 and RC2, and in total, while the algorithm of Repoussis et al. [203] for categories R2 and RC1. On the other hand, the method of Jung and Moon [137] found solutions with the shortest average travel distance for categories

Author	C1	C2	R1	R2	RC1	RC2	Total
Z	10.00	3.00	12.80	3.00	13.00	3.70	
	828.90	589.90	1242.70	1016.40	1412.00	1201.20	
BB	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.48	589.93	1221.10	975.43	1389.89	1159.37	57952.00
HG	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.86	1212.73	955.03	1386.44	1108.52	57192.00
BC	10.00	3.00	12.08	2.73	11.50	3.25	407.00
	828.38	589.86	1209.19	960.95	1386.38	1133.30	57412.00
MBD	10.00	3.00	12.00	2.73	11.50	3.25	406.00
	828.38	589.86	1208.18	954.09	1387.12	1119.70	56812.00
PR	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.38	589.86	1212.39	957.72	1385.78	1123.49	57332.00
AMT	10.00	3.00	13.25	5.55	12.88	6.50	489.00
	828.38	589.86	1183.38	899.90	1341.67	1015.90	55134.27
RTI	10.00	3.00	11.92	2.73	11.50	3.25	405.00
	828.38	589.86	1210.82	952.67	1384.30	1119.72	57215.65

Author: AMT: Alvarenga et al. [7] HG: Homberger and Gehring [129] RTI: Repoussis et al. [203]
BB: Berger and Barkaoui [19] MBD: Mester et al. [171] Z: Zhu [256]
BC: Le Bouthillier and Crainic [158] PR: Pisinger and Ropke [189]

Table 3.9: Average best results from previous studies, grouped by instance category, for the Solomon's benchmark set.

R2 and RC2, and in total, while that of Berger et al. [20] for category R1 and that of Alvarenga et al. [7] for category RC1.

The summary of these best average results is shown in Table 3.10, which presents two main rows corresponding to the results with the smallest number of routes ($\min R$) and with the shortest travel distance ($\min D$). For each row and set category (column), the average number of routes (upper figure) and the average travel distance (lower figure) are presented, along with the author who obtained those results. These results are going to be considered in Chapters 4 and 5 in order to compare the results from the approaches designed in this research and consequently know where the proposed algorithms establish among the previous studies regarding performance. However, the detailed results presented in Tables 3.8 and 3.9 are also going to be referenced for comparison purposes.

Objective	C1	C2	R1	R2	RC1	RC2	Total
min R	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.86	1212.73	952.67	1384.30	1108.52	57192.00
Author			HG	RTI	RTI	HG	HG
min D	10.00	3.00	11.92	5.36	12.88	6.25	486.00
	828.38	589.86	1121.10	878.41	1341.67	1004.20	54779.02
Author			BBB	JM	AMT	JM	JM

Author: AMT: Alvarenga et al. [7] HG: Homberger and Gehring [129] RTI: Repoussis et al. [203]
BBB: Berger et al. [20] JM: Jung and Moon [137]

Table 3.10: Best-known average results, regarding solutions with the lowest number of routes and the shortest travel distance, for the Solomon's benchmark set grouped by categories.

3.3 Other Vehicle Routing Problems

As we have reviewed in the previous two sections, the VRP variants approximate better to practical problems due to the restrictions that are considered. There exist many other variants of the VRP [235, 121, 188], with differences between them in the operational and customer service restrictions. The next sections describe other VRPs that are commonly found in the literature.

It is important to remark that although the developed final algorithm, which will be presented in Chapter 5, was only tested on the CVRP and VRPTW, its current development might permit to equally well tackle other variants of the VRP, specifically those which consider capacity and, to some extent, service time constraints. This could be handled by suitably modifying the instance customer service restrictions and by appropriately setting in the algorithm the operational constraints. Thus, simulation results for other variants than CVRP and VRPTW will not be presented, instead, this is going to be subject of subsequent research.

3.3.1 VRP with Backhauls

The VRP with Backhauls (VRPB) is an extension of the CVRP where the customers are grouped into *linehaul* customers, which have a demand of goods, and *backhaul* customers, from which a quantity of the product has to be collected. A practical example of this customer partition is that of the grocery industry, where supermarkets and shops are the linehaul customers and grocery suppliers are the backhaul customers [237].

An instance of the VRPB has the following definitions [233]:

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N, v_{N+1}, \ldots, v_{N+M}\}$ of $N+M+1$ vertices, representing the geographical location of the depot and customers.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N, v_{N+1}, \ldots, v_{N+M}\}$. Furthermore, subset $\mathcal{V}_L = \{v_1, \ldots, v_N\}$ corresponds to linehaul customers and subset $\mathcal{V}_B = \{v_{N+1}, \ldots, v_{N+M}\}$ represents the backhaul customers. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i) and has a demand of goods $q_i > 0$ to be delivered or collected.

Depot The special vertex v_0 located at (x_0, y_0), with $q_0 = 0$, is the depot, from where customers are serviced and a fleet of vehicles is based.

Vehicles There is a homogeneous fleet of vehicles available to deliver goods to customers, departing from and arriving at the depot, and having capacity $Q \geq \max\{q_i : i = 1, \ldots, N, N+1, \ldots, N+M\}$.

The travel from vertex v_i to vertex v_j has an associated cost c_{ij}, that is, there is a matrix $C = (c_{ij})$ corresponding to travel costs. In VRPB one aims at minimising, first, the number of routes, then, for a given number of routes, the total cost, subject to the following limitations [79]:

- Each vehicle services exactly one route.
- Each customer is visited exactly once by one vehicle.
- A route is not allowed to consist entirely of linehaul or backhaul customers.
- If a route contains both linehaul and backhaul customers, then the backhaul customers must be served after the linehaul customers.
- For each route, the total load associated with linehaul and backhaul customers must not exceed the vehicle capacity Q.

The fourth constraint is justified by the fact that many vehicles are rear-loaded. This makes it problematic to try to load the vehicle with goods heading for the depot before having delivered all goods to the customers, as the collected goods might block access to the delivery goods. The constraint is also justified by the fact that the linehaul customers frequently prefer early deliveries, while backhaul customers prefer late collection [206].

3.3.2 VRP with Pickups and Deliveries

In the VRP with Pickups and Deliveries (VRPPD), a heterogeneous fleet of vehicles, based at multiple terminals, must satisfy a set of transport requests. Each request is defined by a pickup point, a corresponding delivery point, and a demand to be transported between these locations. The requested transport could involve goods or persons. This latter environment is called *dial-a-ride* [69].

The VRPPD requires the following definitions [211, 243]:

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N, v_{N+1}, \ldots, v_{2N}\}$ of $2N + 1$ vertices, representing the geographical location of the depot and customers.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N, v_{N+1}, \ldots, v_{2N}\}$. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i). Moreover, subset $\mathcal{V}_P = \{v_1, \ldots, v_N\}$ corresponds to pick up locations and subset $\mathcal{V}_D = \{v_{N+1}, \ldots, v_{2N}\}$ represents the delivery locations.

Requests The set $\mathcal{W} = \{1, \ldots, N\}$ represents N transportation requests. For each request $i \in \mathcal{W}$, a load q_i has to be transported from customer $v_i \in \mathcal{V}_P$ to customer $v_j \in \mathcal{V}_D$.

Depot The special vertex v_0, located at (x_0, y_0), is the depot, where vehicles are based.

Vehicles There is a homogeneous fleet of vehicles available to service requests, departing from and arriving at the depot, and having capacity $Q \geq \max\{q_i : i \in \mathcal{W}\}$.

The travel from vertex v_i to vertex v_j has an associated cost c_{ij}, that is, there is a matrix $C = (c_{ij})$ corresponding to travel costs. Then, the VRPPD consists of finding a set of exactly K routes with minimum cost, and such that [236]:

- Each routes visits the depot.
- Each customer is visited by exactly one vehicle.
- The load of the vehicle must be non-negative at all times and never exceed the vehicle capacity Q.
- For each request i, the associated pick up location, when different from the depot, must be served in the same route and before the corresponding delivery location.

- For each request i, the associated delivery location, when different from the depot, must be served in the same route and after the corresponding pick up location.

3.3.3 Multiple Depot VRP

Instances of the Multiple Depot VRP (MDVRP) may have more than one depot from which customers could be served. Here, customers are to be assigned to depots, where a fleet of vehicles is based. Then, each route originates from one depot, service the customers assigned to that route, and return to the same depot. The formal definition of the MDVRP is the following [202]:

Vertices There is a set $\mathcal{V} = \{v_1, \ldots, v_N, v_{N+1}, \ldots, v_{N+D}\}$ of $N + D$ vertices, representing the geographical location of customers and depots.

Customers Customers are represented by the vertices in subset $\mathcal{V}_C = \{v_1, \ldots, v_N\}$. Each customer $v_i \in \mathcal{V}_C$ is geographically located at coordinates (x_i, y_i) and has a demand of goods $q_i > 0$.

Depots The vertices in subset $\mathcal{V}_D = \{v_{N+1}, \ldots, v_{N+D}\}$ are the depots. Each depot $v_j \in \mathcal{V}_D$ is located at (x_j, y_j) and has a demand $q_j = 0$.

Vehicles There is a fleet of m_j identical vehicles of capacity $Q \geq \max\{q_i : i = 1, \ldots, N\}$ based at each depot $v_j \in \mathcal{V}_D$, from where customers may be serviced. m_j could be equal to zero, which means that not all depots are necessarily used.

The travel from vertex v_i to vertex v_j has an associated cost c_{ij}, that is, there is a matrix $C = (c_{ij})$ corresponding to travel costs. Thus, the MDVRP consists of constructing a minimum-cost set of routes in such a way that [114]:

- Each route starts and ends at the same depot.

- Each customer is visited exactly once by one vehicle.

- The total demand of each route does not exceed the vehicle capacity Q.

3.3.4 Periodic VRP

In the case of the Periodic VRP (PVRP), the basic VRP is generalised by extending the scheduling service period from 1 to M days. Thus, a solution to this problem may consider vehicles that might not return to the depot in the same day it departs. Additionally, each customer could be visited more than once over the M-day period.

The PVRP is defined as follows [92]:

Period There is a set $\mathcal{P} = \{\delta_1, \ldots, \delta_M\}$ of M days that constitute the planning period.

Schedule A schedule is a collection of days within the planning period in which customers receive service. Allocating a customer to a schedule implies that the customer will receive service in every day of that schedule.

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N\}$ of $N + 1$ vertices, representing the geographical location of the depot and customers.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N\}$. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i), has a total demand of goods $q_i > 0$ over the planning period and requires a fixed number visits ϕ_i.

Depot The special vertex v_0, located at (x_0, y_0), with $q_0 = 0$, is the depot, from where customers are serviced and a fleet of vehicles is based.

Vehicles There is a homogeneous fleet of vehicles available to deliver goods to customers, departing from and arriving at the depot, and having capacity $Q \geq \max\{q_i : i = 1, \ldots, N\}$.

The travel from vertex v_i to vertex v_j has an associated cost c_{ij}, that is, there is a matrix $C = (c_{ij})$ corresponding to travel costs. In the PVRP customers cannot be directly assigned to vehicles, instead, they must be first allocated to an schedule, thereby defining the delivery days for the customer, and then to a route on each of the chosen delivery days [227]. During day δ_m of the planning period, a feasible route is an ordered sequence of customers satisfying the following constraints [104]:

- Each route must start and end at the depot.
- The sequence of customers must be visited by the same vehicle during day δ_m.
- The travel time associated with the sequence must not exceed the daily service time of the vehicle.
- The total demand requested by the customers belonging to the route must not exceed the vehicle capacity Q.

3.3.5 Split Delivery VRP

The Split Delivery VRP (SDVRP) was introduced in the literature by Dror and Trudeaut [77], who showed that there can be savings generated by allowing split deliveries. In this context, SDVRP can be tackled as a relaxation of the VRP in the sense that customers are allowed to be served by different vehicles as long as this service plan minimises the total costs [78]. The SVRP is defined as follows [9]:

Vertices There is a set $\mathcal{V} = \{v_0, \ldots, v_N\}$ of $N + 1$ vertices, representing the geographical location of the depot and customers.

Customers Customers are represented by the vertices in subset $\mathcal{V}' = \mathcal{V} \setminus \{v_0\} = \{v_1, \ldots, v_N\}$. Each customer $v_i \in \mathcal{V}'$ is geographically located at coordinates (x_i, y_i) and has a demand of goods $q_i > 0$.

Depot The special vertex v_0, located at (x_0, y_0), with $q_0 = 0$, is the depot, from where customers are serviced and a fleet of vehicles is based.

Vehicles There is a homogeneous fleet of vehicles available to deliver goods to customers, departing from and arriving at the depot, and having capacity $Q \geq \max\{q_i : i = 1, \ldots, N\}$.

The traversal from vertex v_i to vertex v_j has a corresponding cost c_{ij}, i.e. there is a matrix $C = (c_{ij})$ associated to travel costs. The objective is to minimise the total distance travelled by the vehicles while considering the following restrictions [8]:

- Each vehicle must start and end its route at the depot.
- The demands of the customers must be satisfied.
- The quantity delivered in each route cannot exceed the vehicle capacity Q.

3.3.6 Stochastic VRP

According to Hadjiconstantinou and Roberts [124], the Stochastic VRP (SVRP) differs from the VRP by the introduction of some element of variability of the system in question. Unlike its deterministic equivalent, SVRP is ambiguously defined, since it belongs to a class of a priori optimisation problems for which it is impractical to consider an a posteriori approach that computes an optimum solution whenever the random variables are realised.

Common examples of stochastic elements are:

- Stochastic demands [21].
- Stochastic travel times [155].

Sometimes, the set of customers to be visited is not known with certainty. In such a case, each customer has a probability p_i of being present [107, 108].

3.4 Multi-objective Vehicle Routing Problems

Multi-objective Vehicle Routing Problems are mainly used in three ways [135]: (*i*) to extend classical academic problems in order to improve their practical application (while never losing sight of the initial objective), (*ii*) to generalise classic problems, and (*iii*) to study real-life cases in which the objectives have been clearly identified by the decision-maker and are dedicated to a specific practical problem or application.

In any of the contexts cited above, in addition to the intrinsic economic cost of routing, e.g. number of vehicles, travel distance and delivery time, a number of objectives has been considered to be optimised, among which are the following:

- *Service level.* This objective is related to the minimisation of the longest route length [45, 177, 185, 231].
- *Constraints.* This objective regards the minimisation of the number or the extent of violated constraints [197, 16, 34].
- *Workload imbalance.* Here one aims at minimising the difference of the workload between the longest and the shortest routes [133, 134, 187].
- *Security.* There are problems which concern that the risks of accidents is minimised [112, 168].
- *Accessibility.* This is a dual objective, which corresponds to maximising the coverage of a geographical area while minimising the number of mobile facilities [75, 74].
- *Geography.* Sometimes it is required that customers in the same region are serviced by the same vehicle [248].

The algorithm that will be presented in Chapter 5 can be extended to deal with any number of these objectives, however, as stated earlier, simulation results will only be presented for the standard number of routes $f_1(\mathcal{R})$ in (3.4) and travel distance $f_2(\mathcal{R})$ in (3.5), plus the delivery time $f_3(\mathcal{R})$ in (3.14). The optimisation of further objectives will be suggested as a direction of future research.

3.4.1 Overview of metaheuristic approaches

Multi-objective VRPs account for less research than the two variants surveyed earlier. Some of these studies are described here.

Rahoual et al. [197] tackled the VRPTW with a Genetic Algorithm (GA) based on the first version of the NSGA [223], taking into account the minimisation of the number of routes, the travel distance, and the penalties associated to the violated constraints. The constraints considered were the capacity, distance, and duration limits, in addition to the time windows. This approach considers a randomly generated initial population, single-point crossover [57], and a mutation operator which consists in changing the position of a customer from one vehicle to another using one out of of five different procedures, the choice of which to run is made randomly. The authors presented results to instances in the Solomon's benchmark categories C1 and R1, however, many of them present violated constraints, which means that the solutions obtained are infeasible.

Murata and Itai [177] proposed a two-fold multi-objective Evolutionary Algorithm (EA), based on NSGA-II [67], for solving a class of VRP with normal (NDP) and high (HDP) demands, considering the minimisation of the number of vehicles and the maximum routing time, i.e. the route with maximum duration. This algorithm generates a random initial population, uses the cycle crossover [181], and two mutation operators: one of them modifies the assignment of customers to routes and the other reverses the order in which customers are visited in a route. In the first stage they solved the NDP and use the resulting solutions to initialise the optimisation of the HDP. They also propose the *ratio of the same route* similarity measure in order to evaluate the similitude between solutions to both problems.

Jozefowiez et al. [134, 136] addressed the CVRP *with Route Balancing*, in which the total travel distance and the difference between the longest and shortest routes lengths are to be minimised. They implemented a parallel enhanced version of NSGA-II [67], which considered two crossover operators, namely the RBX of Potvin and Bengio [192] and the Split function of Prins [195], and a mutation based on 2-opt local search.

Ombuki et al. [182] considered VRPTW as a bi-objective optimisation problem, where the number of vehicles and the travel distance are to be minimised, and used a GA for solving it. In this approach 90% of the initial population are randomly generated solutions and the remaining 10% are solutions generated with a greedy procedure based on the nearest neighbour method. The authors introduced the *best cost route crossover* (BCRC), which aims at simultaneously minimising the number of vehicles and travel distance while checking feasibility, and proposed the *constrained route reversal mutation*, which purpose is to invert a sequence of customers.

Tan et al. [228] proposed a hybrid multi-objective EA for solving the VRPTW regarding the minimisation of the number of routes and the travel distance. This approach starts with a randomly generated initial population, which is then submitted to the designed *route-exchange crossover* and to a multi-mode mutation that consists in three operations, namely *partial swap*, *split route*, and *merge routes*, from which only one is executed. Additionally, the λ-interchange, and the proposed *intra_route* and *shortest_pf* local search heuristics were implemented, which were executed every 50 generations for all individuals in the population. This analysis found that, despite categories C1 and C2 have positively correlating objectives, the majority of the Solomon's benchmark instances in categories R1, R2, RC1 and RC2 have conflicting objectives, as was previously noticed from Table 3.5.

Tan et al. [229] slightly modified the previous approach for tackling the *Truck and Trailer* VRP. In this variant, different types of vehicles are considered, which means that some of them have certain limitations. The modified version of this algorithm considers the nearest neighbour density estimation technique in order to preserve population diversity.

Tan et al. [231] proposed a multi-objective evolutionary approach, based on the earlier approach of Tan et al. [228], for solving the SVRP in which the demand is the stochastic parameter. They considered, in addition to the minimisation of the number of routes and travel distance, the minimisation of driver remuneration, i.e. delivery time. An initial solution is generated so that it uses a random number of vehicles and approximately the same number of customers are serviced in each route. In this study, instead of the 2-opt and λ-interchange local search heuristics, two methods were implemented, namely the *shortest path search* and the *which directional search*.

Ghoseiri and Ghannadpour [111] presented a study using a goal programming approach for the formulation of the problem and an adapted a GA to solve it. Part of the initial population is initialised randomly and part is initialised by using the I1 heuristic and the λ-interchange local search. The authors introduced the *best cost-best route crossover* (BCBRC), which selects a best route from each parent and is very similar to the BCRC of Ombuki et al. [182] with minor differences, and the *sequenced based mutation* (it is actually a recombination operator), which, given two offspring solutions produced from the recombination phase, randomly selects an arc to break a route on each of the solutions and then make an exchange on the routes before and after the break points to produce two new offspring. Two local search heuristics are incorporated in the GA, from which one is executed at the end of each generation for a portion of the population. To our best knowledge, this is the only multi-objective study which actually presents the overall Pareto approximation found to each instance of the Solomon's benchmark set.

Pacheco and Martí [185] addressed the problem of routing school buses, which consists of transporting a group of students from their homes to a school, by means of Tabu Search (TS). They considered the minimisation of the total number of buses while simultaneously minimising the maximum time that a student spends in the bus, that is the longest route. The problem is solved by considering both objective functions separately. Since the value of the first objective, the number of routes (or buses), is a discrete number (bounded by the number of locations), the authors followed the simple method that consists of minimising the second objective, the maximum length of a route, for each possible value of the number of routes. This algorithm utilises the two constructive methods proposed by Corberán et al. [45] and the one by Fisher and Jaikumar [87] for generating initial solutions. Then, the TS employs a modified version of the CROSS exchange of Taillard et al. [226] to perform the local search.

Finally, Beham [16] proposed a TS approach for solving the VRPTW, and tested it on the instances of Homberger and Gehring [129], regarding the minimisation of the number of routes, the total travel distance, and the time constraint violation. In addition to the list of forbidden moves, two extra memories were used: a list of non-dominated solutions from previous neighbourhoods, which are used to restart the search process, and the archive containing the overall found non-dominated solutions. A solution can be added to the archive when it is not dominated by the solutions in the archive and this is not full. If the archive is full, the solution is added based on the result of the crowding distance [67]. This method starts by generating an initial solution with the I1 heuristic, which is then improved by selecting one of the local search heuristics 2-opt, Or-opt, and λ-interchange.

3.4.2 Results from previous studies

With the exception of the study of Ghoseiri and Ghannadpour [111], the remaining multi-criteria studies mentioned above which tackled the Solomon's instances, namely those of Rahoual et al. [197], Ombuki et al. [182], and Tan et al. [228], did not make their results available in a proper multi-objective manner. Instead, Ombuki et al. [182] reported their results for the solution with the smallest number of routes and for that with the shortest travel distance. Rahoual et al. [197] and Tan et al. [228] presented their results only for the solution with shortest travel distance.

Following the statement above, and similarly to the single-objective case, results from these multi-objective studies are presented in two ways. The first considers the difference between their best results and the best-known, regarding both the solution with the smallest number of routes and that with the shortest travel distance. The average difference over the instances in each set category are shown in Table 3.11 for the solutions with the smallest number of routes and in Table 3.12 for the solutions with the shortest travel distance. These tables have the same format as Tables 3.6 and 3.7.

In Table 3.11 we see that the approach of Rahoual et al. [197] obtained the smallest difference in the number of routes (6.16%), however they only presented results for two categories. The EA of Tan et al. [228] obtained solutions which nearly doubled (13.93%) the number of routes from those obtained by Ombuki et al. [182] (7.19%), though they correspond to a saving in the travel distance (0.8%).

On the other hand, regarding solutions with the shortest travel distance, we observe in Table 3.12 that the GA of Ombuki et al. [182] achieved the smallest difference

Author		C1	C2	R1	R2	RC1	RC2	Total
Rahoual et al. [197]		0.00		10.78				6.16
		7.18		13.17				10.60
Ombuki et al. [182]		0.00	0.00	7.83	16.67	7.35	8.33	7.19
		0.01	0.13	0.29	1.16	0.03	3.19	0.77
Tan et al. [228]		0.00	0.00	10.12	32.58	7.31	30.21	13.93
		0.06	0.16	-1.70	0.93	-1.75	-2.85	-0.80
Ghoseiri and Ghannadpour [111]		0.00	0.00	9.75	31.82	10.81	16.67	12.26
		0.00	0.28	1.68	9.92	0.90	4.35	3.10

Table 3.11: Average difference, grouped by instance category, between results from previous multi-objective studies and the best-known results, considering the solutions with the smallest number of routes to Solomon's benchmark set.

Author		C1	C2	R1	R2	RC1	RC2	Total
Rahoual et al. [197]		0.00		-0.03				-0.02
		7.18		16.26				12.37
Ombuki et al. [182]		0.00	0.00	1.06	-13.31	2.27	-3.57	-2.57
		0.01	0.13	2.47	1.72	3.72	3.49	1.92
Tan et al. [228]		0.00	0.00	-0.64	-31.62	-2.66	-26.53	-10.52
		0.06	0.16	1.04	8.49	1.25	8.58	3.33
Ghoseiri and Ghannadpour [111]		0.00	0.00	14.38	46.97	14.84	25.00	18.00
		0.00	0.28	0.80	11.15	0.41	3.87	3.01

Table 3.12: Average difference, grouped by instance category, between results from previous multi-objective studies and the best-known results, considering the solutions with the shortest travel distance to Solomon's benchmark set.

between their results and the best-known (1.92%), followed by the approach of Ghoseiri and Ghannadpour [111] (3.01%). In contrast to the single-objective proposals, these studies were not able to find the best-known solutions for all instances in categories C1 and C2, since the difference in the travel distance is not 0%, except for the algorithm of Ghoseiri and Ghannadpour [111] which obtained the best-known solutions only for instances in category C1.

The second way of presenting results is the traditionally found in the literature, which corresponds to the format of Tables 3.8 and 3.9. Results from previous multi-objective studies are shown in Table 3.13, which shows, for each author (row) and instance category (column), the average number of routes (upper figure) and the

Author	C1	C2	R1	R2	RC1	RC	Total
Rahoual et al. [197]	10.00		12.90		12.60		
	887.78		1362.17		1487.00		
Ombuki et al. [182]	10.00	3.00	12.67	3.09	12.38	3.50	427.00
(min R)	828.48	590.60	1212.58	956.73	1379.87	1148.66	57484.35
Ombuki et al. [182]	10.00	3.00	13.17	4.55	13.00	5.63	471.00
(min D)	828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
Tan et al. [228]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ghoseiri and Ghannadpour	10.00	3.00	12.92	3.45	12.75	3.75	439.00
[111] (min R)	828.38	591.49	1228.60	1033.53	1392.09	1162.40	58735.22
Ghoseiri and Ghannadpour	10.00	3.00	13.50	3.82	13.25	4.00	456.00
[111] (min D)	828.38	591.49	1217.03	1049.62	1384.3	1157.41	58671.12

Table 3.13: Best average results from previous multi-objective studies, grouped by instance category, for the Solomon's benchmark set.

average travel distance (lower figure). Additionally, the last column presents, the total number of routes and total travel distance for all 56 instances. In this case, the studies of Ombuki et al. [182] and Ghoseiri and Ghannadpour [111] present two series of results, one corresponding to the solutions with the smallest number of routes (min R) and the other regarding solutions with the shortest travel distance (min D).

From these studies, we see that the one of Ombuki et al. [182] (min R) obtained the solutions with the smallest number of routes, in all categories and in total, though they have an increased travel distance when compared with the other approaches. The EA of Tan et al. [228] achieved solutions with the shortest travel distance to instances in categories R1 and RC1, while that of Ghoseiri and Ghannadpour [111] for category C1 and that of Ombuki et al. [182] (min D) for the remaining categories C2, R2 and RC2, and in total.

These studies will also be considered in Chapters 4 and 5, when evaluating the performance of the designed multi-objective EAs, in order to known how its performance compares with previous multi-objective approaches.

3.5 The way forward

According to the reviews presented in this chapter, there is vast research regarding the solution of the VRP using different heuristic and metaheuristic methods. However, some issues can be highlighted, as they are open useful research directions.

Firstly, Tabu Search and Evolutionary Algorithms have been widely applied to the VRPs of interest to this thesis with successful results. Their performance and computational requirements depend on the use of local search strategies, additional memory for saving previous solutions, and post-optimisation procedures. Importantly for Tabu Search methods is the definition of (reduced) neighbourhoods and the update of the list of forbidden moves.

Secondly, as was pointed out in Section 3.2, despite delivery time not being considered as the primary objective to be minimised in many of the variants of the VRP, in real-life circumstances it plays an important role, as companies offering transportation services are often interested in reducing the overall delivery time, or driver salary cost, as well as the overall travel distance, or fuel cost, and those can be in conflict. With the exception of the study by Tan et al. [231], several objectives are considered to be minimised, but never the delivery time.

From all the publicly available literature regarding the solution of VRPs, the vast majority of the methods has considered the optimisation of prioritised objective functions, e.g. the travel distance is minimised for each number of routes, and only a really small amount of studies has considered multiple non-prioritised objectives to be simultaneously optimised. Furthermore, from the few methods which proposed solving VRPs regarding multiple criteria, very few reported the detail of their results, and, when available, they were not presented in a proper multi-objective manner. The rest of the studies show their results in a traditional single-objective style, i.e.

best and average best results of the optimised objectives, which may be misleading when interpreting multi-objective performance.

On the other hand, only a small amount of the studies proposing evolutionary computation methods have explicitly considered the preservation of population diversity, despite it being well known to be of great importance for the success of any Evolutionary Algorithm.

To lead into the presentation and analysis of our proposal for solving two variants of the VRP, it is useful to quote part of the conclusions of Jozefowiez et al. [135]:

> "Despite this recent increase in the number of studies on multi-objective vehicle routing problems, almost every study appears to have been undertaken independently of all the others. Cleraly [sic], some of these studies could have been linked together. In some cases, the different studies deal with the same or almost the same multi-objective problem; when the entire problem is not exactly the same, at least several objectives are shared. Based on this observation, it would seem that there is a need to define some general multi-objective vehicle routing problems that could be used as the starting points for more complex problems.
>
> "The studies employing multi-objective evolutionary algorithm usually limit themselves to operators from the literature, which were designed to solve for an objective associated to the single-objective problem underlying the studied multi-objective problems. There is a real need for future studies to develop operators for the other objectives, as well as operators that can deal with several objectives simultaneously. The need to define general multi-objective vehicle routing problems is closely connected to this need to develop new operators."

The remainder of this thesis seeks to remedy the unavailability of appropriate presentation of results and analysis regarding the multiple criteria optimisation of VRPs. Furthermore, the design of an Evolutionary Algorithm, which includes specialised crossover and mutation operators, and a mechanism to maintain population diversity, for solving a range of VRPs regarding the optimisation of varied objectives is addressed.

3.6 Summary

This chapter presented the description and formal definition of a number of variants of the Vehicle Routing Problem (VRP), which consists in designing a minimum-cost set of routes in order to service requests from customers. Such routes are covered by using a fleet of vehicles based at the depot, from where customers are supplied. Cost is generally related to the number of routes, or the vehicle cost, and the travel distance, relating to fuel cost, but there are several other sources of cost, e.g. the driver's remuneration or delivery time.

This problem has a number of variants, which consider different restrictions, that are close to real-world applications. For example, the Capacitated VRP has restrictions on the vehicle capacity, which must not be exceeded, and in the VRP with Time Windows, in addition to the capacity constraint, customers have to be serviced within specific times.

Many other variants of the problem have been subject of study, being the difference between them the customer precedence and the operational constraints. Furthermore, several other objective functions have been considered to be optimised, ranging from the associated economic cost, like workload imbalance and route length, to problem-specific objectives, like security, welfare and health.

Some studies that are of importance to this thesis were reviewed. These are related to the solution of a number of variants of the VRP by means of utilising general-purpose heuristic methods, such as Tabu Search (TS) and Evolutionary Algorithms (EA). With the aim of better exploiting the search space and escaping from sub-optimum solutions, many TS and EA methods have been combined with local search heuristics, which make the approaches require additional resources. In particular, the performance of local search techniques depend on the definition of neighbourhoods, and TS methods require efficient strategies for updating the tabu list. Additionally, the majority of the evolutionary approaches did not explicitly considered population diversity preservation.

Many of the surveyed approaches took into account the optimisation of only one objective, and many others assigned priorities to the objectives being optimised, procedure which do not lead to the finding of an appropriate Pareto approximation in a single run. On the other hand, the studies that did perform a proper multiple objective optimisation, did not regard the minimisation of the delivery time, which is often considered in real-world applications to be optimised. Furthermore, they analysed their results in a single-objective manner, which might be deceiving for problems with multiple criteria, instead of applying proper multi-objective quality indicators to compare performance.

Chapter 4

Preliminary approaches to solving VRPs with Time Windows

During the progress of the present study, a series of algorithms for solving the VRPTW as a multi-objective problem were developed. More precisely, an algorithm followed an evolution which was based on the needs and findings suggested by the analysis of the results from the earlier stages of the research: after an exploratory Evolutionary Algorithm, a multi-objective density-restricted Genetic Algorithm (drGA) was proposed, which after some modifications became a Bi-objective Evolutionary Algorithm (BiEA), and finally it turned into an improved Multi-Objective Evolutionary Algorithm (MOEA). This chapter focuses on describing the preliminary development of the aforementioned algorithm and the next will concentrate exclusively on the description of the final MOEA.

4.1 Exploratory Evolutionary Algorithm

The initial approach to solving the VRPTW was a *simple* Evolutionary Algorithm (EA), which aimed at simultaneously minimising the number of routes and travel distance. This prototype was inspired by the fact that mostly all previous research using evolutionary approaches proposed a hybrid optimiser, i.e. an EA combined with other sorts of heuristics.

4.1.1 Experimental analysis

The exploratory EA was tested on the Solomon's benchmark set and, after the analysis of the results, it became clear that population diversity was a crucial factor for the success of the algorithm, in the sense that it suffered from premature lack of population diversity and became stuck in sub-optimum solutions. At this stage of the research, population diversity was estimated by computing the variance in the objective functions values of all solutions in the population.

Figure 4.1 presents six series of plots, one for each of the test instance categories, showing the average normalised variance in each objective function, that is in the number of routes R (upper plot) and in the travel distance D (lower plot), over the generations. Notice that the vertical axis is in logarithmic scale in all plots. In some cases, specially for the variance in R, there is no information plotted, which means that the variance was zero for those generations in the evolutionary process.

The first observation we can make is that, after a few evolutionary generations, the normalised variance in R is below the range of 10^{-3} for all instance categories. If we consider that solutions to these instances have no more than 25 routes, then the variance in R was of only hundredths of a route, that is the number of routes varied approximately 2%.

In the case of the travel distance, the normalised variance was approximately in the range of 10^{-5}. Solutions have between 500 and 2000 units of distance, which means that the variance was of hundredths of a unit, i.e. between 0.005% and 0.02%.

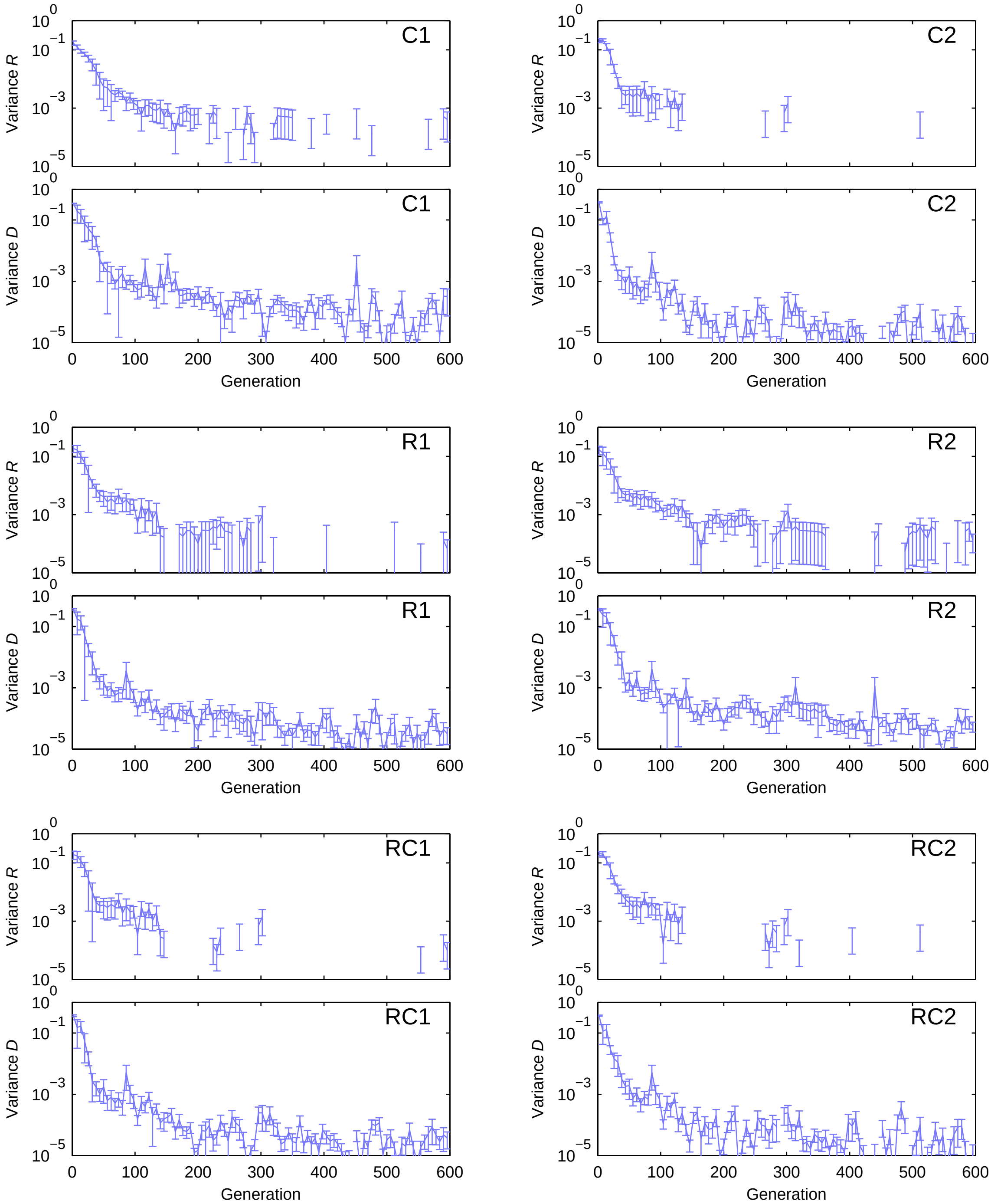


Figure 4.1: Average normalised variance in the objective functions values of all solutions in the population, over the instances in each set category, presented by the exploratory approach.

The figures above indicate that the population actually presented a lack of diversity. Moreover, if we consider that the variance did not present a significant change after the first fourth of the evolutionary process, we could argue that the loss of diversity was premature. Consequently, the need for a method to promote and maintain population diversity was evident.

Results from this approach were not near neither the best-known results nor the best average results presented in Tables 3.5 and 3.10, thus their comparison is not going to be presented in this case.

4.2 Multi-objective density-restricted Genetic Algorithm

This section presents the proposed multi-objective density-restricted Genetic Algorithm (drGA) for solving VRPTW as a bi-objective problem. The main contribution of this algorithm, which is a method to restrict density of solutions as an attempt to preserve diversity, along with preliminary results and analysis are presented.

4.2.1 Algorithm design

EAs have a number of procedures that must be specified in order to define their operation (see Section 2.2.3). Hence, the drGA design, including the solution encoding, fitness function, how density of solutions is computed, and the stages of processing, is described next.

4.2.1.1 Solution encoding

The idea for the solution encoding was inspired by Falkenauer [85] and is depicted in Figure 4.2, which shows the encoding of a solution to an example instance with 10 customers, i.e. $N = 10$. It consists of a chromosome with three parts. The first position in the chromosome indicates the number of routes in the solution (3 in this example). The following N genes are a permutation of the N customers $(1, \ldots, 10)$. The rest of the genes specify the position in the chromosome of the last customer

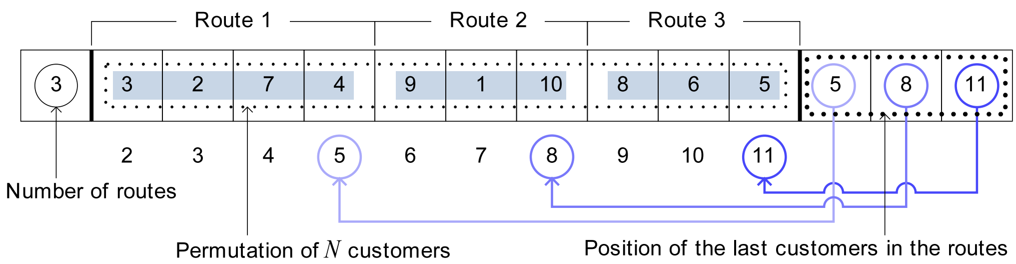


Figure 4.2: Encoding of a solution to an example instance of the VRPTW with 10 customers, which consists of three routes: $\langle 3, 2, 7, 4 \rangle$, $\langle 9, 1, 10 \rangle$, and $\langle 8, 6, 5 \rangle$

in every route. In this example, the first route includes customers $3, 2, 7, 4$, being customer 4 the last customer of the route, as specified in position 12. The order in which these customers are serviced is exactly as they appear in the chromosome, hence the three routes are $\langle 3, 2, 7, 4 \rangle$, $\langle 9, 1, 10 \rangle$ and $\langle 8, 6, 5 \rangle$.

4.2.1.2 Initial population

It is standard practice for an EA to begin with an initial population chosen randomly with the aim of covering the entire search space. Thus, drGA starts with a population P of initial solutions, $|P| = popSize$, each being a randomly generated feasible route. These routes are constructed by the following process: First, a customer is selected at random and placed as the first location to be visited on the first route. Then, a different customer is randomly chosen and, if the capacity and time constraints would be met, it is placed on the current route after the previous customer. If any of the constraints are not met, a new route is created and this customer is the first location to be visited on that route. This process is repeated until all customers have been assigned to a route.

Initial solutions could have been generated by means of the construction heuristics reviewed in Sections 3.1.2 and 3.2.2, however, the aim of this research was to design an EA without additional heuristics.

4.2.1.3 Fitness assignment

drGA assigns fitness to individuals using the dominance depth criterion reviewed in Section 2.4.1.3, where the population is divided into several non-dominated fronts and the depth specifies the fitness of the individuals belonging to them. As a matter of fact, it implements the function FastNonDominatedFront() of Deb et al. [67] presented in Section 2.4.5.2 (Algorithm 2.7).

4.2.1.4 Density of solutions

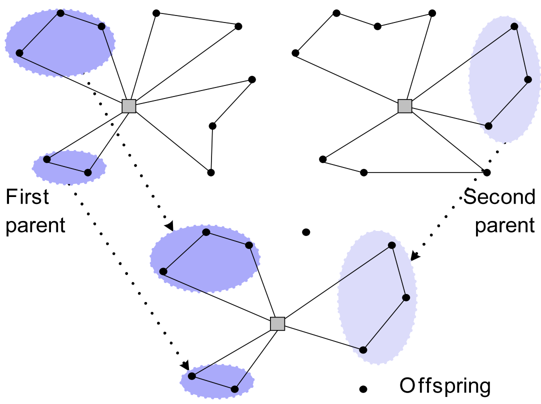
To overcome the situation in which the exploratory algorithm gets stuck in sub-optimum solutions and to prevent the lack of population diversity, a feature to help in the selection process is implemented in drGA. It considers density information, using the nearest neighbour method (see Section 2.4.2.2), to estimate the diversity of solutions. Specifically, it takes into account the number of equal solutions to determine and restrict density.

In other words, drGA introduces a new parameter ϱ, which is called *diversity ratio*, to restrict the density of equal solutions from growing indiscriminately. That is, if $\varrho = popSize$, drGA will allow the whole population to be the same solution. On the other hand, if $\varrho = 1$, drGA restricts all solutions in the population to be unique.

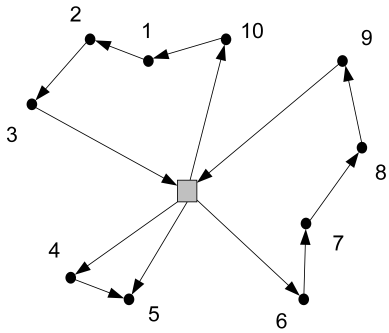
drGA computes the density $\Delta(s_i)$ of solution s_i as

$$\Delta(s_i) = \left|\{s_j \in P \mid \boldsymbol{f}(s_i) = \boldsymbol{f}(s_j), s_i \neq s_j\}\right| . \tag{4.1}$$

That is, drGA counts the number of solutions $s_j \in P$ that have equal objective function values to those of s_i. The density information is used in the parent and survival selection stages to bias the selection of individuals.



(a) Copying routes from parents



(b) Resulting offspring

Figure 4.3: The recombination process in drGA: A random number of routes are copied from the first parent into the offspring and from the second all those routes which are not in conflict with the customers already copied. If there remain any unassigned customers, these are allocated to any of the existing routes

4.2.1.5 Parent selection

The standard Tournament Selection method, reviewed in Section 2.2.3.4, is used in drGA to select two parents for the recombination process. It chooses $Tsize$ individuals s_i randomly from the population P and selects the fittest individual from this group to be one of the parents. In case there is a tie between individuals, density information is used to select the winner, in the sense that the solution having the lowest density wins the tournament, since less common individuals are preferred.

4.2.1.6 Recombination

The recombination operator takes a traditional form suggested by Falkenauer [85], and works as follows. The algorithm here is designed to randomly select and preserve routes from both parents. The recombination of two example parents is shown in Figure 4.3(a). First, a random number of routes are chosen from the first parent and copied into the offspring. In the example, both routes on the left from the first parent were selected to be copied into the offspring. Then all those routes from the second parent which are not in conflict with customers already copied from the first, are copied into the offspring. In the example, only the route on the right can be copied from the second parent, since the other two contain customers already present in the offspring. Finally, if there remain any unassigned customers, these are allocated, in the order they appear in the second parent, to any of the existing routes, as in the example shown in Figure 4.3(b). If there is no way to insert such remaining customers into the existing routes without violating the constraints, a new route is created. The resulting solution s'_i is stored in the offspring population Q, and parent selection and recombination are repeated until $|Q| = popSize$.

4.2.1.7 Mutation

Once an offspring s'_i has been generated, it is submitted to the mutation process. drGA implements six possible mutation operators which can be categorised as inter-route and intra-route operations. In the former, the algorithm will perform changes between two routes, thus modifying the assignment of customers to routes, and in the latter, the changes will be done within a route, hence affecting the travel sequence.

In the first category, three viable processes can be identified:

- removing a sequence of customers from one route and *inserting* them into another,
- *swapping* two sequences of customers from two different routes, and
- *merging* one route into another,

which can be seen as λ-interchange moves. In the case of the intra-route modifications, drGA uses three operations:

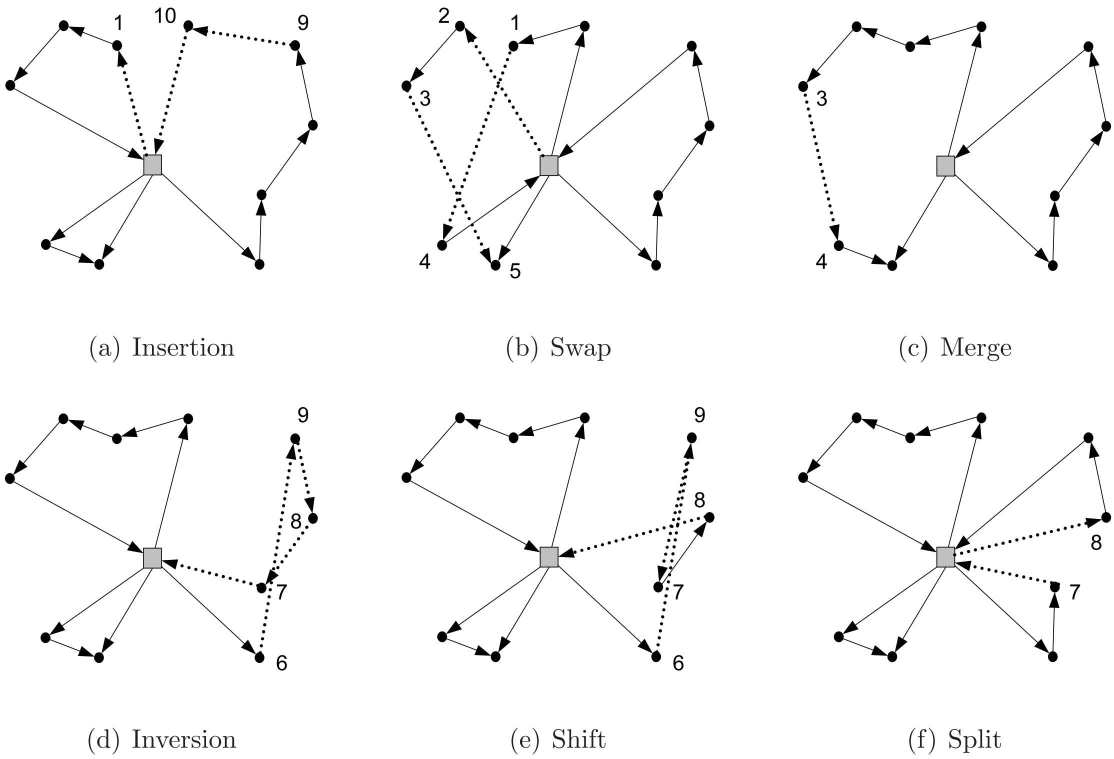


Figure 4.4: The six mutation operations used in drGA which can be categorised as inter-route, the three at the top, and intra-route, the three at the bottom. The original offspring is shown in Figure 4.3(b). The dotted lines in each figure represent the changes in the sequences.

- the *inversion* of a sequence of customers, which is a 2-opt move,
- the *shift* of one customer, which can be considered an Or-opt move, and
- *splitting* a route, which may be regarded as a λ-interchange move.

Examples of these operations are shown graphically in Figure 4.4, which are the mutations of the original routes shown in Figure 4.3(b). The dotted lines in each figure represent the changes made to the routes. In Figure 4.4(a), customer 10 was removed from the upper-left route and inserted into the route on the right. Figure 4.4(b) shows the swap of customer 4 with customers 2 and 3. In Figure 4.4(c), the bottom-left route has been merged into the route on top-left, connecting customer 4 after customer 3. Figure 4.4(d) illustrates the inversion of the sequence of customers $\langle\langle 7, 8, 9\rangle\rangle$ to $\langle\langle 9, 8, 7\rangle\rangle$. In Figure 4.4(e) we see how customer 9 has been shifted between customers 6 and 7. Finally, Figure 4.4(f) depicts how the route on the right has been split between customers 7 and 8. In this latter case, the split operator itself tries to insert the customers belonging to the smallest sub-route into any other existing route.

From these six operations only one is executed, the choice on which to apply is done stochastically, accordingly to the weights ω_{operator} that the operators are pre-assigned, such that the sum of these weights adds up 1.0. This means that, the higher the weight of an operator, more likely it is to be performed. Split and merge share the same weight, however, the decision on splitting a route is inversely proportional to the number of routes in the solution. That is, the probability of splitting a route is higher if the number of routes is small.

4.2.1.8 Survival selection

After the crossover and mutation stages, the offspring population Q is merged into the parent population P and the density of the individuals is computed. If the density $\Delta(s_i)$ of solution s_i has grown to more than the maximum allowed (ϱ), further individuals are removed from the population until the density is rectified. That is, if $\Delta(s_i) > \varrho$, individuals $s_j, \boldsymbol{f}(\boldsymbol{s_j}) = \boldsymbol{f}(\boldsymbol{s_i})$, are removed from the population until $\Delta(s_i) = \varrho$.

Afterwards, the population is grouped into non-dominated fronts and fitness is assigned to individuals. Then, elitism is considered to form the population for the next generation, which is comprised by the individuals belonging to the fittest fronts. If the population size is exceeded in the final selected front, solutions with shortest travel distance are preferred. This process is presented in Algorithm 4.1, which is actually a modified version of the survival selection procedure of NSGA-II (Section 2.4.5.2).

Algorithm 4.1: SurvivalSelection(P)

Input: Population $P = \{s_1, \ldots, s_{popSize}\}$ submitted to the survival selection process
Output: Elite solutions $P^* = \{s_1^*, \ldots, s_{popSize}^*\}$

for all $s_i \in P$ **do**
  for all $s_j \in P \setminus \{s_i\}$ **do**
    if $\boldsymbol{f}(s_i) = \boldsymbol{f}(s_j)$ **then**
      $P \leftarrow P \setminus \{s_j\}$
    end if
  end for
end for
$\mathcal{F} \leftarrow$ FastNonDominatedSort(P)
$P^* \leftarrow \emptyset$
$k \leftarrow 1$
while $|P| + |\mathcal{F}_k| \leq popSize$ **do**
  $P^* \leftarrow P^* \cup \mathcal{F}_k$
  $k \leftarrow k + 1$
end while
$L \leftarrow$ Sort($\mathcal{F}_k, f_2$) /* *Sort front* $\mathcal{F}_k$ *according to travel distance* */
$P^* \leftarrow P^* \cup \{L[1..(popSize - |P^*|)]\}$
return P^*

4.2.2 Experimental analysis

First experiments were carried out for parameter tuning purposes. Several settings were tested, maintaining fixed some of the parameters while varying others, and vice versa. In this context, many parameters have standard values that have been used in many past studies. For example, a very low mutation probability perhaps will not help the algorithm to escape from sub-optimum solutions, and, on the contrary, a very high probability could lead to a deterioration of the results. However, as stated in Section 2.2.3.9, the task of finding the *right* parameter values is a problem itself and time-consuming. Since a parametric study is not the core of this investigation, appropriate analysis of the parameters involved in the algorithms presented in this thesis is proposed as a subsequent research activity. Hence, for the remaining experimental analysis in this chapter an in the next, only the parameter values found to improve the algorithm's performance are going to be presented.

Set Category	$N = 25$				$N = 50$			
	Overall best		Average best		Overall best		Average best	
	%	SD	%	SD	%	SD	%	SD
C1	0.26	0.02	3.65	3.63	1.89	3.23	8.91	5.10
C2	0.39	0.01	2.04	1.57	0.59	0.40	2.14	2.41
R1	0.24	0.03	2.86	0.96	3.30	1.31	6.54	2.01
R2	0.83	0.69	3.96	1.31	2.94	1.33	7.26	2.20
RC1	0.29	0.13	1.99	2.40	1.69	1.50	6.55	3.19
RC2	1.73	3.28	5.74	4.78	0.56	0.59	8.75	2.46
Average	0.62		3.37		1.83		6.69	
St. Dev.	0.59		1.41		1.15		2.46	

Table 4.1: Summary of the results from drGA for the Solomon [218]'s instances size $N = 25$ and $N = 50$.

drGA was run 10 times on each of the Solomon's instances size $N = 25, 50, 100$, set to minimise the number of routes and the travel distance. The values that made the algorithm perform better are the following:

$$\begin{array}{llll} \varrho = 1 & Tsize = 5 & \omega_{\text{insert}} = 0.3 & \omega_{\text{shift}} = 0.15 \\ popSize = 200 & \gamma = 0.8 & \omega_{\text{swap}} = 0.3 & \omega_{\text{split}} = 0.05 \\ numGen = 600 & \mu = 0.2 & \omega_{\text{invert}} = 0.2 & \end{array}$$

4.2.2.1 Comparison with optimum solutions

Solomon's smallest instances, size $N = 25, 50$, have been solved to optimality, minimising first the number of routes and then, for each value of this objective, the travel distance. Hence, results from drGA can be compared with them in order to know its performance.

The summary of the results from drGA is shown in Table 4.1, where they are grouped by instance category, i.e. C1, C2, R1, R2, RC1 and RC2. This table presents, for each instance size N, the average percentage difference (%), and corresponding standard deviation (SD), between the optimum results[1] and the overall best and average best results obtained by drGA after 10 repetitons, where best refers to the lowest travel

[1]Taken from Solomon's web site: `http://w.cba.neu.edu/~msolomon/problems.htm`

distance. Analysing the results in this table, we can see that for instances size $N = 25$, the average difference between the optimum results and the overall best results obtained by drGA is of only 0.62%, and 3.37% if we consider the average best results. For instances size $N = 50$, the differences are of 1.83% and 6.69%, respectively.

4.2.2.2 Comparison with previous studies

Although the performance of drGA was not the best, it was acceptable, thus it was also tested on the instances $N = 100$. Table 4.2 presents the results obtained by drGA, averaged over the instances in each set category, and they are compared with the best average results from Section 3.2.3 and with those from the multi-objective studies in Section 3.4.1. Despite the fact that drGA was set to optimise both objectives, only for this occasion, results from drGA are compared considering exclusively the solutions with the shortest travel distance, since, as we will see, its performance was not close to the results from previous studies and including more information would not lead to a different conclusion in this case.

For each instance, the solution from drGA with the shortest travel distance after all repetitions was taken. Then, the number of routes and travel distance were averaged over the instances in each set category. These averages are presented in Table 4.2. For each author and instance set, the average number of routes (upper figure) and the average travel distance (lower figure) are shown. The last column presents the total number of vehicles and the total travel distance for all 56 instances. The last two rows indicate, for each instance category, the percentage difference between the results from drGA and the best average results, regarding the number of routes (% diff. R) and the travel distance (% diff. D).

Author	C1	C2	R1	R2	RC1	RC2	Total
min R	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.86	1212.73	952.67	1384.30	1108.52	57192.00
min D	10.00	3.00	11.92	5.36	12.88	6.25	486.00
	828.38	589.86	1121.10	878.41	1341.67	1004.20	54779.02
Ombuki et al. [182]	10.00	3.00	13.17	4.55	13.00	5.63	471.00
	828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
Tan et al. [228]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ghoseiri and Ghannadpour	10.00	3.00	13.50	3.82	13.25	4.00	456.00
[111]	828.38	591.49	1217.03	1049.62	1384.3	1157.41	58671.12
drGA	10.00	3.44	13.42	3.82	13.00	4.75	459.00
	845.33	592.74	1252.38	959.44	1412.77	1096.99	58010.31
% diff. R	0.00	14.67	12.68	39.93	13.04	46.15	13.33
% diff. D	2.05	0.49	11.71	9.22	5.30	9.24	5.90

Table 4.2: Number of routes and travel distance, averaged over categories, for the best solutions found by previous studies and by drGA.

Analysing these results, we can observe that for instance sets C1 and C2, drGA obtained the highest costs for both objectives. On the other hand, for the remaining categories and in total, solutions from drGA are between 5% and 12% larger in travel distance than the best from the previous studies, and consider between 12% and 47% more routes. One interesting observation is that results for sets R2 and RC2 suggest a multi-objective nature of VRPTW, in that drGA obtained larger travel distances using a smaller number of vehicles when compared with some of the previous studies, e.g. that of Ombuki et al. [182]. This issue will be analysed later in more detail with the improved algorithms.

The preliminary study described up to here was accepted for publication and presented at the 2008 UK Workshop on Computational Intelligence (UKCI 2008) [99].

4.2.2.3 Analysis of the population diversity

In order to know if drGA preserved a higher diversity than the exploratory EA, both algorithms are compared in Figure 4.1. This figure presents six series of plots,

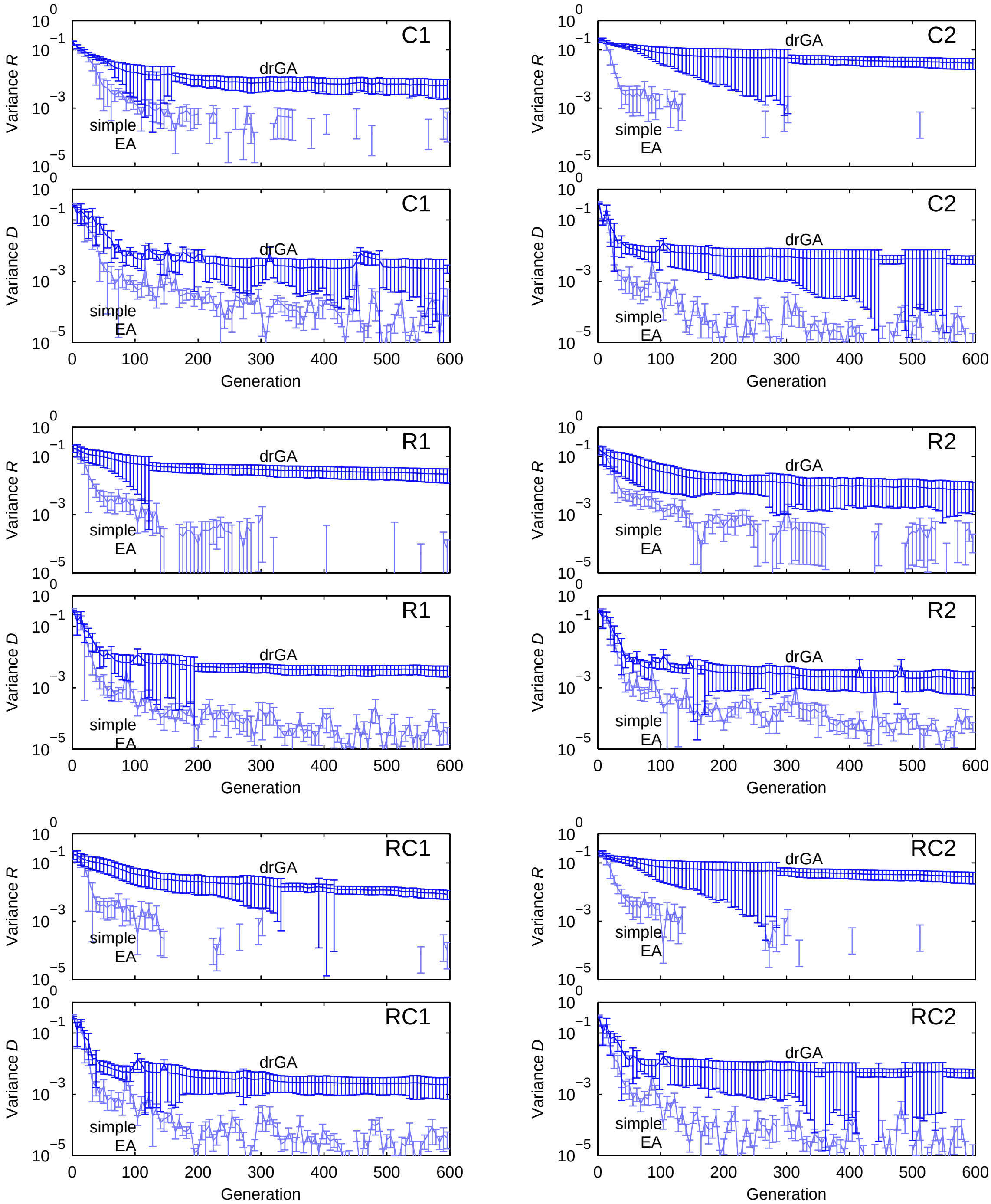


Figure 4.5: Average population variance in the objective functions, over the instances in each set category, presented by the exploratory approach and rdGA.

one for each set category, showing the average normalised variance in each objective function, that is in the number of routes R (upper plot) and in the travel distance D (lower plot), over the generations. Notice that the vertical axis in all plots is in logarithmic scale. We can observe that drGA preserved a higher diversity than the exploratory EA for all instance categories and in some cases, namely for instances in categories R1, RC1 and RC2, the difference in the normalised variance is of nearly two orders of magnitude. This results indicate that the density control is achieving the purpose of promoting and preserving population diversity. However, the fact of having all different solutions in the population was not enough to obtain results comparable to the best-known. Hence, an improved method to promote and guarantee population diversity, which could help the algorithm to better explore and exploit the search space, was necessary.

4.2.3 A note on the presentation of results

As was stated in Section 3.5, because previous studies which considered multiple objectives while solving VRPTW did not present their results in a proper multi-objective manner, results from drGA could not be compared with them from an appropriate multiple criteria point of view. Moreover, the average number of routes and the average travel distance have been used as the standard benchmark for comparing the performance in single-objective and multi-objective optimisation. This is why the comparison with previous studies was made using a conventional table as shown above. In the literature this is the traditional method for comparing results for this problem, thus the format of this table will be used for the remaining of the thesis for comparing results from the proposed approaches with those from earlier studies‘, even though such analyses could be misleading. Tables like this will always present the average of the best results found after all repetitions over the instances in each set category.

On the other hand, the studies with which the proposed algorithms are compared were selected according to the overall success of the results presented in Section 3.2.3, in addition to the fact that some of them are of interest to this research, as they solved the VRPTW as a multi-objective problem.

Nevertheless, later in this chapter and in the next, when the proposed algorithms are compared with each other and proper non-dominated solutions are available, the formal multi-objective metrics reviewed in Section 2.3.2 are going to be used in order to perform more reliable comparisons.

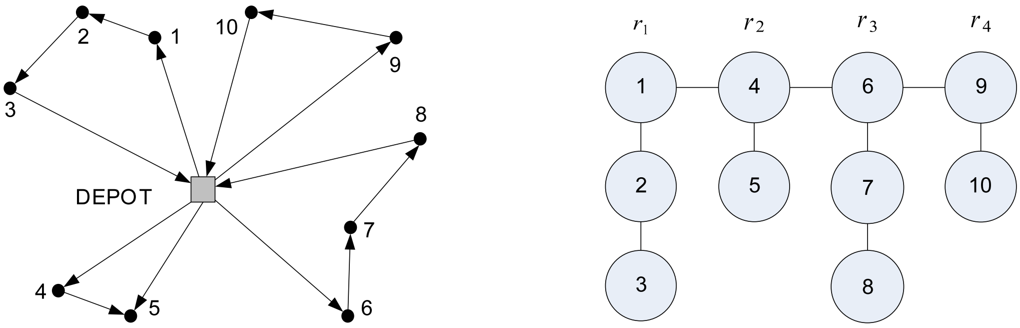
For the remaining of this thesis, similar to this section, each of the proposed algorithms are going to be analysed from several perspectives following the next method. Firstly, the designed approach will be compared with itself using other settings, e.g. different objective functions. Secondly, the algorithm will be contrasted with the previous approach, for example, drGA with the exploratory EA. Afterwards, the best solutions obtained to each instance are going to be compared with those from previous studies using the format of Table 4.2. Subsequently, the average population diversity preserved throughout the evolutionary process will be presented. Finally, specifically in the next chapter, the algorithms will be compared against NSGA-II.

4.3 Bi-objective Evolutionary Algorithm

Results from drGA highlighted the need for a more sophisticated method to preserve diversity and have a wider exploration and exploitation of the search space. For this reason, some modifications were made to drGA in order to enhance its performance. The new modified algorithm is called Bi-objective Evolutionary Algorithm (BiEA). The detail of the applied changes, as well as the improved population diversity techniques, are described below.

4.3.1 Algorithm design

This section describes BiEA design, including a new solution encoding, a solution similarity measure, which is used to know how similar a solution is to the rest of the



(a) Solution (b) Encoding

Figure 4.6: Solution to an example instance of the VRP with 10 customers and its encoding: a route is simply a list of customers which are serviced as they are listed and a solution is a list of routes.

population and to quantify population diversity, and the stages of processing which differ from drGA.

4.3.1.1 Solution encoding

Since a solution to the VRP is a list of routes, which are themselves lists of customers, the solution encoding was changed to a list of lists. Now, a route encoding simply lists the customers in the order they are serviced, and the solution encoding lists a number of routes. This is a more appropriate representation because the route delimiters which were used in drGA are not present any more, hence the decoding task is no longer required, additionally to the ease with which the evolutionary operations are implemented. A solution to an example instance and its encoding are shown in Figure 4.6. In this example, the first route, r_1, is formed by the customers 1, 2 and 3, and they are serviced in that order, i.e. $r_1 = \langle 1, 2, 3 \rangle$. The other routes are $r_2 = \langle 4, 5 \rangle$, $r_3 = \langle 6, 7, 8 \rangle$, and $r_4 = \langle 9, 10 \rangle$.

4.3.1.2 Solution similarity measure

The concept of population diversity not only refers to the number of distinct solutions there are in the population, but also to how different solutions are among them. It is relatively simple to determine the number of distinct solutions there are in a population and to make sure there are no duplicates, however, evaluating how solutions are spread in the search space generally requires the use of encoding-specific and problem-specific tools. If this information is known, it could be used to boost and maintain population diversity.

To accomplish this, the obvious starting point was the edit distance, introduced in Section 2.5.4, however, as will be seen later, it is computationally intensive. Thus a new similarity measure for solutions to the VRP was designed and the two methods are compared in Section 4.3.2.1. The designed solution similarity measure is based on *Jaccard's similarity coefficient* [131, 73], which measures the similarity of two sets as the ratio of the cardinality of the intersection to the cardinality of the union of those sets. Formally, the Jaccard similarity $J(A, B)$ of sets A and B is:

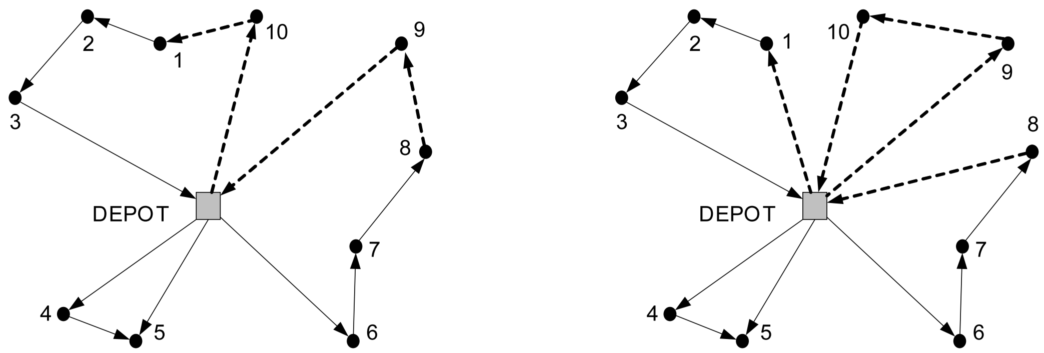
$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \ . \tag{4.2}$$

Thus if both sets contain the same elements, the intersection will equal the union, and $J(A, B) = 1$. On the other hand, if the sets do not share any element at all, the intersection will be the empty set, i.e. $|A \cap B| = 0$, and $J(A, B) = 0$.

The natural way to implement Jaccard similarity for solutions to the VRP is to consider each solution $\mathcal{R}$ as the set of segments or arcs $\big(u(i,k), u(i+1,k)\big)$ of each route r_k, i.e.

$$\mathcal{R} = \bigcup_{r_k \in \mathcal{R}} \bigcup_{i=0}^{N_k} \left\{ \big(u(i,k), u(i+1,k)\big) \right\} \ . \tag{4.3}$$

Then, the similarity of two solutions equals the ratio between the number of arcs that are common to both solutions and the total number of arcs used by them.



(a) Solution $\mathcal{R}$ (b) Solution $\mathcal{Q}$

Figure 4.7: Two potential solutions to an example instance of the VRP. The nine continuous lines in both solutions represent the arcs that they have in common. In total, 18 different arcs are used, therefore $\varsigma_{\mathcal{R}\mathcal{Q}} = 9/18 = 0.5$

Denoting $y_{ij\mathcal{R}} = 1$ if arc (v_i, v_j) from customer v_i to customer v_j is traversed by any vehicle in solution $\mathcal{R}$, and 0 otherwise, the similarity $\varsigma_{\mathcal{R}\mathcal{Q}}$ between solutions $\mathcal{R}$ and $\mathcal{Q}$ is

$$\varsigma_{\mathcal{R}\mathcal{Q}} = \frac{\displaystyle\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}} y_{ij\mathcal{R}} \cdot y_{ij\mathcal{Q}}}{\displaystyle\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}} \operatorname{sign}\left(y_{ij\mathcal{R}} + y_{ij\mathcal{Q}}\right)} \;, \tag{4.4}$$

in which the term in the sum in the numerator will only equal 1 if arc (v_i, v_j) is used by both solutions, while that in the denominator will equal 1 if either solution uses it. Note that arcs (v_i, v_j) and (v_j, v_i) are considered to be different, even if the cost of traversing them is the same, since we are interested in measuring solution similarity on the solution space and not in the objective space. Hence, if solutions $\mathcal{R}$ and $\mathcal{Q}$ are the same, that is if they use the same arcs, $\varsigma_{\mathcal{R}\mathcal{Q}} = 1$, while if they are two completely different solutions with no arc in common, $\varsigma_{\mathcal{R}\mathcal{Q}} = 0$. Figure 4.7 shows two potential solutions to an example instance of the VRP, where the nine continuous lines in both solutions represent the arcs they have in common. Solution $\mathcal{R}$ uses four additional arcs, while solution $\mathcal{Q}$ uses five more. In total, 18 different arcs are used, therefore $\varsigma_{\mathcal{R}\mathcal{Q}} = 9/18 = 0.5$.

Algorithm 4.2: ComputeJaccardsSimilarity($\mathcal{R}$, $\mathcal{Q}$)

Input: Solutions $\mathcal{R}$ and $\mathcal{Q}$ to the VRP
Output: Jaccard similarity $\varsigma_{\mathcal{R}\mathcal{Q}}$ between solutions $\mathcal{R}$ and $\mathcal{Q}$
1: $E \leftarrow \emptyset$
2: **for all** arc (v_i, v_j) in $\mathcal{R}$ **do**
3: $\quad E \leftarrow E \cup \{(v_i, v_j)\}$
4: **end for**
5: $shared \leftarrow 0$
6: $total \leftarrow |E|$
7: **for all** arc (v_i, v_j) in $\mathcal{Q}$ **do**
8: $\quad$ **if** $(v_i, v_j) \in E$ **then**
9: $\quad\quad shared \leftarrow shared + 1$
10: $\quad$ **else**
11: $\quad\quad total \leftarrow total + 1$
12: $\quad$ **end if**
13: **end for**
14: **return** $shared/total$

Algorithm 4.2 is used to compute the Jaccard similarity between solutions $\mathcal{R}$ and $\mathcal{Q}$. The two loops in lines 2–4 and 7–13 are executed at most $2N$ times, because there can be a maximum of N routes, i.e. one route per customer. This means that the worst-case time complexity of this algorithm is $O(N)$.

For the purposes of the proposed algorithm, a measure of how similar a solution is to the rest of the evolutionary population is also required. If P is the population of solutions, the similarity $\sigma_{\mathcal{R}}$ of solution $\mathcal{R} \in P$ with the rest of the solutions in the population will be given by the average similarity of $\mathcal{R}$ with every other solution $\mathcal{Q}_i \in P$, that is

$$\sigma_{\mathcal{R}} = \frac{1}{popSize - 1} \sum_{\mathcal{Q}_i \,\in\, P \setminus \{\mathcal{R}\}} \varsigma_{\mathcal{R}\mathcal{Q}_i} \ . \tag{4.5}$$

Algorithm 4.3 computes this similarity. Line 7, where the similarity between solutions $\mathcal{R}_i$ and $\mathcal{R}_j$ is calculated, is executed $popSize(popSize - 1)/2$ times, i.e. $O(popSize^2)$. Thus the total time complexity of this algorithm is $O(NpopSize^2)$. Line 9 is introduced in order to avoid duplicated computations, i.e. $\varsigma_{\mathcal{R}_i\mathcal{R}_j} = \varsigma_{\mathcal{R}_j\mathcal{R}_i}$.

Algorithm 4.3: ComputePopulationSimilarity(P)

Input: Population $P = \{\mathcal{R}_1, \dots, \mathcal{R}_{popSize}\}$ of solutions to VRP
Output: List of similarities $S = [\sigma_1, \dots, \sigma_{popSize}]$ corresponding to solutions $\mathcal{R}_i \in P$

for $i \leftarrow 1$ to $popSize$ **do**
    $\sigma_i \leftarrow 0.0$
end for
$S \leftarrow [\,]$
for $i \leftarrow 1$ to $popSize$ **do**
    for $j \leftarrow i + 1$ to $popSize$ **do**
        $sim_{ij} \leftarrow \varsigma_{\mathcal{R}_i \mathcal{R}_j}$
        $\sigma_i \leftarrow \sigma_i + sim_{ij}$
        $\sigma_j \leftarrow \sigma_j + sim_{ij}$
    end for
    $\sigma_i \leftarrow \sigma_i \;/\; (popSize - 1)$
    $S[i] \leftarrow \sigma_i$
end for
return S

Finally, we define the *Jaccard diversity* $\delta_{\text{Jaccard}}(P)$ of solutions in P as one minus the average solution similarity, i.e.

$$\delta_{\text{Jaccard}}(P) = 1 - \frac{1}{popSize} \sum_{\mathcal{R}_i \in P} \sigma_{\mathcal{R}_i} \; . \tag{4.6}$$

These solution similarity and population diversity definitions are part of the main contributions of this research. It is important to mention that one of the advantages of the Jaccard similarity measure is that it does not depend on the solution encoding, since only the information about the arcs forming the routes is required and this is known independently of the representation being used. For the same reason, another advantage is that this measure can be used for any variant of the VRP, since solutions to these problems can be represented as a set of arcs. Moreover, the worst-case time complexity of the solution similarity algorithm is $O(N)$, and that of the population similarity is $O(NpopSize^2)$.

4.3.1.3 Parent selection

BiEA uses the tournament selection method as drGA does. It differs from most EAs in that, in addition to using fitness to select good parents, it also uses the similarity to maintain population diversity: the first of two parents is chosen on the basis of fitness and the second on the basis of similarity.

This is done with the intention of, after recombination, generating an offspring with similar fitness to that of the first parent, which could be located in a region of the search space that has not been properly explored.

4.3.1.4 Mutation

Here, BiEA discards the *merge* mutation operator used drGA, since the *insertion* operator was modified in order to allow a complete route to be taken and inserted into another. All other operators remained, however, they are not pre-assigned any weight, since it is desirable to have a better control over the parameters involved [63]. Instead, the following procedure is executed each time an offspring is going to be mutated. First, the split operator is performed with a probability equal to the inverse of the number of routes in the solution. Then, the solution is submitted to one of the inter-route operators: insertion or swap. The decision of which to apply is random. Finally, one of the intra-route operators, inversion or shift, is applied to the solution. The complete mutation process is shown in Algorithm 4.4.

4.3.1.5 Survival selection

After the mutation process, the algorithm evaluates the objective functions for each solution in the offspring population, and combines both parent and offspring populations to assign fitness. Those solutions having the highest fitness are taken to

Algorithm 4.4: MUTATE($\mathcal{R}$)

Input: Solution $\mathcal{R}$ to the VRPTW
Output: Mutated solution $\mathcal{R}'$
$\mathcal{R}' \leftarrow \mathcal{R}$
if RANDOMNUMBER($[0, 1]$) $\leq \frac{1}{|\mathcal{R}'|}$ **then**
    $\mathcal{R}' \leftarrow$ SPLIT($\mathcal{R}'$)
end if
if RANDOMNUMBER($[0, 1]$) ≤ 0.5 **then**
    $\mathcal{R}' \leftarrow$ INSERT($\mathcal{R}'$)
else
    $\mathcal{R}' \leftarrow$ SWAP($\mathcal{R}'$)
end if
if RANDOMNUMBER($[0, 1]$) ≤ 0.5 **then**
    $\mathcal{R}' \leftarrow$ INVERT($\mathcal{R}'$)
else
    $\mathcal{R}' \leftarrow$ SHIFT($\mathcal{R}'$)
end if
return $\mathcal{R}'$

Algorithm 4.5: SURVIVALSELECTION(P)

Input: Population $P' = P \cup Q = \{s_1, \ldots, s_{2\,popSize}\}$ submitted to the elitism process
Output: Elite solutions $P^* = \{s_1^*, \ldots, s_{popSize}^*\}$
$\mathcal{F} \leftarrow$ FASTNONDOMINATEDSORT(P')
$P^* \leftarrow \emptyset$
$k \leftarrow 1$
while $|P^*| + |\mathcal{F}_k| \leq popSize$ **do**
    COMPUTEPOPULATIONSIMILARITY($\mathcal{F}_k$)
    $P^* \leftarrow P^* \cup \mathcal{F}_k$
    $k \leftarrow k + 1$
end while
$L \leftarrow$ SORT($\mathcal{F}_k, \sigma$)   */* Sort front $\mathcal{F}_k$ according to similarity */*
$P^* \leftarrow P^* \cup \{L[1..(popSize - |P^*|)]\}$
return P^*

the next generation. If the last selected front is in conflict with the population size, unlike how drGA does, similarity is computed for the solutions belonging to that front. Solutions in this front which have the lowest evaluation for each objective function, i.e. extreme solutions, are assigned a similarity of zero with the aim of being selected. Then, the less common individuals are considered for the next iteration. This procedure is presented in Algorithm 4.5. Finally, similarity is computed for the entire population and the evolutionary process is repeated.

4.3.2 Experimental analysis

This time BiEA was only tested on Solomon's instances size $N = 100$ and was run 30 times for each instance. Solutions in the Pareto approximation were recorded after each execution. BiEA was set to minimise both the number of routes and the travel distance.

The modifications made to the algorithm, that is the mutation stage and the inclusion of the similarity measure, led to a new parameter tuning. However, some parameters only affected the speed of evolution and not the final performance, e.g. BiEA did not need the 200-individual population nor the 600-generation evolutionary process set in drGA. Hence, there was an initial series of experiments with the aim of obtaining the parameter values that resulted in improved results. The parameters for which the algorithm worked better were as follows:

$$\begin{array}{lllllllll} popSize & = & 100 & \quad Tsize & = & 10 & \quad \gamma & = & 0.9 \\ numGen & = & 500 & & & & \quad \mu & = & 0.1 \end{array}$$

The analysis of the results in this section has three objectives: (*i*) to compare the performance of BiEA set to use either edit distance or Jaccard similarity, (*ii*) to examine the effect of the similarity measure on performance, (*iii*) to compare results from single-objective and bi-objective algorithms, and (*iv*) to compare the performance of BiEA with that of other algorithms proposed in previous studies.

4.3.2.1 Edit distance v Jaccard similarity

In Section 2.5.4 was stated that the edit distance could be used to quantify the distance between solutions to the VRP, thus its use is considered here. With the purpose of evaluating the proposed Jaccard similarity against the edit distance, both were implemented in BiEA, allowing a comparison of the results from both techniques. As was stated earlier, BiEA chooses the second parent according to solution similarity. For this, we have at least two options when the set of $Tsize$ individuals has been selected to compete in the tournament: it can be chosen the individual that is the least similar to the whole population (W) or the least similar to the first parent (P). The algorithm which implements Jaccard similarity and selects the individual that is the least similar to the whole population will be identified as BiEA-JW, and as BiEA-JP the one which selects the least similar to the first parent. Analogously, the algorithms which use edit distance will be labelled as BiEA-EW and BiEA-EP.

With the aim of making a proper comparison, the outcome non-dominated solutions from each implementation of BiEA were evaluated using the three performance metrics reviewed in Section 2.3.2, i.e. coverage, $\mathsf{M_C}$ in (2.7), convergence, $\mathsf{M_D}$ in (2.9), and hypervolume, $\mathsf{M_H}$ in (2.10). In order to apply the coverage metric, for each given instance and ordered pair of implementations BiEA-X and BiEA-Y, $\mathsf{M_C}(\text{BiEA-X}_i, \text{BiEA-Y}_j)$, $\forall\ i, j = 1, \ldots, 30$, that is 900 $\mathsf{M_C}$ values, were computed. BiEA-X$_i$ refers to the outcome set from the i-th execution of BiEA-X. After these computations, the $\mathsf{M_C}$(BiEA-X$_i$, BiEA-Y$_j$) values were averaged ($\overline{\mathsf{M}}_\mathsf{C}$) over all the instances within each set category, and the resulting 900 values were collected together. These $\overline{\mathsf{M}}_\mathsf{C}$ values are presented in Figure 4.8 as box-and-whisker diagrams, which represent the distribution of the $\overline{\mathsf{M}}_\mathsf{C}$ values for each ordered pair of algorithms. Each cell, which range is 0 at the bottom and 1 at the top, contains six box-and-whisker plots, corresponding to categories C1, C2, R1, R2, RC1, and RC2 from left to right, referring to the average coverage of the algorithm in the corresponding column by the algorithm in the corresponding row. Each box indicates where the middle 50% of the data is located, on which the central mark is the median and the lower and upper edges are the first and third quartiles respectively. Dashed lines specify the most extreme data values that are not considered as outliers.

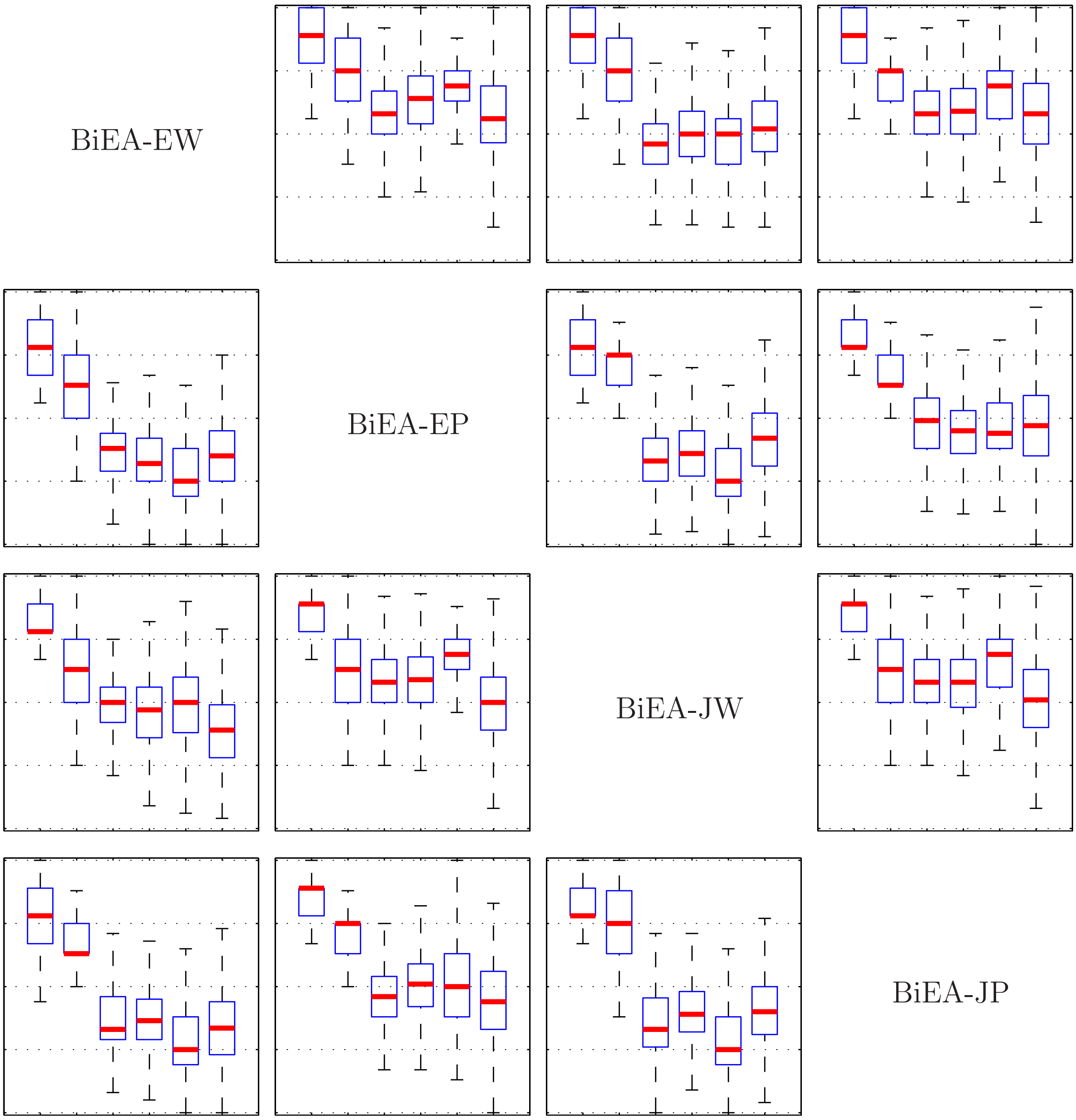


Figure 4.8: Box-and-whisker plots representing the distribution of the $\overline{\mathsf{M}}_{\mathsf{C}}$ values for each ordered pair of the implementations EW, EP, JW, and JP of BiEA.

We observe that, for categories C1 and C2, all four algorithms found the optimum solutions for almost all the instances, this is why the plots corresponding to these categories, the two leftmost boxes on each cell, show similar heights. For the remaining categories, we can observe that the height of the medians corresponding to the coverage by BiEA-EP and BiEA-JP, second and forth rows, is always lower than 0.5, and those by the implementations BiEA-EW and BiEA-JW, first and third rows, are higher than 0.5, with some exceptions in the cells between these two. This means

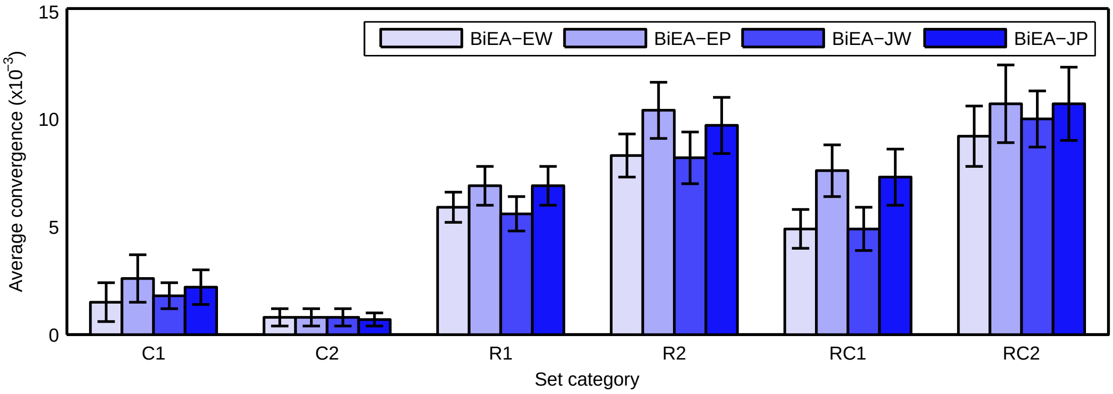


Figure 4.9: Bar plots representing the $\mathsf{M_D}$ values, averaged over instance category, for the results obtained by the implementations of BiEA with similarity and selection settings EW, EP, JW, and JP.

that the latter methods have a higher coverage of the former than the inverse cases. Finally, for the implementations BiEA-EW and BiEA-JW, the coverage between them appears to be similar.

In order to compare the algorithms using the convergence metric, it is necessary to have a reference set for every instance since the true Pareto fronts are not known. For each algorithm and instance, the overall non-dominated solutions were extracted from the 30 Pareto approximations. Then, a composite non-dominated reference set $\mathcal{R}$ was found using the overall non-dominated sets from the four algorithms. Afterwards, for each implementation BiEA-X, $\mathsf{M_D}(\text{BiEA-X}_i, \mathcal{R}), \forall\ i = 1, \ldots, 30$, were computed. Finally, the $\mathsf{M_D}$ values were normalised according to the distance from point $\mathbf{z} = (N, D^{\max})$ to the origin, where $\boldsymbol{z}$ corresponds to the solution with largest number of routes, i.e. number of customers N, and longest travel distance $D^{\max}$ equal to twice the sum of the distances of all customers from the depot. These normalised $\mathsf{M_D}$ values were grouped by the instances in each set category and the average $\overline{\mathsf{M}}_\mathsf{D}$ and standard error were calculated. Figure 4.9 presents these results as bar plots, which heights represent the averages. We can see that solutions from BiEA-EP and BiEA-JP, second and fourth bars on each group, are the farthest to the reference set in five of the six categories, while those from BiEA-EW and BiEA-JW, first and third bars, are the nearest and present similar performance.

On the other hand, the hypervolume metric has been utilised to compare only algorithms BIEA-EW and BiEA-JW, since they have been the most competitive methods regarding the other two quality indicators. Computing the hypervolume metric $\mathsf{M_H}$ requires an appropriate reference point $\boldsymbol{z}$ to be set. As was done for the normalisation of the convergence metric, each instance has an obvious maximal solution, that with largest number of routes N and longest travel distance $D^{\max}$, hence the reference point for each instance was set at $\mathbf{z} = (N, D^{\max})$. For each implementation BiEA-X, $\mathsf{M_H}(\text{BiEA-X}_i, \boldsymbol{z}), \forall\ i = 1, \ldots, 30$, were computed. Then, the $\mathsf{M_H}$ values were normalised according to the space defined between point $\boldsymbol{z}$ and the origin. These $\mathsf{M_H}$ values were grouped by the instances in each set category and the average $\overline{\mathsf{M}}_\mathsf{H}$ was computed.

The details of the hypervolume metric, along with those of the coverage and convergence metrics, comparing the results from BiEA-EW and BiEA-JW are presented in Table 4.3. The averages $\overline{\mathsf{M}}_\mathsf{C}$, $\overline{\mathsf{M}}_\mathsf{D}$ and $\overline{\mathsf{M}}_\mathsf{H}$ over the instances in each category are shown, along with the number of instances, in brackets, where the algorithm performs better than the other at the 95% significance level. The statistical significance for each instance was determined by applying a two-tailed t-test for two samples with unequal variance to the results of $\mathsf{M_C}(\text{BiEA-EW}_i, \text{BiEA-JW}_j)$ and $\mathsf{M_C}(\text{BiEA-JW}_j, \text{BiEA-EW}_i)$, $\mathsf{M_D}(\text{BiEA-EW}_i, \mathcal{R})$ and $\mathsf{M_D}(\text{BiEA-JW}_j, \mathcal{R})$, and $\mathsf{M_H}(\text{BiEA-EW}_i, \boldsymbol{z})$ and $\mathsf{M_H}(\text{BiEA-JW}_j, \boldsymbol{z})$, $\forall\ i, j = 1, \ldots, 30$. The result of the t-test specifies, with 95% of confidence, if the true means of the data do differ. In Table 4.3 we see that the averages of the coverage, convergence, and hypervolume metrics present a small difference between the results from BiEA-EW and BiEA-JW for all categories, which may suggest that both algorithms perform similarly. However, despite the narrow gap, BiEA-EW have a significantly better performance,

Algorithm	Metric	C1	C2	R1	R2	RC1	RC2
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.85 (5)	0.73 (5)	0.47 (4)	0.51 (5)	0.47 (2)	0.53 (5)
BiEA-EW	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-4})$	14.80 (0)	7.684 (0)	58.69 (1)	82.72 (0)	48.57 (0)	92.01 (2)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	76.95 (0)	87.30 (0)	66.39 (1)	78.46 (3)	69.69 (1)	80.44 (4)
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.81 (2)	0.64 (1)	0.48 (7)	0.47 (3)	0.49 (3)	0.39 (1)
BiEA-JW	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-4})$	17.63 (0)	8.099 (0)	56.44 (0)	82.05 (1)	48.65 (1)	99.61 (0)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	76.92 (0)	87.29 (0)	66.43 (1)	78.39 (0)	69.64 (0)	80.21 (0)

Table 4.3: Averages $\overline{\mathsf{M}}_{\mathsf{C}}$, $\overline{\mathsf{M}}_{\mathsf{D}}$ and $\overline{\mathsf{M}}_{\mathsf{H}}$ over instance category for the solutions obtained by BiEA-EW and BiEA-JW. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

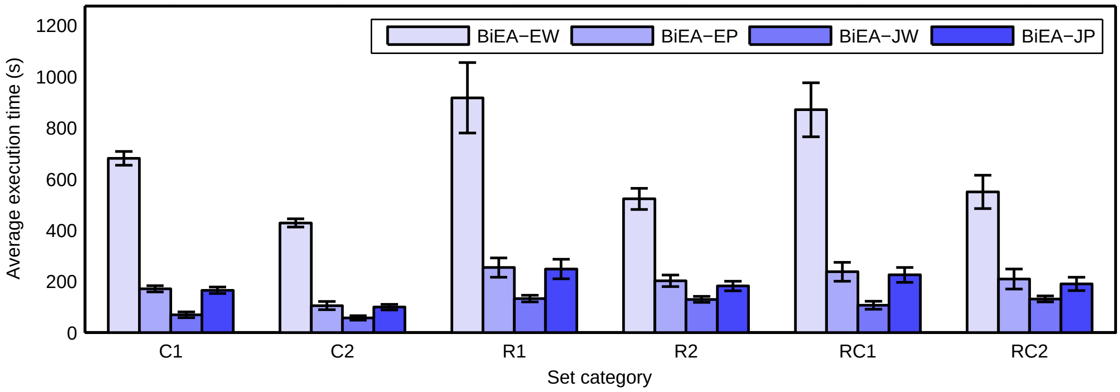


Figure 4.10: Execution time, averaged over instance categories, of the implementations of BiEA with similarity and selection settings EA, EP, JA, and JP.

regarding the coverage metric, for more instances in set categories C1, C2, R2, and RC2, while BiEA-JW for more instances in categories R1 and RC1. We can also observe that BiEA-EW delimits a significantly larger hypervolume for some instances in categories R2 and RC2.

Lastly, Figure 4.10 represents the average execution time, corresponding to each set category, of the four implementations of BiEA. It is clearly visible that BiEA-JW is the quickest method, while BiEA-EW is the slowest, the latter taking at least 300% more time in executing 500 generations than the former, and more than 500% on average. This is due to the fact that the edit distance has a quadratic time complexity, in contrast to the linear behaviour of the Jaccard similarity.

Based on the analysis of the three performance metrics, we conclude that implementations BiEA-EW and BiEA-JW have a better performance that the implementations BiEA-JP and BiEA-EP, since the former obtain better results for all three quality indicators. Considering execution time, BiEA-JW is the quickest algorithm, running, on average, 500% faster than BiEA-EW, the slowest. This last fact make us believe that the saving in time is worth the minimal overall differences in the performance metrics between algorithms BiEA-JW and BiEA-EW. However, both edit distance and Jaccard similarity will be tested later with the final algorithm in order to analyse if they present the same behaviour. Meanwhile, the following experiments and analysis will be made considering the results from BiEA set to use Jaccard similarity, and hereafter BiEA-JW will be simply called BiEA.

4.3.2.2 Effect of the similarity measure

The same series of experiments was carried out using BiEA without considering the similarity measure (BiEA-nS), in which the parent and survival selections took only fitness into account. In these experiments, the crossover probability γ was set to 1.0 with the aim of maximising diversity, and the mutation probability μ was set to zero in order not to interfere with the diversity control. The purpose of this analysis is two-fold: to compare the performance of BiEA with and without the similarity measure, and to determine whether the similarity measure is accomplishing the goal of diversifying the population.

Table 4.4 presents the results of the three performance metrics comparing the results from BiEA-nS and BiEA. The averages $\overline{\mathsf{M}}_\mathsf{C}$, $\overline{\mathsf{M}}_\mathsf{D}$ and $\overline{\mathsf{M}}_\mathsf{H}$, which were computed as before, over the instances in each category are shown, along with the number of instances, in brackets, where the algorithm performs significantly better than the other. As we can observe, BiEA outperforms BiEA-nS in all six categories, since $\overline{\mathsf{M}}_\mathsf{C}(\text{BiEA}, \text{BiEA-nS}) > \overline{\mathsf{M}}_\mathsf{C}(\text{BiEA-nS}, \text{BiEA})$, $\overline{\mathsf{M}}_\mathsf{D}(\text{BiEA}, \mathcal{R}) < \overline{\mathsf{M}}_\mathsf{H}(\text{BiEA-nS}, \mathcal{R})$,

Algorithm	Metric	C1	C2	R1	R2	RC1	RC2
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.23 (0)	0.39 (0)	0.00 (0)	0.11 (0)	0.00 (0)	0.14 (0)
BiEA-nS	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-2})$	1.99 (0)	0.58 (0)	2.37 (0)	1.44 (0)	2.40 (0)	1.38 (0)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	75.22 (0)	86.73 (0)	64.29 (0)	77.21 (0)	67.30 (0)	79.12 (0)
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.96 (6)	0.77 (7)	0.99 (7)	0.87 (11)	0.99 (6)	0.83 (8)
BiEA	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-2})$	0.28 (8)	0.34 (6)	0.53 (12)	0.68 (11)	0.54 (8)	0.82 (7)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	76.76 (8)	86.96 (7)	66.34 (12)	77.97 (11)	69.54 (8)	79.92 (7)

Table 4.4: Averages $\overline{\mathsf{M}}_{\mathsf{C}}$, $\overline{\mathsf{M}}_{\mathsf{D}}$ and $\overline{\mathsf{M}}_{\mathsf{H}}$ over instance category for the solutions obtained by BiEA-nS and BiEA. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

and $\overline{\mathsf{M}}_{\mathsf{H}}(\text{BiEA}, \boldsymbol{z}) > \overline{\mathsf{M}}_{\mathsf{H}}(\text{BiEA-nS}, \boldsymbol{z})$. Furthermore, BiEA performs significantly better than BiEA-nS for the vast majority of the instances. The difference in these results is the effect of including the similarity measure, since one of the parents is selected to be not so similar to the rest of the population, thus looking for solutions in other regions of the search space. This selection could result, after recombination, in an offspring with good quality and different from current individuals.

The population diversity preserved by both algorithms, averaged over the instances in each set category, along with the average hypervolume difference between both algorithms, are shown in Figure 4.11, which contains six plots, each for the instance category printed at the top-right. In this case, the population diversity δ_{Jaccard} in (4.6) was computed.

It is noticeable that BiEA preserves a higher diversity than BiEA-nS for sets R1, R2, RC1, and RC2. Moreover, the lines corresponding to BiEA present a more gentle slope in all cases, except for C2. This behaviour suggests that BiEA is having a wider exploration of the search space, which is due to the utilisation of the Jaccard similarity. On the other hand, we can see that the hypervolume difference is negative for the first evolutionary generations in all categories, which means that BiEA-nS delimits a wider objective space due to it converged to better solutions than those found by BiEA. However, after some generations, the hypervolume difference

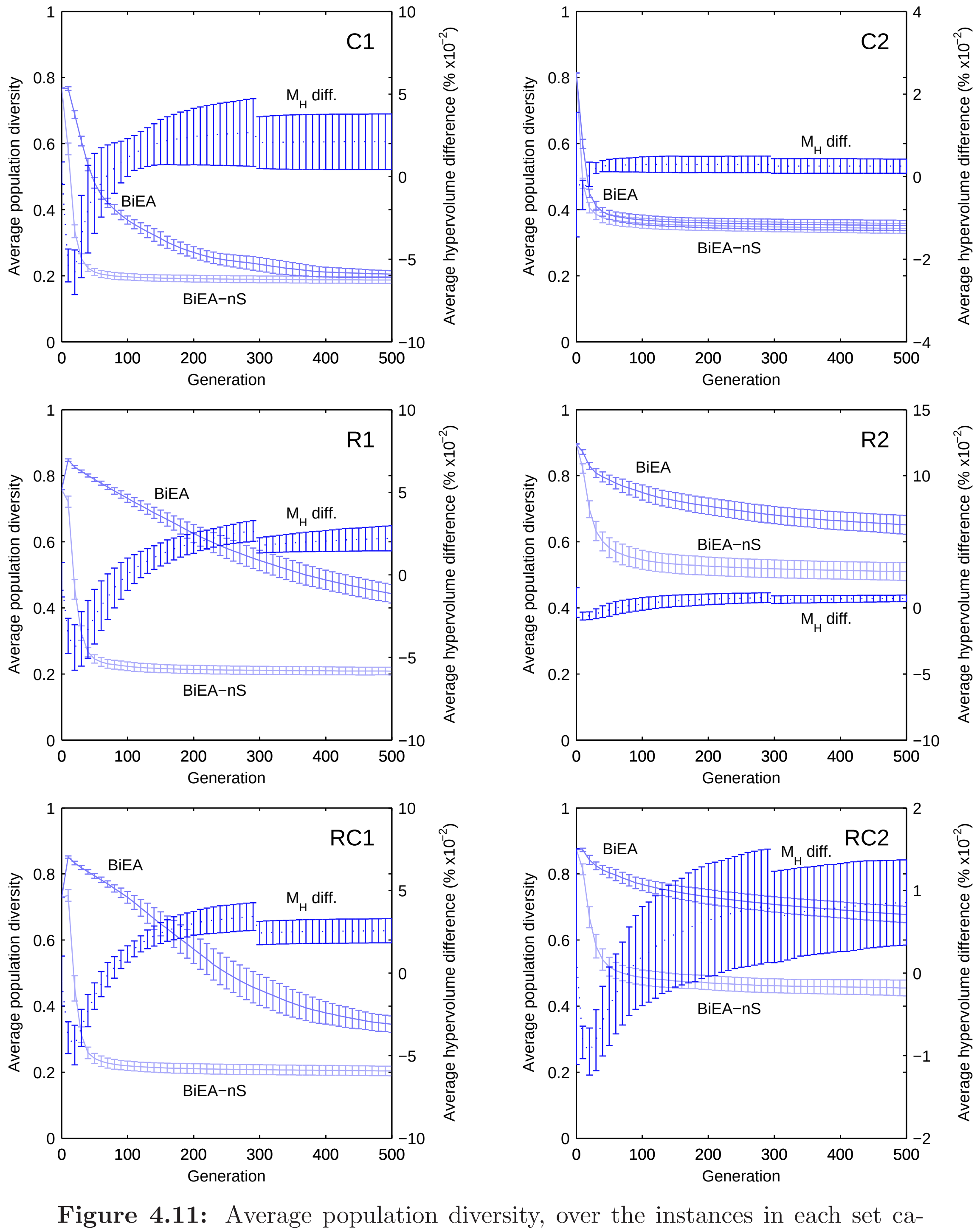


Figure 4.11: Average population diversity, over the instances in each set category, preserved by BiEA-nS and BiEA, and average hypervolume difference between both algorithms.

decreased and became positive, which means BiEA had outperformed BiEA-nS by finding better solutions because of the wider exploration of the search space.

Algorithm	C1	C2	R1	R2	RC1	RC2
BiEA-nS	1.08 (0)	1.05 (0)	1.31 (0)	1.65 (0)	1.35 (0)	1.93 (1)
BiEA	1.05 (0)	1.04 (0)	1.47 (2)	1.73 (2)	1.70 (4)	2.02 (2)

Table 4.5: Size of the Pareto approximations, averaged over instance categories, obtained with BiEA-nS and BiEA. Shown in brackets are the number of instances for which the non-dominated sets are significantly larger than the other approach.

Finally, the size of the Pareto approximations, averaged over instance categories, are shown in Table 4.5, and in brackets are the number of instances where the difference in the sizes is significant. We see that, although the averages present a small difference for all categories, the non-dominated sets found by BiEA are significantly larger than those from BiEA-nS for some of the instances in categories R1, R2, RC1, and RC2, for which population diversity was also higher.

It is important to recall that objectives in categories C1 and C2 are not really in conflict, consequently the Pareto front does not contain multiple solutions and diversity is not really needed in these cases. Additionally, due to the customers locations are clustered, the probability to obtain good solutions, in which customers from different clusters are serviced by one route, is low. These are the reasons why the population diversity preserved by BiEA-nS and BiEA experienced an early drastic loss and ended up being low. We would expect, then, to have the same behaviour with all algorithms for instances in these categories.

4.3.2.3 Single-objective v bi-objective optimisation

This section compares the results from BiEA with those from a single-objective EA, namely the version of BiEA which only minimises one of the two objective functions. For simplicity, the one that minimises the number of routes will be called BiEA-R and the one that minimises the travel distance will be labelled as BiEA-D. This comparison is performed to determine whether better results can be achieved by

Algorithm	C1	C2	R1	R2	RC1	RC2	Total
BiEA-R	10.43	3.07	13.08	3.14	12.91	3.60	442.00
	1685.22	898.33	1550.45	1448.03	1742.10	1769.78	84982.41
BiEA-D	10.05	3.01	13.52	4.00	13.51	4.81	467.00
	908.21	601.42	1273.83	954.24	1464.37	1111.69	59376.27
BiEA ($\min R$)	10.00	3.00	12.51	3.09	12.23	3.46	423.60
	838.63	594.67	1215.41	983.13	1376.60	1194.52	58273.34
BiEA ($\min D$)	10.00	3.00	12.91	3.84	12.66	4.50	448.50
	838.63	594.67	1206.49	953.60	1370.12	1118.91	57184.87
% diff. R	4.30	2.33	4.56	1.62	5.56	4.05	4.34
% diff. D	8.30	1.14	5.58	0.07	6.88	-0.65	3.83

Table 4.6: Number of routes and travel distance, averaged over categories, for the best solutions found by BiEA-R, BiEA-D, and BiEA.

considering VRPTW as a multi-objective problem and to known if BiEA is having a performance at least as good as a single-objective EA.

In Table 4.6 are presented the average best results, where the best result found for each instance is averaged over all iterations, and then averaged over the instance set. For each algorithm and set category, the average number of routes (upper figure) and the average travel distance (lower figure) are shown. The last column presents the total average number of routes and the total average travel distance for all 56 instances. For BiEA, the solutions with the smallest number of routes ($\min R$) and the shortest travel distance ($\min D$) were considered to compute the average. The last two rows indicate, for each instance category, the percentage difference in the number of routes between the results from BiEA and BIEA-R (% diff. R), and in the travel distance (% diff. D) between the results from BiEA and BiEA-D.

Let us analyse first the solutions with the smallest number of routes. We can see that the average number of routes from BiEA-R is lower than that from BiEA-D in four of the six categories and in total. BiEA ($\min R$) obtained solutions with smaller number of routes and shorter travel distance than BiEA-R in all categories and in total, while achieving better results for both objectives than BiEA-D in all categories, except for R2 and RC2, and in total. With respect to the solutions with the shortest travel distance, solutions from BiEA-D always have shorter travel distance than those from BiEA-R. With the exception of category RC2, BiEA (min D) achieved solutions with shorter travel distance than those from BiEA-D for the remaining categories and in total, while performing better for both objectives than BiEA-R in four of the categories. These results indicate that considering VRPTW as a bi-objective problem leads to find better solutions. This is in part due to BiEA-R and BiEA-D are only minimising one objective, the number of routes and the travel distance respectively, and do not consider an additional objective that could help the algorithm escape from sub-optimum regions of the search space.

4.3.2.4 Comparison with previous studies

Results from BiEA are shown in Table 4.7, where they are compared with the best average results from Section 3.2.3 and with those from the multi-objective studies in Section 3.4.1. This table has the same format as that of Table 4.2, with the difference that now the approach of Ombuki et al. [182], that of Ghoseiri and Ghannadpour [111] and BiEA have two rows, one considering the solutions with the smallest number of routes (min R) and other for the solutions with the shortest travel distance (min D). For each instance, the solution with the smallest number of routes and that with the shortest travel distance were taken and the average over the instances in each set category was computed. The last two rows indicate, for each instance set, the percentage difference between the results from BiEA and the best average results, regarding the number of routes (% diff. R) and the travel distance (% diff. D).

With respect to the number of routes, we can see that for instance categories C1 and C2, BiEA (min R) obtained the best-known results. For categories R1, RC1, RC2, and in total, the results are between 2.1% and 4.4% above the best average results, and for category R2 they are nearly 13% higher. Regarding the travel distance,

Author	C1	C2	R1	R2	RC1	RC2	Total
min R	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.86	1212.73	952.67	1384.30	1108.52	57192.00
min D	10.00	3.00	11.92	5.36	12.88	6.25	486.00
	828.38	589.86	1121.10	878.41	1341.67	1004.20	54779.02
Ombuki et al. [182]	10.00	3.00	12.67	3.09	12.38	3.50	427.00
(min R)	828.48	590.60	1212.58	956.73	1379.87	1148.66	57484.35
Ombuki et al. [182]	10.00	3.00	13.17	4.55	13.00	5.63	471.00
(min D)	828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
Tan et al. [228]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ghoseiri and Ghannadpour	10.00	3.00	12.92	3.45	12.75	3.75	439.00
[111] (min R)	828.38	591.49	1228.60	1033.53	1392.09	1162.40	58735.22
Ghoseiri and Ghannadpour	10.00	3.00	13.50	3.82	13.25	4.00	456.00
[111] (min D)	828.38	591.49	1217.03	1049.62	1384.3	1157.41	58671.12
BiEA (min R)	10.00	3.00	12.17	3.09	12.00	3.38	417.00
	828.45	589.86	1208.67	947.53	1353.71	1146.78	57105.67
BiEA (min D)	10.00	3.00	12.75	4.00	12.50	4.63	448.00
	828.45	589.86	1184.71	915.48	1346.12	1070.85	55797.43
% diff. R	0.00	0.00	2.16	13.22	4.35	3.85	2.96
% diff. D	0.01	0.00	5.67	4.22	0.33	6.64	1.86

Table 4.7: Number of routes and travel distance, averaged over categories, for the best solutions found by previous studies and BiEA.

BiEA (min D) obtained the best-known results for category C2, and for category C1 it is slightly 0.01% above them. In respect to the remaining categories, results from BiEA are no more than 6.7% higher than the best from previous studies. These results indicate an improvement over drGA when they are compared with the best of the previous results.

If we refer to the results obtained by the reviewed studies in Section 3.2.3 (Tables 3.8 and 3.9 in pages 86 and 87), we observe that BiEA achieved improved results when compared with some of them, e.g. smaller number of routes and shorter travel distance with respect to the results from Potvin et al. [194], Potvin and Bengio [192], Cordone and Wolfler Calvo [50], and Zhu [256].

On the other hand, considering exclusively the multi-objective studies, we see in Table 4.7 that BiEA obtained better results for both objectives than the approach of Ombuki et al. [182] and that of Ghoseiri and Ghannadpour [111], regarding solutions with the smallest number of routes ($\min R$). In respect to solutions with the shortest travel distance ($\min D$), BiEA improved the approach of Ombuki et al. [182] and that of Ghoseiri and Ghannadpour [111] in categories C1, C2, R1 and RC1, and in total, and achieved better results than that of Tan et al. [228] in categories C1, C2, and R1.

The study regarding BiEA described up to here was accepted for publication and presentation in two top conferences, the 5th International Conference on Evolutionary Multi-criterion Optimization (EMO'09) [100] and the Genetic and Evolutionary Computation Conference 2009 (GECCO 2009) [98, 101].

4.3.2.5 Influence of the mutation operators on performance

In order to know what the influence of the improved mutation operators was, two additional series of experiments were performed. Firstly, BiEA was set to use all possible combinations of four out of the five mutation operators, that is, five different settings, each excluding a different operation. Secondly, the algorithm was set to use only one mutation operator, i.e. five additional settings.

The outcome set of non-dominated solutions from each algorithm was evaluated using the coverage and convergence performance metrics. For each algorithm and instance, the overall non-dominated solutions were extracted from the 30 Pareto approximations. Then, a composite non-dominated reference set $\mathcal{R}$ was found using the overall non-dominated sets from the five algorithms ran for each series of experiments. Afterwards, for each implementation BiEA-X, where X is related to the different mutation operator settings, $\mathsf{M_C}(\text{BiEA-X}_i, \mathcal{R})$ and $\mathsf{M_D}(\text{BiEA-X}_i, \mathcal{R}), \forall\ i =$

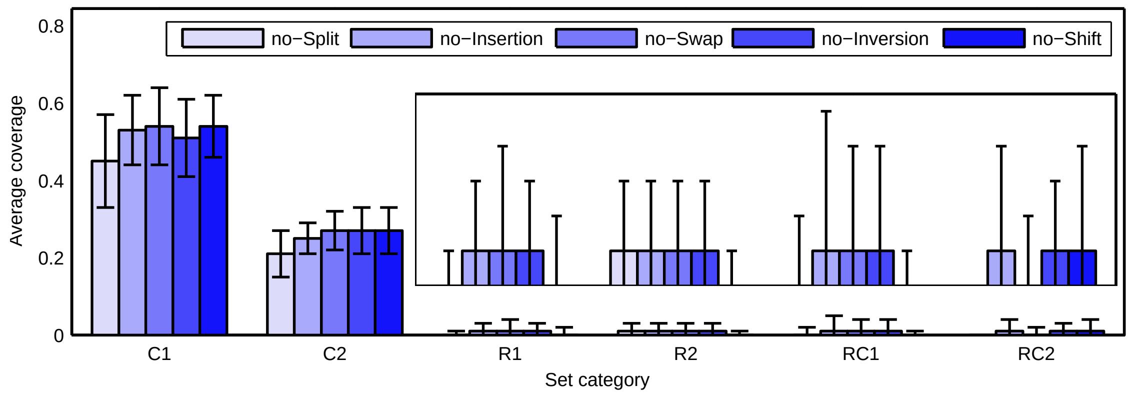


Figure 4.12: The $\mathsf{M_C}$ values, averaged over instance category, for the results obtained by BiEA set to exclude one of the mutation operators.

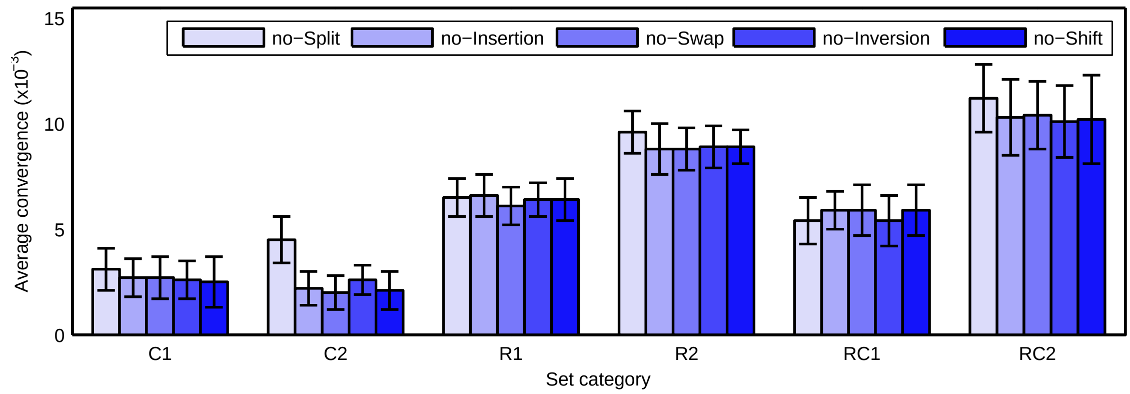


Figure 4.13: The $\mathsf{M_D}$ values, averaged over instance category, for the results obtained by BiEA set to exclude one of the mutation operators.

$1, \ldots, 30$, were computed. These $\mathsf{M_C}$ and $\mathsf{M_D}$ values were grouped by the instances in each set category and the averages $\overline{\mathsf{M}}_\mathsf{C}$ and $\overline{\mathsf{M}}_\mathsf{D}$, and corresponding standard errors, were calculated.

Figures 4.12 and 4.13 show, respectively, the $\overline{\mathsf{M}}_\mathsf{C}$ and $\overline{\mathsf{M}}_\mathsf{D}$ values for the first series of experiments, that is for the algorithms set to exclude one of the mutation operators. We observe in Figure 4.12 that the bar corresponding to the algorithm that was set to exclude the split mutation operator, the left-most bar in each group, is always lower than the others, except for category R2, which means that this algorithm has, on average, the narrowest coverage of the reference set $\mathcal{R}$. All other algorithms obtained similar coverage metric values. Moreover, we see in Figure 4.13 that the

Setting	no-Split	no-Insertion	no-Swap	no-Inversion	no-Shift
no-Split		2	2	3	3
no-Insertion	10		1	1	1
no-Swap	10	2		1	2
no-Inversion	13	4	4		0
no-Shift	11	3	1	2	

Table 4.8: Number of instances for which there is a significant difference, between each pair of algorithms set to exclude one of the mutation operators, in the results from the coverage, upper diagonal, and convergence, lower diagonal, performance metrics.

bar corresponding to this algorithm is always higher than the other bars, except for category RC1, which indicates that the non-dominated solutions from this algorithm are, on average, the farthest to $\mathcal{R}$. The solutions from the rest of the algorithms are equally distant from the reference set.

Additionally, Table 4.8 presents the number of instances for which there is a significant difference, between each pair of algorithms, in the coverage metric, shown in the upper diagonal, and in the convergence metric, shown in the lower diagonal. We see here that both the row and column corresponding to the algorithm excluding the split operator have the largest number of instances with significant differences. This corroborates that setting BiEA to exclude the split mutation operator results in low-quality non-dominated solutions.

On the other hand, Figures 4.14 and 4.15 present the $\overline{\mathsf{M}}_{\mathsf{C}}$ and $\overline{\mathsf{M}}_{\mathsf{D}}$ values for the second series of experiments, i.e. for the algorithms set to perform only one of the mutation operators. Figure 4.14 shows that the algorithm that was set to execute only the split mutation operator has the widest coverage of the reference set, since the bar corresponding to this algorithm, the left-most bar in each group, is the highest in all six categories. Furthermore, Figure 4.15 indicates that the non-dominated solutions from this algorithm are, on average, the closest to $\mathcal{R}$, since the bar corresponding to this algorithm is the smallest in all six categories.

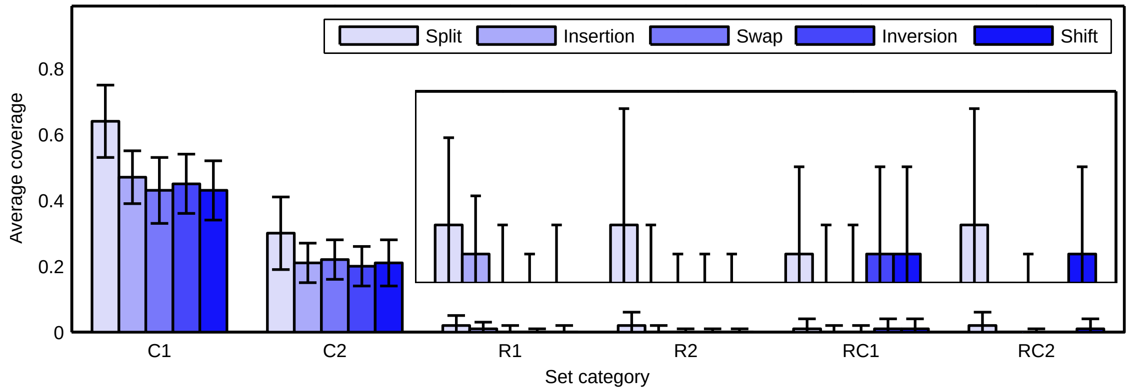


Figure 4.14: The $\mathsf{M_C}$ values, averaged over instance category, for the results obtained by BiEA excluding one of the mutation operators.

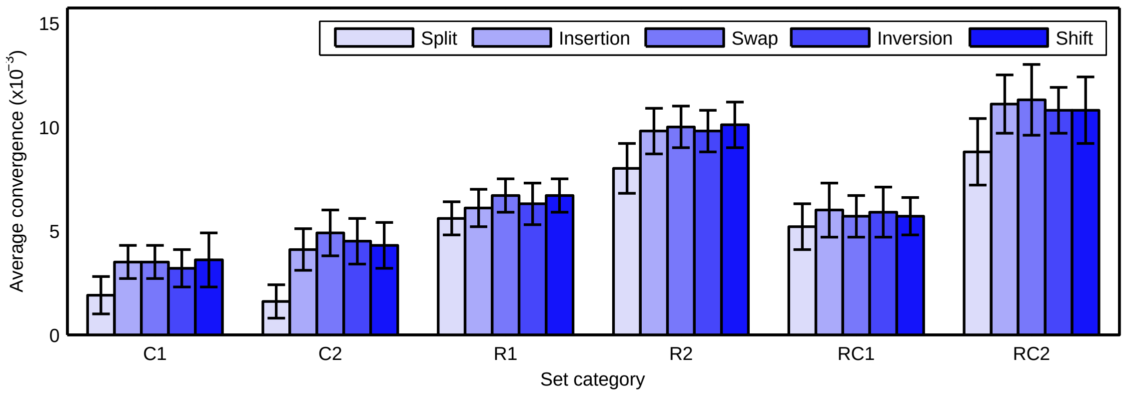


Figure 4.15: The $\mathsf{M_D}$ values, averaged over instance category, for the results obtained by BiEA excluding one of the mutation operators.

Additionally, Table 4.9 shows the number of instances for which there is a significant difference, between each pair of algorithms, in the results from the coverage, upper diagonal, and convergence, lower diagonal, performance metrics. We observe that the row and column corresponding to the algorithm set to execute only the split operator have the largest number of instances with significant difference. This confirms that setting BiEA to perform only the split mutation operator results in high-quality non-dominated solutions.

This analysis indicates that split is the mutation operator that has the strongest influence on BiEA's performance, which suggests and could lead to the improvement of the mutation process. As a reminder, what this operator does, is to split a route

Setting	Split	Insertion	Swap	Inversion	Shift
Split		5	6	5	5
Insertion	22		0	0	0
Swap	26	7		1	0
Inversion	24	2	3		1
Shift	25	4	3	5	

Table 4.9: Number of instances for which there is a significant difference, between each pair of algorithms set to perform only one mutation operator, in the results from the coverage, upper diagonal, and convergence, lower diagonal, performance metrics.

and attempts to reallocate the customers assigned to the smallest sub-route to the other existing routes. Thus, the next evident enhancement to BiEA is the mutation process.

4.4 Summary

The first stage of the preliminary study proposed a multi-objective density-restricted Genetic Algorithm (drGA), for which the main feature was the control of the number of distinct solutions in the population. It considered a recombination operator which was designed for preserving routes from both parents and a mutation process with six operators which were applied according to a weight they were pre-assigned. Three of these can be categorised as inter-route operators, modifying the assignment of customers to routes, and the other three as intra-route, modifying the sequence of customers within a route. If, after the crossover and mutation stages, the density of any solution had grown to more than the maximum allowed, further individuals were removed from the population until the density rectified. Fittest solutions were taken to the next generations and in case the population size was compromised, solutions with the shortest travel distance were preferred. Results from this algorithm uncovered the need for a better method to preserve population diversity, which not only aimed at maintaining different solutions in the population, but also at contemplating solutions in different regions in the search space.

In order to overcome the problem of getting stuck in sub-optimum solutions and to have a better exploration of the search space, a more sophisticated and efficient technique to diversify the population was proposed as part of a Bi-objective Evolutionary Algorithm (BiEA). Experiments with several possibilities led to the use of a particular form of the Jaccard's similarity coefficient, which measures, in a straightforward manner, how similar two sets are as the ratio of the number of elements they have in common to the number of total elements in both sets. Some of the advantages of this Jaccard similarity are that it does not depend on the solution encoding, since it uses information of the actual routes, and, for the same reason, this measure can be applied on not only VRPTW, but on any variant of the VRP. Furthermore, the worst-case time complexity of this algorithm is $O(N)$.

BiEA differs from drGA in several respects: the solution encoding, the parent selection, the mutation process and the elitism procedure. Parent and survival selection use the information provided by the Jaccard similarity, in the sense that less similar individuals are preferred. The mutation process now involves only five operators, two in the inter-route category and three in the intra-route. Results from BiEA showed a significant improvement over drGA when they are compared with those from previous studies, and specifically from the multi-objective approaches. It was also shown, by means of applying three multi-objective performance metrics to the non-dominated solutions, that these results are similar to those obtained when edit distance is used instead of Jaccard similarity, though the difference in the execution time is *huge*, since the algorithm set to use edit distance takes, on average, 500% longer.

Finally, an analysis of the mutation process revealed that the split mutation operator is having the strongest influence on BiEA's performance, which suggests that the process and operations can be enhanced.

Chapter 5

Multi-Objective Evolutionary Algorithm for solving VRPs

This chapter focuses on the description of the final algorithm resulting from the preliminary study introduced in the previous chapter. Specifically, an improved mutation process, which involves new operators, was incorporated in the Multi-Objective Evolutionary Algorithm (MOEA). Additionally, the application of the new algorithm to two VRPs is presented: the VRP with Time Windows (VRPTW) and the Capacitated VRP (CVRP). The former is solved by optimising, first, the number of routes and the travel distance, and then, an additional objective is considered, namely the delivery time. The latter is solved by optimising the first two objectives only, since travel distance and delivery time are never in conflict in this case.

5.1 Multi-Objective Evolutionary Algorithm

After analysing the results from BiEA, it was necessary to know what the effect of the mutation operators on performance was. An analysis was conducted and BiEA was tested with different mutation operator settings. The result indicated that not all the operators were contributing substantially to the good performance of the algorithm. Consequently, the mutation process and operators were altered, and all other stages in the new Multi-Objective Evolutionary Algorithm (MOEA) remained unchanged. The modified mutation procedure and operations are now presented in detail.

5.1.1 Mutation

The mutation process in the improved algorithm now involves the use of three basic functions and three mutation operators.

5.1.1.1 Basic functions

The three basic functions considered in the mutation stage of the new MOEA are the following.

SELECTROUTE($\mathcal{R}$) This is a stochastic process that selects a route r_k^* from solution $\mathcal{R} = \{r_1, \dots, r_K\}$ according to the proportion of the travel distance to the number of customers related to route r_k, i.e. routes with a larger travel distance and fewer customers are more likely to be selected. This function is implemented using the Roulette Wheel Selection (RWS) presented in Section 2.2.3.4, where function

$$\varphi(r_k) = \frac{d(r_k)}{N_k} \tag{5.1}$$

is the ratio of the travel distance $d(r_k)$ to the number of customers N_k associated with route r_k. This function applies RWS and returns the selected route r_k^*.

SELECTCUSTOMER(r_k) This function stochastically chooses customer v_i^* from route $r_k = \langle v_1, \dots, v_{N_k} \rangle$ according to the lengths $d_{i-1\,i}$ and $d_{i\,i+1}$ of its inbound and outbound arcs, i.e. $(i-1, i)$ and $(i, i+1)$, respectively. That is, customers with longer associated travel distance are more likely to be chosen. Special cases exist for

Algorithm 5.1: INSERTCUSTOMER(v_i, r_k)

Input: Customer v_i to be inserted into route $r_k = \langle v_1, \ldots, v_{N_k} \rangle$
Output: **true** if customer was inserted, **false** otherwise

/ Verify whether capacity constraint will be satisfied if v_i is inserted in r_k */*
if $q(r_k) + q_i \leq Q$ **then**
    $minDist \leftarrow \infty$
    $p \leftarrow$ **null**   */* Insertion position where the lowest distance is obtained */*
    for $j \leftarrow 1$ **to** $N_k + 1$ **do**
        if Inserting customer v_i between v_{j-1} and v_j would satisfy time constraint **then**
            Insert customer v_i between v_{j-1} and v_j
            if $d(r_k) < minDist$ **then**
                $minDist \leftarrow r(r_k)$
                $p \leftarrow j$
            end if
            Remove v_i from r_k
        end if
    end for
    if $p \neq$ **null then**
        Insert customer v_i between v_{p-1} and v_p
        return true
    end if
end if
return false

the first and last customers in a route, where only the outbound and inbound arcs, respectively, are taken into account. This function is also based on the utilisation of RWS, where function

$$\varphi(v_i) = \frac{d_{i-1\,i} + d_{i\,i+1}}{d(r_k)} \tag{5.2}$$

is the ratio of the travel distance associated with customer v_i to the travel distance related to route r_k. This function returns the customer v_i^* chosen by RWS.

INSERTCUSTOMER(v_i, r_k) This procedure deterministically inserts customer v_i into route $r_k = \langle v_1, \ldots, v_{N_k} \rangle$ in the position p where the lowest travel distance is obtained. It tests all feasible insertion positions and each time a new minimum travel distance is found, this is recorded as well as the insertion position p where it was obtained. The detailed function is shown in Algorithm 5.1. If v_i can not be inserted due to the capacity or time constraints, **false** is returned, **true** otherwise.

Algorithm 5.2: $\textsc{Reallocate}(r_k, \mathcal{R})$

Input: Route $r_k = \langle v_1, \ldots, v_{N_k} \rangle \in \mathcal{R}$ from which customers are going to be reallocated

$v_i \leftarrow \textsc{SelectCustomer}(r_k)$
$v_j \leftarrow \textsc{SelectCustomer}(r_k)$
Remove $q = \langle\langle v_i, \ldots, v_j \rangle\rangle$ from r_k
for all $v_i \in q$ **do**
  inserted $\leftarrow$ **false**
  for all $r_j \in \mathcal{R} \setminus \{r_k\}$ **and while** *inserted* $=$ **false do**
    inserted $\leftarrow \textsc{InsertCustomer}(v_i, r_j)$
  end for
  if *inserted* $=$ **false then**
    $\textsc{InsertCustomer}(v_i, r_k)$
  end if
end for

5.1.1.2 Mutation operators

The two basic functions $\textsc{SelectCustomer}()$ and $\textsc{InsertCustomer}()$ described above are used by the following new mutation operators.

$\textsc{Reallocate}(r_k, \mathcal{R})$ This operator removes a sequence of customers from route $r_k = \langle v_1, \ldots, v_{N_k} \rangle$ and allocates them to any other route $r_j \in \mathcal{R}$. First, function $\textsc{SelectCustomer}(r_k)$ is used to choose two customers v_i and v_j from route r_k. These are removed from the route, along with the sequence $\langle\langle v_{i+1}, \ldots, v_{j-1} \rangle\rangle$ of all those customers in between them, and collected in $q = \langle\langle v_i, v_{i+1}, \ldots, v_{j-1}, v_j \rangle\rangle$. Then, $\textsc{InsertCustomer}(v_i)$ attempts to reallocate all removed customers $v_i \in q$ into any of the existing routes $r_j \in \mathcal{R}$. If customer $v_i \in q$ could not be reallocated, it is reinserted into route r_k. This operation is shown in Algorithm 5.2.

An example of this operator is illustrated in Figure 5.1. In this example, customers 5 and 8 were selected from the route on the right and the sequence $\langle\langle 5, 6, 8 \rangle\rangle$ was removed from the route. Then, the operator attempted to insert those customers into the route on the left, however, due to capacity or time constraints, the insertion of customer 5 was the only successful move. Customers 6 and 8 were reinserted into their original route.

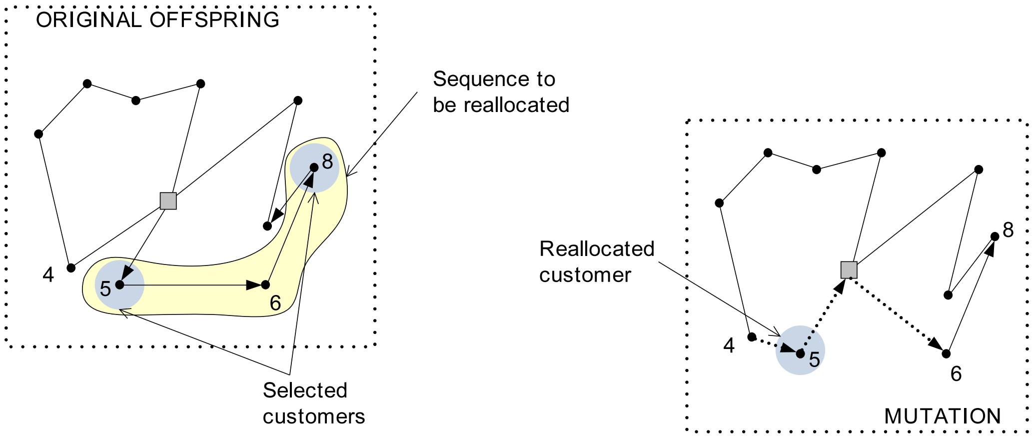


Figure 5.1: Example of the reallocate mutation operator: It selects customers 5 and 8, and removes the sequence $\langle\langle 5, 6, 8\rangle\rangle$ from the route on the right. Then, attempts to reinsert them into the route on the left, however, only customer 5 is successfully inserted. Customers 6 and 8 are returned to their original route.

Exchange(r_k, r_l) In this operation, two sequences of customers q_k and q_l extracted from routes r_k and r_l, respectively, are swapped. First, SelectCustomer(r_k) chooses two customers v_i and v_j from route r_k. These customers and the sequence of customers $\langle\langle v_{i+1}, \ldots, v_{j-1}\rangle\rangle$ in the middle of them are then removed from r_k and collected in $q_k = \langle\langle v_i, v_{i+1}, \ldots, v_{j-1}, v_j\rangle\rangle$. The same procedure is performed for route r_l. Then function InsertCustomer(v_i, r_l) attempts to reallocate all customers $v_i \in q_k$ into route r_l. If customer v_i cannot be inserted into the other route due to the capacity or time constraints, InsertCustomer(v_i, r_j) is applied to attempt to insert them into the other existing routes $r_j \in \mathcal{R}$. The same process is repeated for all customers $v_i \in q_l$ and route r_k. Algorithm 5.3 presents this operator.

An example is shown in Figure 5.2, where customers 5 and 8 are selected from the route on the right and customers 10 and 2 from the route in the left. Then, sequences $\langle\langle 5, 6, 8\rangle\rangle$ and $\langle\langle 10, 1, 2\rangle\rangle$ are removed from their routes. After the attempt to insert customers 5, 6 and 8 into the route on the left, and customers 10, 1 and 2 into the route on the right, the only feasible insertions were customers 5 and 10, all other customers were returned to their original routes.

Algorithm 5.3: EXCHANGE$(r_k, r_l, \mathcal{R})$

Input: Routes $r_k = \langle v_1, \ldots, v_{N_k} \rangle, r_l = \langle u_1, \ldots, u_{N_l} \rangle \in \mathcal{R}$ which are going to exchange customer

for $m = k, l$ **do**
  $v_i \leftarrow$ SELECTCUSTOMER(r_m)
  $v_j \leftarrow$ SELECTCUSTOMER(r_m)
  Remove $q_m = \langle\langle v_i, v_{i+1}, \ldots, v_{j-1}, v_j \rangle\rangle$ from r_m
end for
for $m = k, l$ **and** $n = l, k$ **do**
  for all $v_i \in q_m$ **do**
    INSERTCUSTOMER(v_i, r_n)
  end for
end for
for $m = k, l$ **do**
  for all $v_i \in q_m$ **do**
    inserted $\leftarrow$ **false**
    for all $r_j \in \mathcal{R} \setminus \{r_m\}$ **and while** *inserted* $=$ **false do**
      inserted $\leftarrow$ INSERTCUSTOMER(v_i, r_j)
    end for
    if *inserted* $=$ **false then**
      INSERTCUSTOMER(v_i, r_m)
    end if
  end for
end for

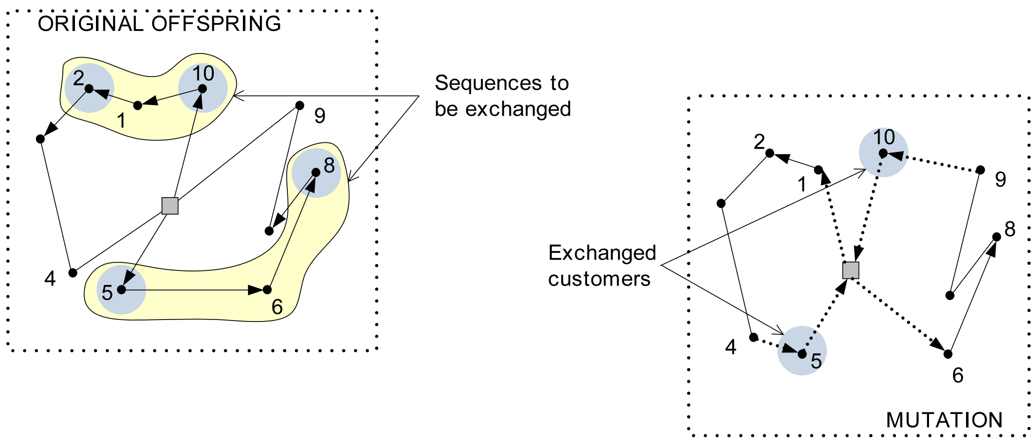


Figure 5.2: Example of the exchange mutation operator: It selects customers 5 and 8 from the route on the right and customers 10 and 2 from the route on the left. Then, it removes sequences $\langle\langle 5, 6, 8 \rangle\rangle$ and $\langle\langle 10, 1, 2 \rangle\rangle$ from their routes. After the attempt to insert customers 5, 6 and 8 into the route on the left, and customers 10, 1 and 2 into the route on the right, the only feasible insertions are customers 5 and 10, all other customers are returned to their original routes.

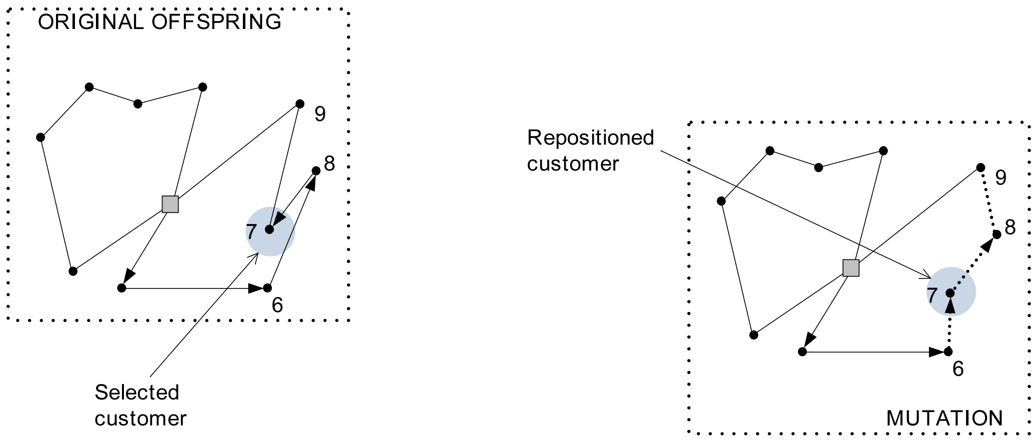


Figure 5.3: Example of the reposition mutation operator: It selects customer 7 to be repositioned. Then, customer 7 is removed from the route and reinserted between customers 6 and 8.

REPOSITION(r_k) This operation makes use of function SELECTCUSTOMER(r_k) to select one customer v_i from route r_k, and calls INSERTCUSTOMER(v_i, r_k) to reinsert v_i into r_k in the position where the shortest distance is achieved. Note that customer v_i can always be reinserted into route r_k, since route r_k was feasible before v_i was removed. Figure 5.3 presents an example of this operator. Here, customer 7 was selected to be repositioned. Then, it was removed from the route and reinserted between customers 6 and 8.

The overall mutation stage is described in Algorithm 5.4 and proceeds as follows. First, two routes r_k and r_l are chosen using function SELECTROUTE($\mathcal{R}$). If the selected routes are the same, i.e. $r_k = r_l$, REALLOCATE($r_k, \mathcal{R}$) is performed, otherwise function EXCHANGE($r_k, r_l, \mathcal{R}$) is called. Finally, r_k = SELECTROUTE($\mathcal{R}$) and then REPOSITION(r_k) are executed.

These were the modifications made to the mutation stage of the redesigned MOEA, with the aim of enhancing this process and, consequently, the algorithm's performance. The new mutation, which operates according to the inherent objective functions values of the solutions, represents one of the main contributions of this thesis.

Algorithm 5.4: MUTATE($\mathcal{R}$)

Input: Solution $\mathcal{R}$ to be mutated
Output: Mutated solution $\mathcal{R}'$

$\mathcal{R}' \leftarrow \mathcal{R}$
$r_k \leftarrow$ SELECTROUTE($\mathcal{R}'$)
$r_l \leftarrow$ SELECTROUTE($\mathcal{R}'$)
if $r_k = r_l$ **then**
  REALLOCATE($r_k, \mathcal{R}'$)
else
  EXCHANGE($r_k, r_l, \mathcal{R}'$)
end if
$r_k \leftarrow$ SELECTROUTE($\mathcal{R}'$)
REPOSITION(r_k)
return $\mathcal{R}'$

5.2 Bi-objective optimisation of VRPs with Time Windows

As in the drGA and BiEA cases, MOEA was tested using the Solomon's instances size $N = 100$. First, the two standard objectives, namely the number of routes (R), $f_1(\mathcal{R})$ in (3.4), and travel distance (D), $f_2(\mathcal{R})$ in (3.5), were considered, and then, the additional objective delivery time (T), $f_3(\mathcal{R})$ in (3.14). This section will focus on the experimental analysis of the bi-objective optimisation case, for which MOEA will be labelled as MOEA-RD.

To provide reliable statistics, MOEA-RD and all other algorithms with which it is going to be compared were run 30 times, using different random number seeds, for each benchmark instance. The population diversity and the solutions in the fittest front were recorded at the end of every evolutionary generation for later analysis. Since MOEA has a new mutation process, preliminary experiments for parameter tuning were required. The evolutionary parameters were set to the following suitable values determined by the tuning procedure:

$$popSize = 100 \qquad Tsize = 2 \qquad \gamma = 1.0$$
$$numGen = 500 \qquad \mu = 0.1$$

It is important to mention that the parameters set for MOEA-RD were fixed throughout the experiments described in this chapter and they were used for all settings and algorithms with which it was compared.

Results from MOEA-RD were analysed from three different perspectives: (i) to provide an indication of the number of instances for which MOEA is able to find proper multi-solution Pareto approximations, (ii) to know if the changes made to the mutation stage lead to performance improvement, (iii) to compare the results obtained by MOEA-RD with those from previous studies, and (iv) to compare the MOEA-RD solutions with those from NSGA-II, the latter involving features that provide a useful contrast to the new MOEA, by means of using multi-objective performance metrics.

5.2.1 Analysis of the Pareto approximations

The first issue to be studied is whether MOEA-RD does find appropriate Pareto approximations to the Solomon's test instances. That is, are the outcome from MOEA-RD such that there are trade-offs between the objectives that result in more that one single solution in the Pareto approximation sets?

Let us recall that, according to the best-known results from previous studies shown in Table 3.5, many of the instances have conflicting objectives, as indicated by the extreme solutions. However, these solutions were found by independent studies which prioritised the optimisation of either the number of routes or the travel distance, consequently the true Pareto front is unknown.

Figure 5.4 shows the number of instances for which MOEA-RD found overall approximation sets with multiple solutions, i.e. instances with conflicting objectives, out of the total number of instances in each category (wider clear bars). For comparison purposes, the corresponding numbers from the previous multi-objective ap-

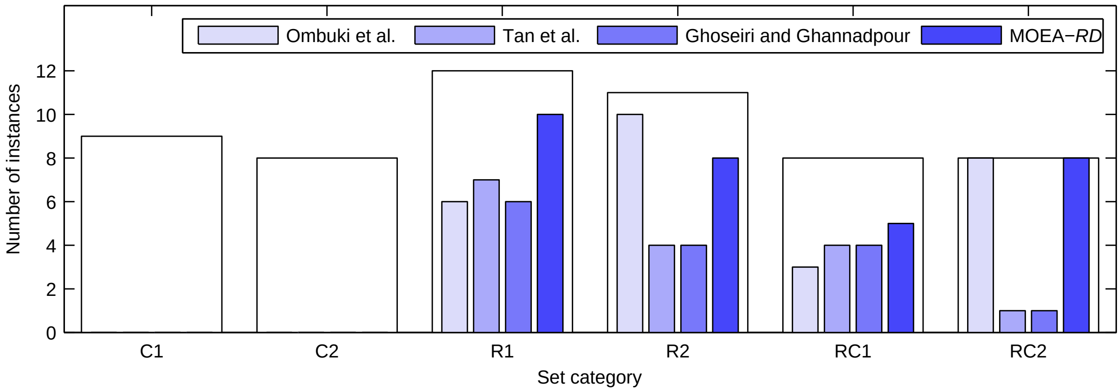


Figure 5.4: Number of instances with conflicting objectives found by previous multi-objective studies and by MOEA-RD, out of the total number of instances in each category.

proaches of Ombuki et al. [182], Tan et al. [228], and Ghoseiri and Ghannadpour [111] are also shown. MOEA-RD managed to find more approximation sets with multiple solutions than the previous methods for instances in categories R1, RC1 and RC2, while that of Ombuki et al. [182] for category R2. For categories C1 and C2, as was stated earlier, all four approaches agreed did not include any instances with conflicting objectives. In total, MOEA-RD found approximation sets with incompatible objectives for 31 instances out of 56, the GA of Ombuki et al. [182] for 27, the EA of Tan et al. [228] for 16, and the approach of Ghoseiri and Ghannadpour [111] for 15.

Additionally, for each instance, the overall Pareto approximation from the previous multi-objective studies was taken, and this was compared with that from MOEA-RD. Table 5.1 shows the number of instances for which the overall Pareto approximations from previous studies dominate ($\prec$) and are dominated ($\succ$) by those from MOEA-RD. We see that the overall Pareto approximation from MOEA-RD dominate those from the previous studies for many of the instances, while they are dominated for only few of them.

Figure 5.5 presents bar plots showing the number of instances for different sizes of the Pareto approximation found by MOEA-RD, revealing that the 31 instances

Author	C1		C2		R1		R2		RC1		RC2		Total	
	$\prec$	$\succ$	$\prec$	$\succ$	$\prec$	$\succ$	$\prec$	$\succ$	$\prec$	$\succ$	$\prec$	$\succ$	$\prec$	$\succ$
Ombuki et al. [182]	0	1	0	1	1	11	1	3	1	6	0	3	3	25
Tan et al. [228]	0	4	0	4	3	5	0	10	2	4	0	4	5	31
Ghoseiri and Ghannadpour [111]	0	0	0	3	1	11	0	10	0	7	0	6	1	37

Table 5.1: Number of instances for which the overall Pareto approximations from previous studies dominate ($\prec$) and are dominated ($\succ$) by those from MOEA-*RD*

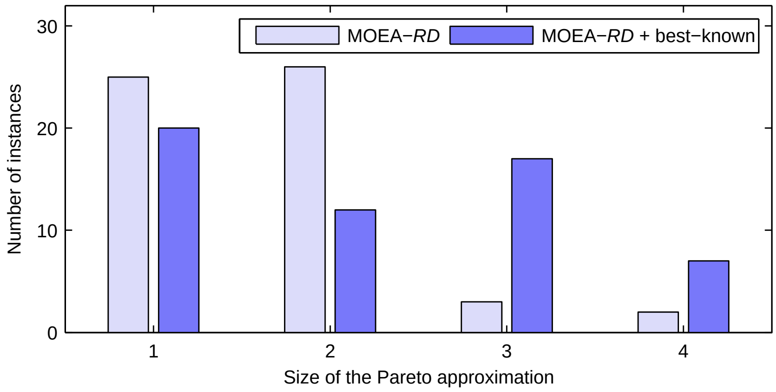


Figure 5.5: Number of instances for each number of solutions in the Pareto approximations found by MOEA-*RD*, and considering the best-known solutions.

mentioned above have 2, 3, and 4 solutions in their non-dominated set. Furthermore, if the best-known solutions from Table 3.5 are considered in addition to those from MOEA-*RD*, the number of instances with three and four solutions in its best-known Pareto approximation increases. This confirms the multi-objective nature of the VRPTW, and indicates the extent to which MOEA-*RD* is finding the best-known Pareto approximation.

In the past, fully multi-objective comparisons between algorithms have been hampered by a lack of published solution details. Until now, only the study of Ghoseiri and Ghannadpour [111] has appropriately presented the solutions in the Pareto approximations for each instance. To remedy this for future studies and to promote proper multi-objective comparisons, Table 5.2 presents the full details of the number of routes (R) and travel distance (D), associated with the solutions in the overall

Instance	R	D	R	D	Instance	R	D	R	D
R101	20	1642.88	19	1650.80	RC102	14	1480.26	13	1501.11
R102	18	1474.19	17	1486.12	RC105	15	1519.44	14	1540.18
R103	14	1219.37	13	1308.28	RC106	13	1379.68	12	1395.70
R104	11	984.56	10	990.79	RC107	12	1215.06	11	1234.49
R105	15	1364.91	14	1377.11	RC108	11	1122.98	10	1158.22
R106	13	1241.65	12	1261.52	RC201	7	1299.58	6	1316.25
R107	11	1083.30	10	1154.38		5	1329.26	4	1438.43
R108	10	960.03	9	984.75	RC202	5	1120.15	4	1165.57
R109	13	1154.61	12	1157.76	RC203	4	954.51	3	1061.47
R110	12	1088.61	11	1094.75	RC204	4	792.84	3	802.71
R201	5	1194.07	4	1254.77	RC205	7	1205.06	6	1214.49
R202	5	1050.41	4	1087.29		5	1259.00	4	1318.71
R203	5	905.34	4	912.24	RC206	5	1077.48	4	1085.82
	3	950.90				3	1191.62		
R205	4	968.09	3	1040.29	RC207	5	1001.51	4	1001.73
R206	4	899.83	3	930.58		3	1133.27		
R208	3	712.98	2	736.90	RC208	4	780.07	3	844.96
R209	4	878.05	3	921.97					
R210	4	936.68	3	961.36					

Table 5.2: Number of routes and travel distance for the instances where both objectives are in conflict, corresponding to the solutions in the overall Pareto approximations obtained by MOEA-RD.

Pareto approximation obtained by MOEA-RD, for the 31 instances in which the two objectives are in conflict.

5.2.2 Comparison with BiEA

Since the full multi-objective results are available, this comparison will be done according to the coverage, convergence and hypervolume performance metrics, which were applied over the non-dominated sets obtained by BiEA and MOEA-RD. To apply the coverage metric, $\mathsf{M_C}(\text{BiEA}_i, \text{MOEA-}RD_j)$ and $\mathsf{M_C}(\text{MOEA-}RD_j, \text{BiEA}_i)$ were computed for all repetitions $i, j = 1, \ldots, 30$, giving 900 $\mathsf{M_C}$ values for each instance. Figure 5.6 presents six series of box-and-whisker plots, one for each set category, to display the resultant distributions of the $\mathsf{M_C}$ values. For each instance there are two boxes, the one on the left showing the distribution of the $\mathsf{M_C}(\text{BiEA}, \text{MOEA-}RD)$

values, and the one on the right $\mathsf{M_C}(\text{MOEA-}RD, \text{BiEA})$. On each box, the central mark is the median $\widetilde{\mathsf{M}}_\mathsf{C}$, while the lower and upper edges correspond to the first and third quartiles. This time, boxes have an additional notch, which display the variability of the median. The width of the notches is computed so that boxes which notches do not overlap have different medians at the 95% significance level [167].

We observe that for all instances in categories C1 and C2, the boxes corresponding to $\mathsf{M_C}(\text{MOEA-}RD, \text{BiEA})$ are agglomerated at the top, except for instance C204 that extends from 0 to 1, which means that solutions from MOEA-RD always cover those from BiEA, while those regarding $\mathsf{M_C}(\text{BiEA}, \text{MOEA-}RD)$ are agglomerated at the top, at the bottom, and some extend over the whole range, which means that solutions from BiEA not always cover those from MOEA-RD. For instances in categories R2 and RC2, $\widetilde{\mathsf{M}}_\mathsf{C}(\text{MOEA-}RD, \text{ BiEA}) = 1$, with the exception of two instances in RC2 where it is located at 0.5, while $\widetilde{\mathsf{M}}_\mathsf{C}(\text{BiEA, MOEA-}RD)$ is always zero, which means that the medians significantly differ. In the case of the sets R1 and RC1, we observe mixed results, thus we will analyse them in more detail later.

To compare the algorithms using the convergence metric, for each algorithm and instance, the overall non-dominated solutions were extracted from the 30 Pareto approximations. Then, from the overall non-dominated sets, a composite reference set $\mathcal{R}$ was found. Afterwards, $\mathsf{M_D}(\text{BiEA}_i, \mathcal{R})$ and $\mathsf{M_D}(\text{MOEA-}RD_i, \mathcal{R})$, $\forall\, i = 1, \ldots, 30$, were computed and normalised regarding the maximal solution as in Section 4.3.2.2.

The distribution of these $\mathsf{M_D}$ values are shown in Figure 5.7 as box-and-whisker plots. For each instance there are two boxes, the one on the left showing the distribution of the $\mathsf{M_D}(\text{BiEA}, \mathcal{R})$ values, and the one on the right $\mathsf{M_D}(\text{MOEA-}RD, \mathcal{R})$. We observe here that both algorithms present similar performance for instances in categories C1 and C2, with some exceptions where solutions from MOEA-RD are closer to $\mathcal{R}$ than those from BiEA. For categories R2 and RC2, MOEA-RD sur-

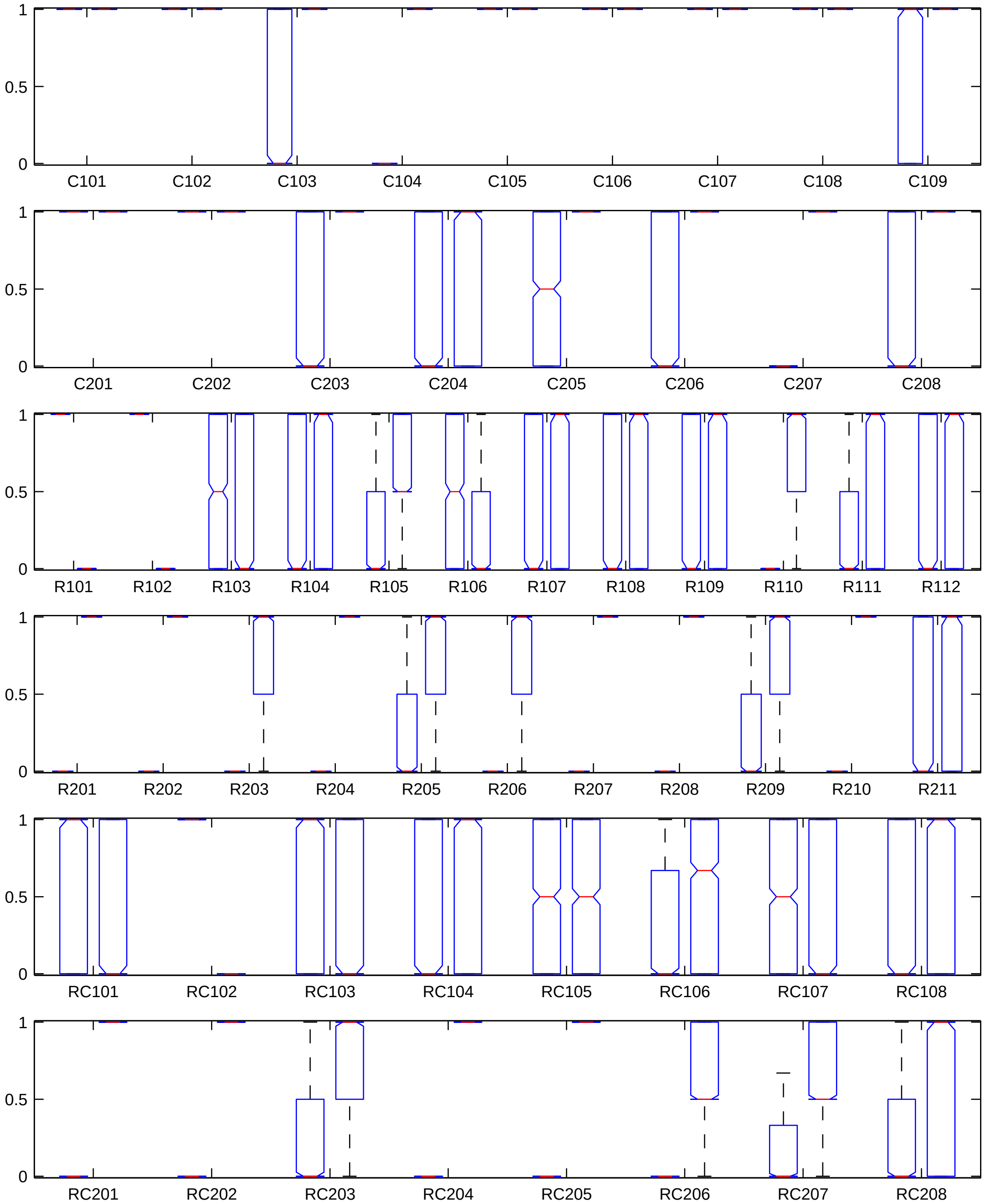


Figure 5.6: Box-and-whisker plots representing the distribution of the $\mathsf{M_C}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_C}(\text{BiEA}, \text{MOEA-}RD)$, and the one on the right $\mathsf{M_C}(\text{MOEA-}RD, \text{BiEA})$.

passed BiEA in the majority of the instances, in the sense that $\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{MOEA-}RD, \mathcal{R})$ is lower than $\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{BiEA}, \mathcal{R})$ and the notches in the boxes, and in some cases the

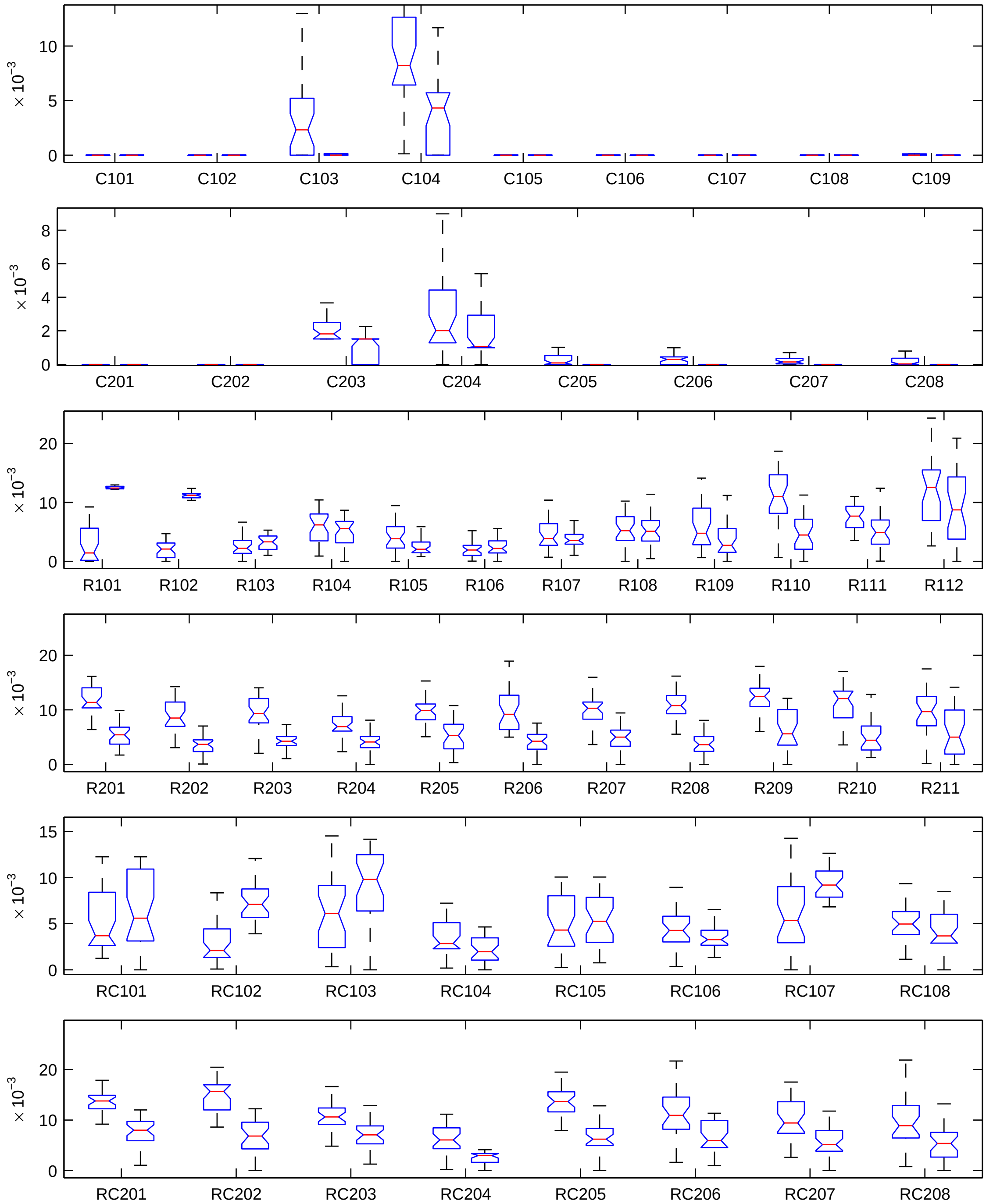


Figure 5.7: Box-and-whisker plots representing the distribution of the $\mathsf{M_D}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_D}(\text{BiEA}, \mathcal{R})$, and the one on the right $\mathsf{M_D}(\text{MOEA-}RD, \mathcal{R})$.

complete boxes, do not overlap. Results for the instances in categories R1 and RC1 do not show a clear visual difference.

Algorithm	Metric	C1	C2	R1	R2	RC1	RC2
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.77 (0)	0.50 (0)	0.47 (2)	0.12 (0)	0.51 (4)	0.14 (0)
BiEA	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-2})$	0.18 (0)	0.08 (0)	0.55 (3)	1.00 (0)	0.51 (3)	1.14 (0)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	76.92 (0)	87.29 (0)	66.43 (3)	78.39 (0)	69.64 (1)	80.21 (0)
	$\overline{\mathsf{M}}_{\mathsf{C}}$	0.94 (8)	0.94 (6)	0.48 (8)	0.85 (11)	0.44 (3)	0.79 (8)
MOEA-*RD*	$\overline{\mathsf{M}}_{\mathsf{D}}(\times 10^{-2})$	0.07 (2)	0.04 (3)	0.58 (4)	0.48 (11)	0.60 (0)	0.62 (8)
	$\overline{\mathsf{M}}_{\mathsf{H}}(\times 10^{-2})$	77.01 (2)	87.34 (6)	66.37 (4)	78.90 (11)	69.60 (0)	80.73 (7)

Table 5.3: Averages $\overline{\mathsf{M}}_{\mathsf{C}}$, $\overline{\mathsf{M}}_{\mathsf{D}}$ and $\overline{\mathsf{M}}_{\mathsf{H}}$ over instance categories for the solutions obtained with BiEA and MOEA-*RD*. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

On the other hand, in order to compute the hypervolume metric M_{H}, the vector $\mathbf{z} = (N, D^{\max})$, corresponding to the maximal solution with largest number of routes N and longest travel distance $D^{\max}$, was considered the reference point for each instance. Then, $\mathsf{M}_{\mathsf{H}}(\text{BiEA}_i, \boldsymbol{z})$ and $\mathsf{M}_{\mathsf{H}}(\text{MOEA-}RD_i, \boldsymbol{z}), \forall\, i = 1, \ldots, 30$, were computed and normalised according to the space defined between point $\boldsymbol{z}$ and the origin. These M_{H} values were grouped by the instances in each set category and the average $\overline{\mathsf{M}}_{\mathsf{H}}$ was calculated. These M_{H} results, along with M_{C} and M_{D}, were analysed more rigorously by counting the significant performance difference for individual problem instances.

Table 5.3 presents the averages $\overline{\mathsf{M}}_{\mathsf{C}}$, $\overline{\mathsf{M}}_{\mathsf{D}}$ and $\overline{\mathsf{M}}_{\mathsf{H}}$ over all the instances within each problem category, and the numbers of instances, in brackets, for which there was significant improvement over the other approach. We can observe here more clearly that MOEA-*RD* has a better performance than BiEA regarding the coverage metric, since for all categories, with the exception of RC1, $\overline{\mathsf{M}}_{\mathsf{C}}(\text{MOEA-}RD, \text{BiEA}) > \overline{\mathsf{M}}_{\mathsf{C}}(\text{BiEA}, \text{MOEA-}RD)$, and the difference is significant for the majority of the instances. We see a similar pattern with respect to the convergence metric, since solutions from MOEA-*RD* are closer to $\mathcal{R}$ than those from BiEA, and the difference is significant for many of the instances. An analogous situation occurs for the hypervolume metric, which is larger for MOEA-*RD* and the difference is also significant for many of the instances.

Based on these results, we conclude that MOEA-RD is a significant improvement of BiEA, since the solutions found by the former cover more widely those from the latter, are closer to the composite reference sets, and delimit a wider objective space for the majority of the benchmark instances. We argue that this improvement is due to the modified mutation process and operators, since this is the difference between both algorithms. Hence, the new mutation procedure is accomplishing the purpose of finding better solutions, consequently enhancing the algorithm's performance.

5.2.3 Comparison with previous studies

Here, the results found by MOEA-RD are compared with the best published results known, previously presented in Section 3.2.1 (Table 3.5 in page 76). Tables 5.4 and 5.5 show, for each instance and both the solution with the smallest number of routes ($\min R$) and the solution with the shortest travel distance ($\min D$), the best-known results for each objective, along with the best result found by MOEA-RD after 30 runs. For the results from MOEA-RD, the percentage difference with respect to the best-known result is displayed ($\%R$ and $\%D$), which is averaged over instances in each set category. A negative percentage indicates improvement over the best-known result.

Let us analyse first the results for the solutions with lowest number of routes ($\min R$). We can see that MOEA-RD found the best-known solutions, shown in bold, for all instances in categories C1 and C2, and two more in R1. Additionally, for 13 instances, MOEA-RD achieved solutions with the best-known number of routes and no more than 2% above the corresponding travel distance, shown in italic. Overall, solutions from MOEA-RD have, on average, 5.46% more routes than the best-known, which correspond to a saving of 0.42% regarding travel distance. In respect to the solutions with the shortest travel distance ($\min D$), MOEA-RD found the best-known solutions, shown in bold, for all instances in categories C1 and C2.

	min R						min D					
Inst.	Best-known		MOEA-RD				Best-known		MOEA-RD			
	R	D	R	D	$\%R$	$\%D$	R	D	R	D	$\%R$	$\%D$
C101	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C102	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C103	10	828.06	**10**	**828.06**	0.00	0.00	10	828.06	**10**	**828.06**	0.00	0.00
C104	10	824.78	**10**	**824.78**	0.00	0.00	10	824.78	**10**	**824.78**	0.00	0.00
C105	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C106	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C107	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C108	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
C109	10	828.94	**10**	**828.94**	0.00	0.00	10	828.94	**10**	**828.94**	0.00	0.00
Category average					0.00	0.00					0.00	0.00
R101	18	1613.59	19	1650.80	5.56	2.31	18	1613.59	20	1642.88	11.11	1.82
R102	17	1486.12	**17**	**1486.12**	0.00	0.00	18	1454.68	*18*	*1474.19*	0.00	1.34
R103	13	1292.68	*13*	*1308.28*	0.00	1.21	14	1213.62	*14*	*1219.37*	0.00	0.47
R104	9	1007.24	10	990.79	11.11	-1.63	10	974.24	11	984.56	10.00	1.06
R105	14	1377.11	**14**	**1377.11**	0.00	0.00	15	1360.78	*15*	*1364.91*	0.00	0.30
R106	12	1251.98	*12*	*1261.52*	0.00	0.76	13	1240.47	*13*	*1241.65*	0.00	0.10
R107	10	1104.66	10	1154.38	0.00	4.50	11	1073.34	*11*	*1083.30*	0.00	0.93
R108	9	960.88	9	984.75	0.00	2.48	10	947.55	*10*	*960.03*	0.00	1.32
R109	11	1194.73	12	1157.76	9.09	-3.09	13	1151.84	*13*	*1154.61*	0.00	0.24
R110	10	1118.59	11	1094.75	10.00	-2.13	12	1072.41	*12*	*1088.61*	0.00	1.51
R111	10	1096.72	11	1061.37	10.00	-3.22	12	1053.50	*11*	*1061.37*	-8.33	0.75
R112	9	982.14	10	980.83	11.11	-0.13	10	953.63	10	980.83	0.00	2.85
Category average					4.74	0.09					1.06	1.06
RC101	14	1696.94	15	1625.26	7.14	-4.22	15	1623.58	*15*	*1625.26*	0.00	0.10
RC102	12	1554.75	13	1501.11	8.33	-3.45	14	1461.23	*14*	*1480.26*	0.00	1.30
RC103	11	1261.67	*11*	*1278.19*	0.00	1.31	11	1261.67	*11*	*1278.19*	0.00	1.31
RC104	10	1135.48	*10*	*1144.39*	0.00	0.78	10	1135.48	*10*	*1144.39*	0.00	0.78
RC105	13	1629.44	14	1540.18	7.69	-5.48	16	1518.58	*15*	*1519.44*	-6.25	0.06
RC106	11	1424.73	12	1395.70	9.09	-2.04	13	1371.69	*13*	*1379.68*	0.00	0.58
RC107	11	1222.16	*11*	*1234.49*	0.00	1.01	12	1212.83	*12*	*1215.06*	0.00	0.18
RC108	10	1139.82	*10*	*1158.22*	0.00	1.61	11	1117.53	*11*	*1122.98*	0.00	0.49
Category average					4.03	-1.31					-0.78	0.60

Table 5.4: Best-known results from literature and the best results obtained by MOEA-RD, considering the lowest number of routes and the shortest travel distance, for Solomon's instance categories C1, R1 and RC1.

Furthermore, for 22 instances, MOEA-RD achieved solutions with an increase of no more than 2% in the travel distance and with equal or less number of routes compared with the best-known, shown in talic. Overall, MOEA-RD solutions are,

	min R						min D					
Inst.	Best-known		MOEA-RD				Best-known		MOEA-RD			
	R	D	R	D	$\%R$	$\%D$	R	D	R	D	$\%R$	$\%D$
C201	3	591.56	**3**	**591.56**	0.00	0.00	3	591.56	**3**	**591.56**	0.00	0.00
C202	3	591.56	**3**	**591.56**	0.00	0.00	3	591.56	**3**	**591.56**	0.00	0.00
C203	3	591.17	**3**	**591.17**	0.00	0.00	3	591.17	**3**	**591.17**	0.00	0.00
C204	3	590.60	**3**	**590.60**	0.00	0.00	3	590.60	**3**	**590.60**	0.00	0.00
C205	3	588.88	**3**	**588.88**	0.00	0.00	3	588.88	**3**	**588.88**	0.00	0.00
C206	3	588.49	**3**	**588.49**	0.00	0.00	3	588.49	**3**	**588.49**	0.00	0.00
C207	3	588.29	**3**	**588.29**	0.00	0.00	3	588.29	**3**	**588.29**	0.00	0.00
C208	3	588.32	**3**	**588.32**	0.00	0.00	3	588.32	**3**	**588.32**	0.00	0.00
Category average					0.00	0.00					0.00	0.00
R201	4	1252.37	*4*	*1254.77*	0.00	0.19	9	1149.68	5	1194.07	-44.44	3.86
R202	3	1191.70	4	1087.29	33.33	-8.76	8	1034.35	*5*	*1050.41*	-37.50	1.55
R203	3	939.54	*3*	*950.90*	0.00	1.21	6	874.87	5	905.34	-16.67	3.48
R204	2	825.52	3	752.83	50.00	-8.81	4	736.52	3	752.83	-25.00	2.21
R205	3	994.42	3	1040.29	0.00	4.61	5	954.16	*4*	*968.09*	-20.00	1.46
R206	3	906.14	3	930.58	0.00	2.70	5	879.89	4	899.83	-20.00	2.27
R207	2	837.20	3	818.97	50.00	-2.18	4	799.86	3	818.97	-25.00	2.39
R208	2	726.75	*2*	*736.90*	0.00	1.40	4	705.45	*3*	*712.98*	-25.00	1.07
R209	3	909.16	*3*	*921.97*	0.00	1.41	5	859.39	4	878.05	-20.00	2.17
R210	3	939.34	3	961.36	0.00	2.34	5	910.70	4	936.68	-20.00	2.85
R211	2	892.71	3	785.97	50.00	-11.96	4	755.96	3	785.97	-25.00	3.97
Category average					16.67	-1.62					-25.33	2.48
RC201	4	1406.91	4	1438.43	0.00	2.24	6	1134.91	7	1299.58	16.67	14.51
RC202	3	1367.09	4	1165.57	33.33	-14.74	8	1095.64	5	1120.15	-37.50	2.24
RC203	3	1049.62	*3*	*1061.47*	0.00	1.13	5	928.51	4	954.51	-20.00	2.80
RC204	3	798.41	*3*	*802.71*	0.00	0.54	4	786.38	*4*	*792.84*	0.00	0.82
RC205	4	1297.19	*4*	*1318.71*	0.00	1.66	7	1157.55	7	1205.06	0.00	4.10
RC206	3	1146.32	3	1191.62	0.00	3.95	7	1054.61	5	1077.48	-28.57	2.17
RC207	3	1061.14	3	1133.27	0.00	6.80	6	966.08	5	1001.51	-16.67	3.67
RC208	3	828.14	3	844.96	0.00	2.03	4	779.31	*4*	*780.07*	0.00	0.10
Category average					4.17	0.45					-10.76	3.80
Overall average					5.46	-0.42					-6.40	1.34
St. Dev.					12.7	3.70					12.78	2.17

Table 5.5: Best-known results from literature and the best results obtained by MOEA-RD, considering the lowest number of routes and the shortest travel distance, for Solomon's instance categories C2, R2 and RC2.

on average, 1.34% above the best-known travel distance, however they reduce the number of routes in 6.40%.

If we compare the average difference between results from MOEA-RD and the best-known with that from previous studies shown in Tables 3.6 and 3.7 (pages 84 and

85), we can see that MOEA-RD surpasses the approaches of Cordone and Wolfler Calvo [50] and of Ghoseiri and Ghannadpour [111], in the sense that the differences from MOEA-RD are smaller in both objectives. However, if only the number of routes is taken into account, MOEA-RD additionally surpasses the approaches of Jung and Moon [137] and of Alvarenga et al. [7], while if the travel distance is exclusively considered, MOEA-RD surpasses all approaches, except those of Jung and Moon [137] and of Alvarenga et al. [7].

It is important to observe that, with respect to solutions with the shortest travel distance ($\min D$), there is a considerable saving in the number of routes for categories R2 and RC2, 25.33% and 10.76%, respectively, though the travel distance is, correspondingly, 2.48% and 3.80% larger. This might suggest that MOEA-RD is having difficulties for exploring the search space where solutions with a larger number of routes are located, which could potentially reduce the travel distance. As was described in Section 3.2.1, instances in these categories have a wider search space than that for instances in the rest of the categories, feature which makes them even harder to solve. Consequently, in order to know exaclty what prevents the algorithm achieving solutions with larger numbers of routes, a deeper analysis is going to be proposed as a future research activity.

Results from MOEA-RD constitute one of the contributions of this thesis, since, although they are not the overall best, they are comparable to many, in the sense that they have one of the objectives within 2% difference with respect to the best-known, while the other objective is equal or better.

On the other hand, Table 5.6 presents the average results, over the instances in each set category, from MOEA-RD, as well as the best average results from past studies and those from previous multi-objective approaches. This table presents the same format as that of Table 4.7, used in the previous chapter for comparing results from

Author	C1	C2	R1	R2	RC1	RC2	Total
min R	10.00	3.00	11.91	2.73	11.50	3.25	405.00
	828.38	589.86	1212.73	952.67	1384.30	1108.52	57192.00
min D	10.00	3.00	11.92	5.36	12.88	6.25	486.00
	828.38	589.86	1121.10	878.41	1341.67	1004.20	54779.02
Ombuki et al. [182]	10.00	3.00	12.67	3.09	12.38	3.50	427.00
(min R)	828.48	590.60	1212.58	956.73	1379.87	1148.66	57484.35
Ombuki et al. [182]	10.00	3.00	13.17	4.55	13.00	5.63	471.00
(min D)	828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
Tan et al. [228]	10.00	3.00	12.92	3.55	12.38	4.25	441.00
	828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ghoseiri and Ghannadpour	10.00	3.00	12.92	3.45	12.75	3.75	439.00
[111] (min R)	828.38	591.49	1228.60	1033.53	1392.09	1162.40	58735.22
Ghoseiri and Ghannadpour	10.00	3.00	13.50	3.82	13.25	4.00	456.00
[111] (min D)	828.38	591.49	1217.03	1049.62	1384.3	1157.41	58671.12
MOEA-RD (min R)	10.00	3.00	12.33	3.09	12.00	3.38	419.00
	828.38	589.86	1209.04	931.08	1359.69	1119.59	56758.86
MOEA-RD (min D)	10.00	3.00	13.17	3.91	12.63	5.13	457.00
	828.38	589.86	1188.03	900.29	1345.66	1028.90	55330.28
% diff. R	0.00	0.00	3.55	13.22	4.35	3.85	3.46
% diff. D	0.00	0.00	5.97	2.49	0.30	2.46	1.01

Table 5.6: Number of routes and travel distance, averaged over categories, for the best solutions found by previous studies and MOEA-RD.

BiEA. Let us remind that for each instance, the solutions with the smallest number of routes (min R) and with the shortest travel distance (min D) were taken and the average over the instances in each set category was computed for each objective.

With respect to the number of routes (min R), we see that, for instance categories C1 and C2, MOEA-RD obtained the best-known results. For categories R1, RC1 and RC2, and in total, results are between 3.5% and 4.5% higher, while for R2 they are nearly 13% above. Regarding the travel distance, MOEA-RD (min D) obtained the best-known results for categories C1 and C2. In respect to categories R2, RC1 and RC2, and in total, the difference is of no more than 2.5%, and below 6% for category R1.

Considering exclusively the multi-objective studies, MOEA-RD found similar or better solutions than those obtained by Ombuki et al. [182] and Ghoseiri and Ghannadpour [111] in all categories and in total regarding solutions with the smallest number of routes ($\min R$), and in categories C1, C2, R1, and RC1, and in total considering the solutions with shortest travel distance ($\min D$). In respect to the the study of Tan et al. [228], solutions from MOEA-RD are better in categories C1 and C2. In general, we conclude that MOEA-RD maintains the good performance of BiEA, in the sense that it is comparable or better than that of previous multi-objective approaches.

5.2.4 Comparison with NSGA-II

As was stated in Section 2.4.6, NSGA-II [67] has established itself as a benchmark algorithm for multi-objective optimisation and is currently renowned as one of the leading general purpose multi-objective optimisers. Hence, given the operational similitude, NSGA-II is the best algorithm to compare MOEA with. In this case, an additional series of experiments was carried out in order to compare the performance of MOEA-RD set to use edit distance instead of Jaccard similarity. This algorithm will be labelled as MOEA-ERD.

Results from NSGA-II, MOEA-ERD and MOEA-RD are now analysed by means of the coverage, convergence, and hypervolume performance metrics. In order to make a fair comparison, NSGA-II was coded with the same recombination and mutation operators, and parameter values as MOEA. The difference lies in how parent and survival selections are performed: while MOEA considers solution similarity in both cases, NSGA-II uses exclusively fitness in the former and crowding distance in the latter.

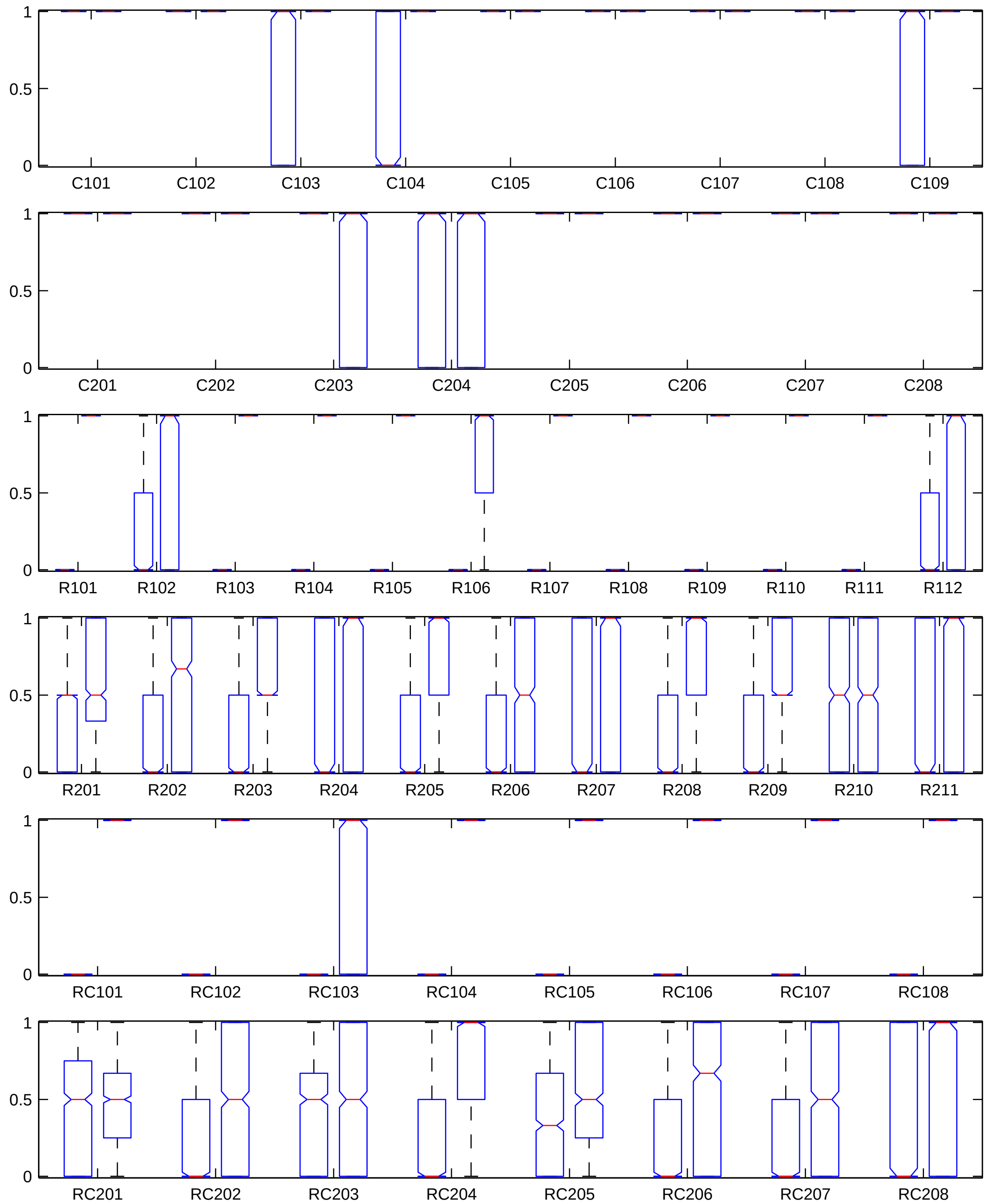


Figure 5.8: Box-and-whisker plots representing the distribution of the $\mathsf{M_C}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_C}$(NSGA-II, MOEA-*RD*), and the one on the right $\mathsf{M_C}$(MOEA-*RD*, NSGA-II).

The procedure to calculate the $\mathsf{M_C}$ values was the same as previously applied to compare BiEA and MOEA-*RD*. Figure 5.8 presents six series of box-and-whisker plots to display the resultant distributions of the $\mathsf{M_C}$ values. For each instance there

Algorithm	Covers	C1	C2	R1	R2	RC1	RC2
NSGA-II	MOEA-ERD	0.80 (0)	0.91 (1)	0.14 (0)	0.41 (4)	0.08 (0)	0.37 (1)
	MOEA-RD	0.81 (0)	0.91 (2)	0.14 (0)	0.35 (0)	0.11 (0)	0.35 (0)
MOEA-ERD	NSGA-II	0.95 (4)	0.88 (1)	0.82 (12)	0.54 (7)	0.89 (8)	0.53 (6)
	MOEA-RD	0.92 (2)	0.90 (2)	0.49 (7)	0.41 (1)	0.57 (6)	0.42 (1)
MOEA-RD	NSGA-II	0.93 (4)	0.88 (0)	0.81 (12)	0.60 (10)	0.85 (8)	0.57 (7)
	MOEA-ERD	0.89 (0)	0.90 (1)	0.46 (2)	0.55 (9)	0.38 (0)	0.48 (5)

Table 5.7: Averages $\overline{\mathsf{M}}_\mathsf{C}$ over instance categories for the solutions obtained with NSGA-II, MOEA-ERD, and MOEA-RD. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

are two boxes, the one on the left showing $\mathsf{M}_\mathsf{C}(\text{NSGA-II}, \text{MOEA-}RD)$, and the one on the right $\mathsf{M}_\mathsf{C}(\text{MOEA-}RD, \text{NSGA-II})$. Both algorithms show similar coverage of each other for instances in categories C1 and C2, since the median $\widetilde{\mathsf{M}}_\mathsf{C} = 1$ for all cases, except for instance C104, where $\widetilde{\mathsf{M}}_\mathsf{C}(\text{NSGA-II}, \text{MOEA-}RD) = 0$. For instances in categories R1 and RC1, solutions in the non-dominated sets found by MOEA-RD completely cover those obtained by NSGA-II, except for three instances in category R1 and one in RC1, and solutions from NSGA-II never cover those from MOEA-RD, with the exception of two instances in R1. For categories R2 and RC2, although for several instances $\widetilde{\mathsf{M}}_\mathsf{C}(\text{NSGA-II}, \text{MOEA-}RD) < \widetilde{\mathsf{M}}_\mathsf{C}(\text{MOEA-}RD, \text{NSGA-II})$, and the notches do not overlap, we need more details for a proper analysis.

Table 5.7 presents the averages $\overline{\mathsf{M}}_\mathsf{C}$, over all the instances within each problem category, of the coverage metric applied to every ordered pair of the algorithms NSGA-II, MOEA-ERD, and MOEA-RD, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. We can observe that, for categories C1, R1, R2, RC1, and RC2, $\overline{\mathsf{M}}_\mathsf{C}(\text{MOEA-E}RD, \text{NSGA-II})$ and $\overline{\mathsf{M}}_\mathsf{C}(\text{MOEA-}RD, \text{NSGA-II})$ are greater than the opposite cases, and the difference is significant for the majority of the instances in those sets. Between them, solutions from MOEA-ERD significantly cover those from MOEA-RD in more instances in categories R1 and RC1, and the latter significantly cover the former in many instances in categories R2 and RC2.

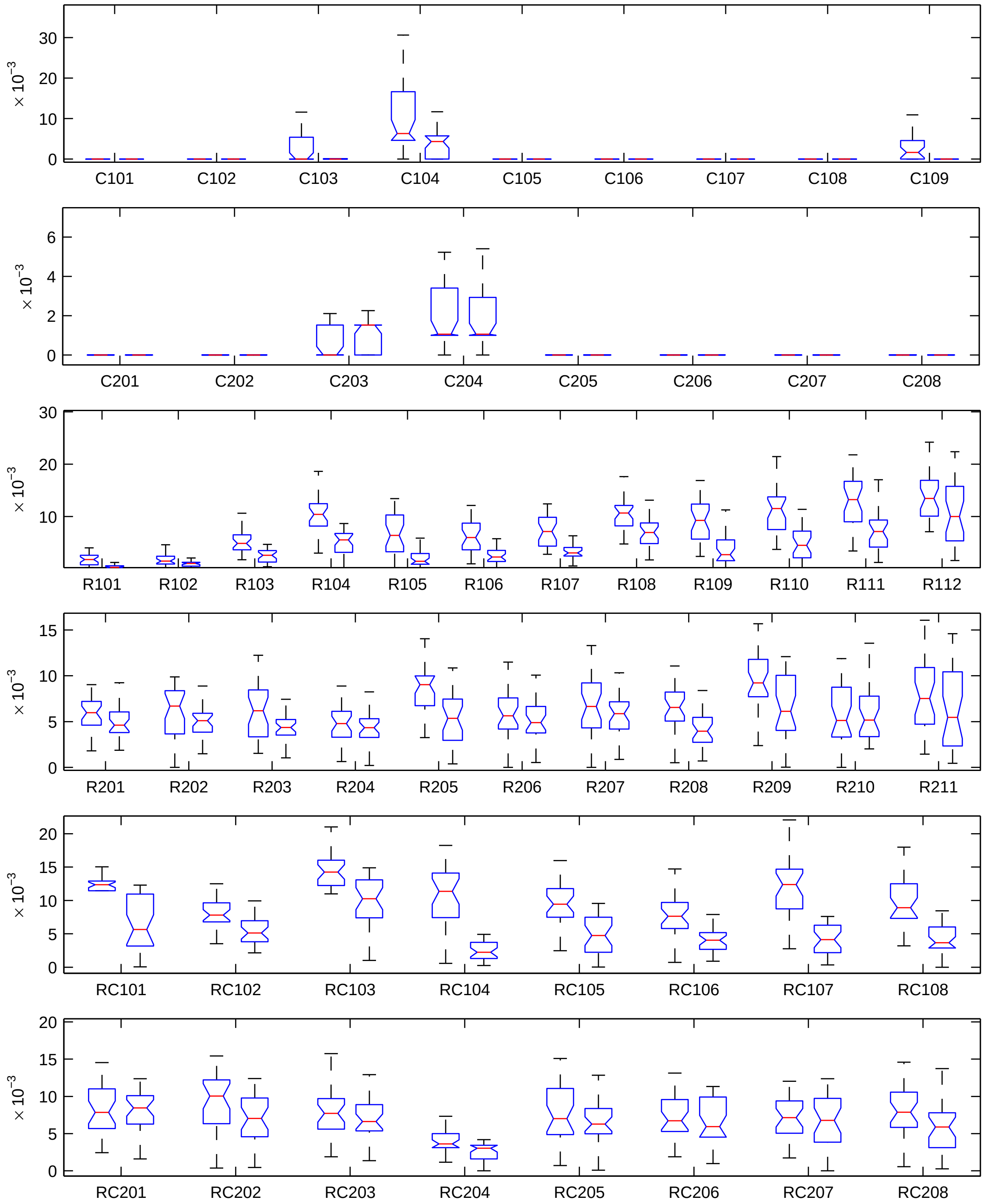


Figure 5.9: Box-and-whisker plots representing the distribution of the $\mathsf{M_D}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_D}(\text{NSGA-II}, \mathcal{R})$, and the one on the right $\mathsf{M_D}(\text{MOEA-}RD, \mathcal{R})$.

To compare the algorithms using the convergence metric, the same procedure used to compare BiEA and MOEA-RD was followed, however, the overall Pareto approximations from MOEA-ERD were also taken into account to find the composite

Algorithm	Compares	C1		C2		R1		R2		RC1		RC2	
NSGA-II	MOEA-ERD	1.87	(0)	0.38	(0)	8.05	(0)	6.74	(2)	10.24	(0)	7.45	(0)
	MOEA-RD		(0)		(0)		(0)		(0)		(0)		(0)
MOEA-ERD	NSGA-II	0.55	(3)	0.34	(0)	4.34	(12)	6.38	(3)	4.72	(8)	7.15	(1)
	MOEA-RD		(0)		(0)		(1)		(1)		(1)		(0)
MOEA-RD	NSGA-II	0.74	(3)	0.38	(0)	4.17	(12)	5.29	(4)	5.28	(8)	6.40	(1)
	MOEA-ERD		(0)		(0)		(1)		(5)		(0)		(0)

Table 5.8: Averages $\overline{\mathsf{M}}_{\mathsf{D}}$ ($\times 10^{-3}$) over instance categories for the solutions obtained with NSGA-II, MOEA-ERD, and MOEA-RD. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

reference set $\mathcal{R}$. The distribution of the M_{D} values are shown in Figure 5.9 as box-and-whisker plots. For each instance there are two boxes, the one on the left showing $\mathsf{M}_{\mathsf{D}}(\text{NSGA-II}, \mathcal{R})$, and the one on the right showing $\mathsf{M}_{\mathsf{D}}(\text{MOEA-}RD, \mathcal{R})$. In this figure we can see that, in general, both algorithms present similar performance for all instances in categories C1 and C2, since $\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{NSGA-II}, \mathcal{R}) \approx \widetilde{\mathsf{M}}_{\mathsf{D}}(\text{MOEA-}RD, \mathcal{R})$, except for instance C203 where notches do not overlap. For the majority of the instances in categories R1 and RC1, results from MOEA-RD are significantly closer to $\mathcal{R}$ than those from NSGA-II are, as indicated by the lower medians $\widetilde{\mathsf{M}}_{\mathsf{D}}$ and the non-overlapping notches. For the rest of the instances in these categories the algorithms perform similarly. Finally, for categories R2 and RC2, both algorithms seem to perform equally well, since there is no visual significant difference, thus more details are required for this analysis.

Table 5.8 presents the averages $\overline{\mathsf{M}}_{\mathsf{D}}$, over all instances within each problem category, of the convergence metric applied to the solutions from algorithms NSGA-II, MOEA-ERD, and MOEA-RD, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. In this respect, results from MOEA-ERD and MOEA-RD are closer to the reference set than those from NSGA-II for all categories, except for C2, and they perform significantly better than NSGA-II in many of the instances. According to this metric, NSGA-II

Algorithm	Compares	C1		C2		R1		R2		RC1		RC2	
NSGA-II	MOEA-ERD	76.91	(0)	87.34	(0)	65.90	(0)	78.76	(0)	68.83	(0)	80.58	(0)
	MOEA-RD		(0)		(0)		(0)		(0)		(0)		(0)
MOEA-ERD	NSGA-II	77.03	(3)	87.34	(0)	66.40	(12)	78.89	(5)	69.75	(8)	80.81	(4)
	MOEA-RD		(0)		(0)		(3)		(1)		(2)		(0)
MOEA-RD	NSGA-II	77.01	(3)	87.34	(0)	66.37	(12)	78.90	(6)	69.60	(8)	80.73	(3)
	MOEA-ERD		(0)		(0)		(1)		(0)		(0)		(0)

Table 5.9: Averages $\overline{\mathsf{M}}_{\mathsf{H}}$ ($\times 10^{-3}$) over instance categories for the solutions obtained with NSGA-II, MOEA-ERD, and MOEA-RD. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

never performs significantly better than MOEA-RD. MOEA-ERD and MOEA-RD perform similarly well, however, MOEA-RD significantly improves MOEA-ERD in more instances in category R2.

Hypervolume values were computed using the method followed to compare BiEA and MOEA-RD. Table 5.9 presents the average $\overline{\mathsf{M}}_{\mathsf{H}}$, over the instances within each set category, of the hypervolume metric applied to the solutions from algorithms NSGA-II, MOEA-ERD, and MOEA-RD, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. We see that, although all three algorithms present similar average hypervolume for all categories, there is a significant difference between MOEA-ERD and MOEA-RD with NSGA-II for instances in five out of the six categories, while MOEA-ERD has a slightly better performance than MOEA-RD for few instances in categories R1, R2, and RC1.

To understand the differences, it is instructive to look at the population diversity preserved by algorithms NSGA-II and MOEA-RD. Figure 5.10 presents six plots, one for each instance category, showing the average population diversity on the vertical axis, along with the average hypervolume difference between both algorithms, as a function of the first 500 generations on the horizontal axis. It is noticeable that MOEA-RD preserves a higher diversity for categories R1, R2, RC1 and RC2,

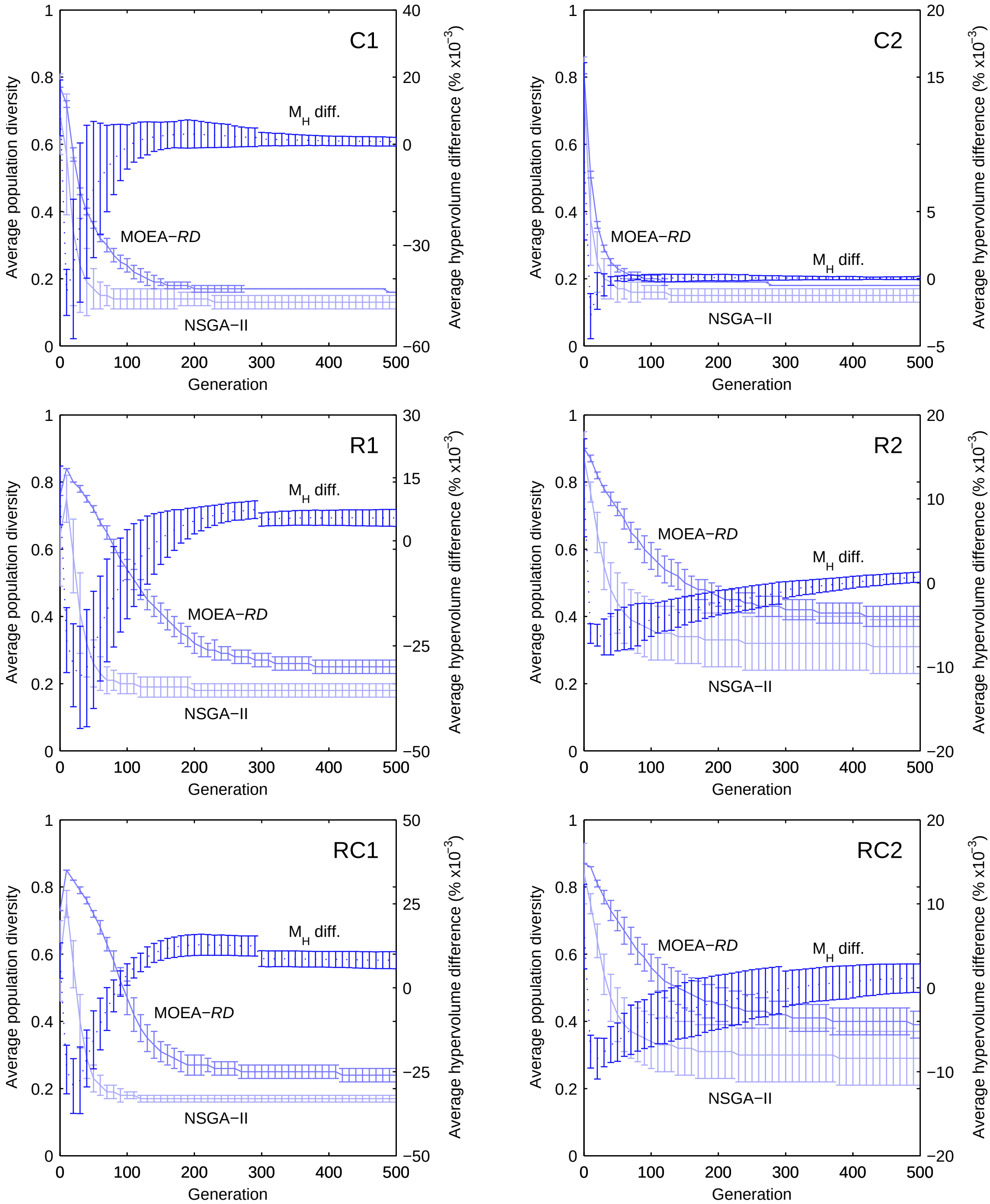


Figure 5.10: Average population diversity, over the instances in each set category, preserved by NSGA-II and MOEA-*RD*, and average hypervolume difference between both algorithms.

for which improvements over NSGA-II were observed. Moreover, the MOEA-*RD* diversities present a more gentle drop in all categories, except for C2, suggesting that MOEA-*RD* performs a wider exploration of the search space before settling on

Algorithm	C1	C2	R1	R2	RC1	RC2
NSGA-II	1.14 (0)	1.00 (0)	1.31 (0)	1.66 (1)	1.48 (0)	1.86 (0)
MOEA-*RD*	1.28 (2)	1.00 (0)	1.56 (5)	1.70 (0)	1.95 (5)	2.01 (0)

Table 5.10: Size of the Pareto approximations, averaged over instance categories, obtained with NSGA-II and MOEA-*RD*. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

its final solutions. This means that MOEA-*RD* is presenting the same behaviour of BiEA regarding the exploration of the search space. Additionally, we see that the hypervolume difference between both algorithms is negative for the first evolutionary generations, which means NSGA-II achieved better non-dominated solutions than those found by MOEA-*RD*. However, solutions from MOEA-*RD* improved in the last generations and the hypervolume difference finished to be positive. This means that the diversity preservation mechanism helped MOEA-*RD* to better explore the search space, obtain better non-dominated solutions than NSGA-II and actually achieve the objective it was designed for.

Table 5.10 presents the size of the Pareto approximations, averaged over instance categories, and the number of instances, in brackets, where the difference in the sizes is significant. We observe that, although the difference in the averages is small in all categories, there are a number of instances in categories C1, R1, and RC1, for which the non-dominated sets found by MOEA-*RD* are significantly larger than those from NSGA-II.

Lastly, Figure 5.11 shows the average execution time over a 500-generation period, corresponding to each instance category, of the algorithms analysed in this section, including those of BiEA and MOEA-E*RD* for comparison purposes. Notice that the vertical axis is in logarithmic scale. We observe that MOEA-*RD* and BiEA have similar execution times, while NSGA-II executes faster than both, 15% on average. MOEA-E*RD* still presents the slowest performance, using nearly 700% more time. The difference between BiEA and MOEA-*RD* with NSGA-II is due to the former

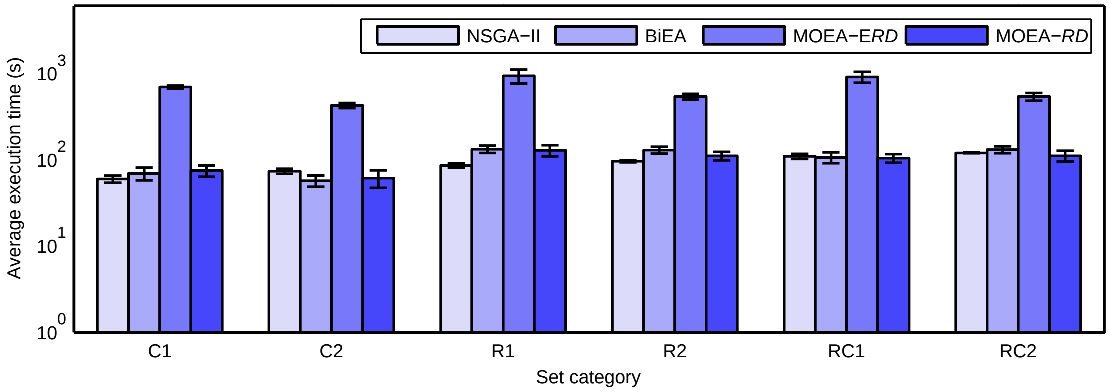


Figure 5.11: Execution time, averaged over instance categories, of the algorithms NSGA-II, BiEA, MOEA-ERD, and MOEA-RD.

compute Jaccard similarity in the survival selection stage, which has $O(N^2)$ time complexity, while NSGA-II calculates crowding distance, which is $O(N \log N)$.

We conclude that, with the exception of the instances in categories C1 and C2, which instances are not really multi-objective and for which the performance of NSGA-II and MOEA-RD proved to be similar, MOEA-RD has a significantly better performance than the observed by NSGA-II for all instances in categories R1 and RC1 and for many in sets R2 and RC2. We can attribute this superior performance to the broader exploration of the search space, which is achieved through the use of the solution similarity measure. Additionally, MOEA-RD set to use edit distance still presents a slightly better performance than set to use Jaccard similarity for few instances in categories R1 and RC1, and the latter presents a significantly better performance than the former for few instances in categories R2 and RC2.

With these results, presentation and analysis, we argue that the research objectives regarding the bi-objective solution of the VRPTW and its comparison against NSGA-II are achieved. Additionally, this also represents one of the main contributions of this thesis, since an appropriate performance analysis of the two evolutionary approaches, regarding the two criteria optimisation of the problem, has been accomplished.

5.3 Tri-objective optimisation of VRPs with Time Windows

To demonstrate the ease with which MOEA is able to address additional objectives, this section concentrates on the experimental analysis of the tri-objective optimisation of the VRPTW. The additional objective to be considered is the delivery time (T), $f_3(\mathcal{R})$ in (3.14) (see Section 3.2).

MOEA was run 30 times for each of the Solomon's benchmark instances size $N = 100$. The population diversity and the non-dominated solutions were recorded after each evolutionary generation for later analysis. The evolutionary parameters were set as in the bi-objective optimisation case.

Results were analysed from three different perspectives: (i) to analyse MOEA's performance with different objective settings and to test the effect of the optimisation of the additional delivery time objective, (ii) to compare the results from MOEA with those from previous studies, and (iii) to compare the results obtained by MOEA with those from NSGA-II using the multi-objective performance metrics.

5.3.1 Effect of the minimisation of the delivery time

The first aim is to analyse the performance of MOEA with different objective settings, in order to test the effect of the optimisation of the additional delivery time objective. The number of routes (R), travel distance (D) and delivery time (T) were first set to be minimised in pairs, giving three objective settings: RD, RT and DT. Then, all three of them together: RDT. For each of these settings, MOEA will be labelled as MOEA-X, where X refers to the objective setting. The results from all four algorithms with different objective settings are compared using the coverage, convergence and hypervolume performance metrics.

In order to apply the coverage metric, for each given instance and ordered pair of algorithms MOEA-X and MOEA-Y, $\mathsf{M_C}(\text{MOEA-X}_i, \text{MOEA-Y}_j), \forall\ i, j = 1, \ldots, 30$, that is 900 $\mathsf{M_C}$ values, were computed regarding all three objectives. MOEA-X$_i$ refers to the outcome set from the i-th execution of MOEA-X. After these computations, the $\mathsf{M_C}(\text{MOEA-X}_i, \text{MOEA-Y}_j)$ values were averaged ($\overline{\mathsf{M}}_\mathsf{C}$) over all the instances within each set category, and the resulting 900 values were collected together. These $\overline{\mathsf{M}}_\mathsf{C}$ values are presented in Figure 5.12 as box-and-whisker diagrams, which represent the distribution of the $\overline{\mathsf{M}}_\mathsf{C}$ values for each ordered pair of algorithms. Each cell, which range is 0 at the bottom and 1 at the top, contains six box-and-whisker plots corresponding to categories C1, C2, R1, R2, RC1 and RC2, from left to right, referring to the coverage of the algorithm in the corresponding column by the algorithm in the corresponding row.

From this figure we can make the following observations: All algorithms show a relatively high coverage of each other for categories C1 and C2, as indicated by the two leftmost boxes in each cell. For the rest of the instance categories, plots in the top row, the coverage of MOEA-RT, MOEA-DT and MOEA-RDT, by MOEA-RD, show that the median of the coverage $\widetilde{\mathsf{M}}_\mathsf{C}$ is low, with some boxes extending up to nearly 0.2. Similarly, we see in the MOEA-RT row that the coverage of the solutions from the other algorithms is extremely low, since the $\widetilde{\mathsf{M}}_\mathsf{C}$ for all categories in each cell, and actually the complete box, is very close to zero. The most interesting cases are MOEA-DT and MOEA-RDT, because it is clearly visible that their coverage of MOEA-RD and MOEA-RT is higher than the inverse cases. Between them, there is not an evident difference in the coverage of each other. These cases will be analysed later in more detail.

Regarding the convergence metric, for each algorithm and instance, the overall non-dominated solutions were extracted from the 30 Pareto approximations. Then, a composite non-dominated reference set $\mathcal{R}$ was found using the overall non-dominated

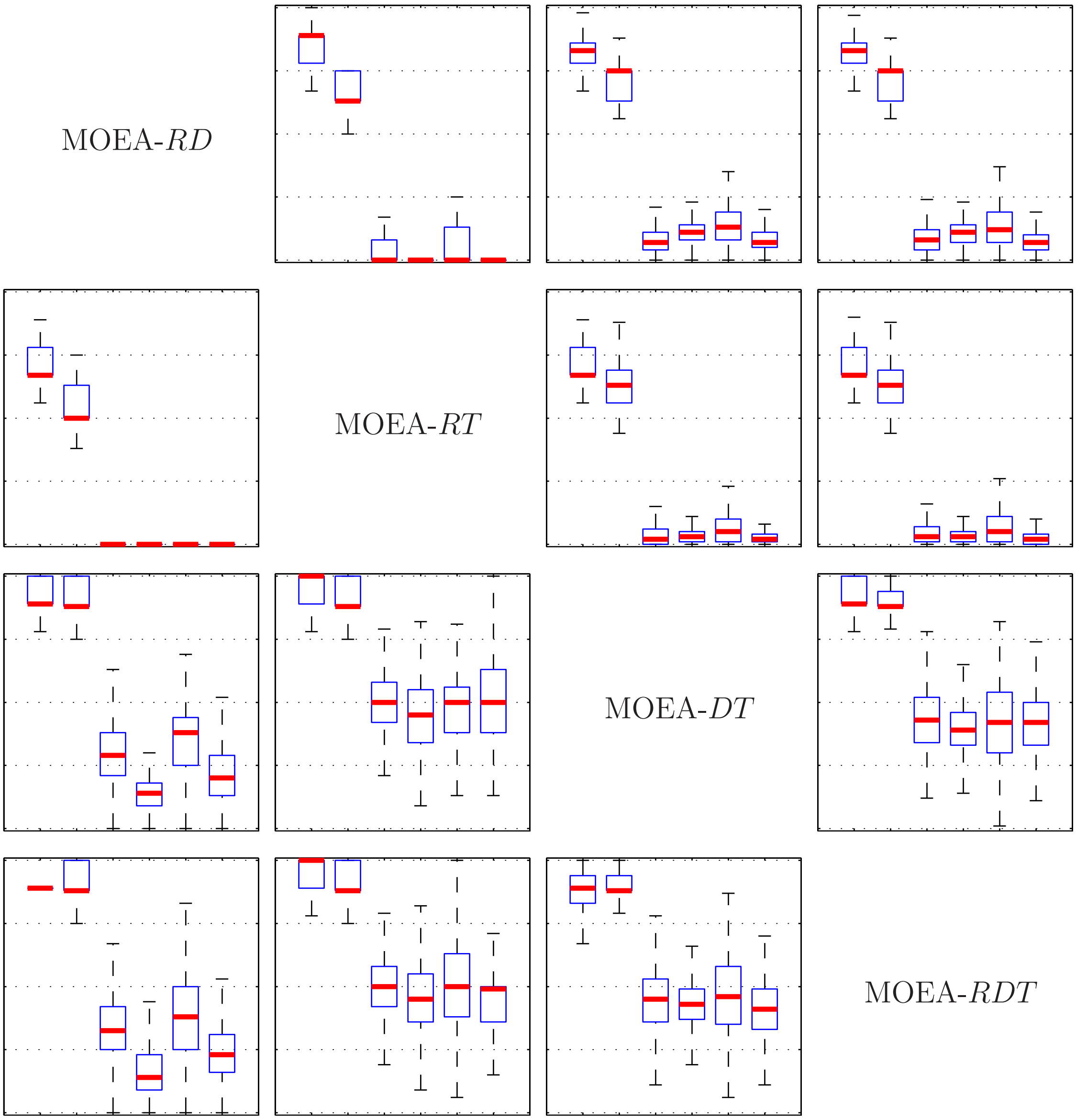


Figure 5.12: Box-and-whisker plots representing the distribution of the $\overline{\mathsf{M}}_{\mathsf{C}}$ values for each ordered pair of the results obtained by MOEA with objective settings *RD*, *RT*, *DT* and *RDT*.

sets from the four algorithms. Afterwards, $\mathsf{M}_{\mathsf{D}}(\text{MOEA-X}_i, \mathcal{R}), \forall\, i = 1, \ldots, 30$, were computed for each algorithm MOEA-X and instance, and normalised regarding the maximal solutions. These M_{D} values were grouped by the instances in each set category and the average $\overline{\mathsf{M}}_{\mathsf{D}}$ and corresponding standard error were calculated. Figure 5.13 presents these values as bar plots, where the height of the bars represent the $\overline{\mathsf{M}}_{\mathsf{D}}$ values. It is clear that, in general, solutions from MOEA-*DT* and MOEA-

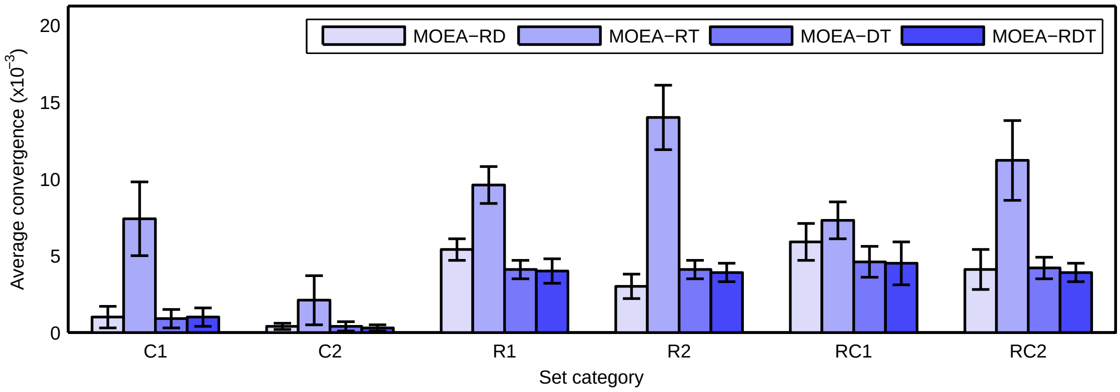


Figure 5.13: The $\mathsf{M_D}$ values, averaged over instance categories, for the results obtained by MOEA with objective settings RD, RT, DT and RDT.

RDT are the closest to $\mathcal{R}$, while those obtained by MOEA-RD and MOEA-RT are the farthest. Moreover, the former appear to have similar performance, thus more information is needed for analysis.

To help with the analysis of the coverage and convergence between algorithms MOEA-DT and MOEA-RDT, and to present the result of the hypervolume metric, which was computed as in the bi-objective case, Table 5.11 presents the averages $\overline{\mathsf{M}}_\mathsf{C}$, $\overline{\mathsf{M}}_\mathsf{D}$ and $\overline{\mathsf{M}}_\mathsf{H}$ over the instances within each problem category, as well as the numbers of instances, in brackets, for which there was significant improvement over the other approach. We observe that the differences of the $\overline{\mathsf{M}}_\mathsf{C}$, $\overline{\mathsf{M}}_\mathsf{D}$ and $\overline{\mathsf{M}}_\mathsf{H}$ between both algorithms are very narrow for all categories, however, $\overline{\mathsf{M}}_\mathsf{C}(\text{MOEA-}RDT, \text{MOEA-}DT)$ is still higher than $\overline{\mathsf{M}}_\mathsf{C}(\text{MOEA-}DT, \text{MOEA-}RDT)$ in four categories, and the coverage by MOEA-RDT is significantly larger than that by MOEA-DT in more instances. In respect to the convergence metric, solutions from MOEA-RDT are closer to $\mathcal{R}$ in five set categories than those from MOEA-DT, and the former are significantly closer than the latter in one instance. On the other hand, despite $\overline{\mathsf{M}}_\mathsf{D}(\text{MOEA-}RD, \mathcal{R}) < \overline{\mathsf{M}}_\mathsf{D}(\text{MOEA-}RDT, \mathcal{R})$ in the remaining category, MOEA-DT never performs significantly better than MOEA-RDT. Finally, solutions from

Algorithm	Metric	C1	C2	R1	R2	RC1	RC2
MOEA-*DT*	$\overline{\mathsf{M}}_\mathsf{C}$	0.91 (2)	0.88 (2)	0.43 (4)	0.40 (4)	0.42 (2)	0.42 (3)
	$\overline{\mathsf{M}}_\mathsf{D}(\times 10^{-3})$	0.90 (0)	0.40 (0)	4.10 (0)	4.10 (0)	4.60 (0)	4.20 (0)
	$\overline{\mathsf{M}}_\mathsf{H}(\times 10^{-2})$	77.01 (0)	87.34 (0)	65.33 (0)	77.29 (0)	68.68 (0)	78.74 (0)
MOEA-*RDT*	$\overline{\mathsf{M}}_\mathsf{C}$	0.89 (2)	0.89 (1)	0.44 (5)	0.43 (5)	0.46 (6)	0.41 (3)
	$\overline{\mathsf{M}}_\mathsf{D}(\times 10^{-3})$	1.00 (0)	0.30 (0)	4.00 (0)	3.90 (0)	4.50 (0)	3.90 (1)
	$\overline{\mathsf{M}}_\mathsf{H}(\times 10^{-2})$	77.00 (0)	87.34 (0)	65.37 (2)	77.30 (1)	68.75 (1)	78.78 (0)

Table 5.11: Averages $\overline{\mathsf{M}}_\mathsf{C}$, $\overline{\mathsf{M}}_\mathsf{D}$, and $\overline{\mathsf{M}}_\mathsf{H}$ over instance categories for the solutions obtained with MOEA-*DT* and MOEA-*RDT*. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

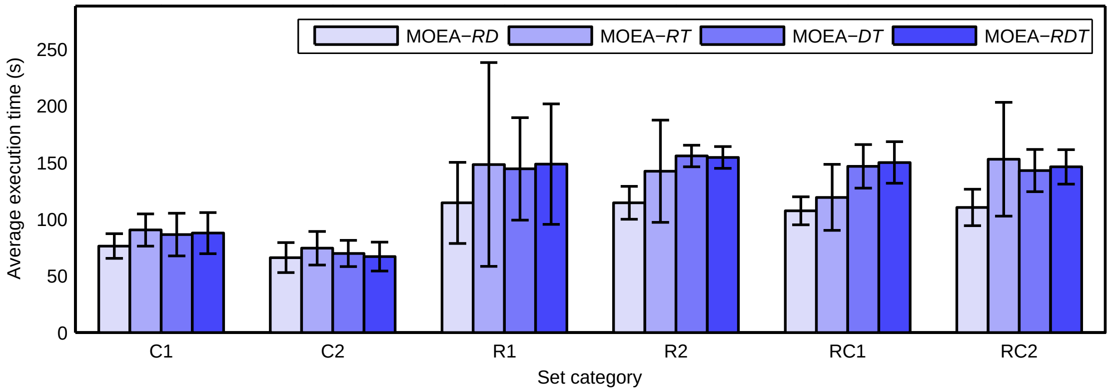


Figure 5.14: Execution time, averaged over instance categories, of MOEA with objective settings *RD*, *RT*, *DT*, and *RDT*.

MOEA-*RDT* delimit a moderately larger hypervolume that those from MOEA-*DT* in four categories, and the difference is significant in four instances.

Figure 5.14 shows the average execution time, over the instances in each set category, of MOEA with the four different objective settings. We notice that MOEA-*RD* executes quicker than the other three algorithms, running approximately 23% faster. This difference is a consequence of the $O(FM^2)$ non-dominated sort, where F is the number of objective functions. Since MOEA-*RDT* considers three objective functions, its increased execution time is clear. However, the other three algorithms consider two objectives. The difference in these cases is the number of solutions M in each front. Table 5.12 presents the size of the Pareto approximations averaged over instance category. We notice that they all present similar approximation sizes

Algorithm	C1	C2	R1	R2	RC1	RC2
MOEA-*RD*	1.28	1.00	1.56	1.7	1.95	2.01
MOEA-*RT*	1.21	1.15	20.34	43.22	21.45	53.31
MOEA-*DT*	1.80	1.43	23.56	19.56	22.25	20.60
MOEA-*RDT*	1.91	1.47	22.57	18.72	23.28	21.21

Table 5.12: Size of the Pareto approximations, averaged over instance categories, obtained by MOEA with objective settings RD, RT, DT, and RDT.

for categories C1 and C2, which corresponds to similar execution times. For the remaining categories, however, the size of the Pareto approximations from MOEA-RT and MOEA-DT is larger than that from MOEA-RD and so they are their execution times.

Considering the results from the three performance metrics, we conclude that algorithms MOEA-DT and MOEA-RDT perform significantly better than MOEA-RD and MOEA-RT, since solutions from the former have a higher coverage of those from the latter, are closer to the reference sets, and delimit a wider objective space. Between them, MOEA-RDT and MOEA-DT, despite their solutions are equally distant from the reference sets, solutions from MOEA-RDT have a significantly larger coverage of those from MOEA-DT in more instances than the opposite case, and they delimit a significantly wider objective space than the defined by the solutions from MOEA-DT. These results indicate that setting MOEA to minimise all three objectives does lead to find even better non-dominated solutions for many of the instances.

5.3.2 Comparison with previous studies

As noted earlier, previous multi-objective studies tackling the VRPTW have not presented their results in a proper multi-objective manner, however, it is still worth comparing results from MOEA-RDT in the traditional single-objective style with them.

Author		C1	C2	R1	R2	RC1	RC2	Total
$\min R$		10.00	3.00	11.91	2.73	11.50	3.25	405.00
		828.38	589.86	1212.73	952.67	1384.30	1108.52	57192.00
$\min D$		10.00	3.00	11.92	5.36	12.88	6.25	486.00
		828.38	589.86	1121.10	878.41	1341.67	1004.20	54779.02
Ombuki et al. [182] ($\min R$)		10.00	3.00	12.67	3.09	12.38	3.50	427.00
		828.48	590.60	1212.58	956.73	1379.87	1148.66	57484.35
Ombuki et al. [182] ($\min D$)		10.00	3.00	13.17	4.55	13.00	5.63	471.00
		828.48	590.60	1204.48	893.03	1384.95	1025.31	55740.33
Tan et al. [228]		10.00	3.00	12.92	3.55	12.38	4.25	441.00
		828.91	590.81	1187.35	951.74	1355.37	1068.26	56293.06
Ghoseiri and Ghannadpour [111] ($\min R$)		10.00	3.00	12.92	3.45	12.75	3.75	439.00
		828.38	591.49	1228.60	1033.53	1392.09	1162.40	58735.22
Ghoseiri and Ghannadpour [111] ($\min D$)		10.00	3.00	13.50	3.82	13.25	4.00	456.00
		828.38	591.49	1217.03	1049.62	1384.3	1157.41	58671.12
MOEA-RDT ($\min R$)		10.00	3.00	12.33	3.00	12.00	3.38	418.00
		829.07	589.93	1256.36	1097.01	1411.45	1335.52	61300.28
MOEA-RDT ($\min D$)		10.00	3.00	12.83	3.82	12.63	4.38	446.00
		828.38	589.86	1191.30	916.32	1349.24	1060.80	55829.68
% diff. R		0.00	0.00	3.55	9.89	4.35	3.85	3.21
% diff. D		0.00	0.00	6.26	4.32	0.56	5.64	1.92

Table 5.13: Number of routes and travel distance, averaged over categories, for the best solutions found by previous studies and by MOEA-RDT.

This comparison is going to be made in the same manner as in the bi-objective case, that is, the best results from the literature will be the reference to those from MOEA-RDT. Table 5.13 presents the average results, over the instances in each set category, from MOEA-RDT, as well as the best average results from past studies and those from previous multi-objective approaches. The format of this table is similar to that of Table 5.6.

We can observe that MOEA-RDT presents a similar performance to that obtained with MOEA-RD, in the sense that, with respect to the number of routes, MOEA-RDT ($\min R$) obtained the best-known results for all instances in categories C1 and C2. For category R2, results from MOEA-RDT are no more than 10% above the best-known, while for the remainig categories and in total the difference is below

4.4%. Concerning the travel distance, MOEA-RDT ($\min D$) obtained the best-known results for all instances in categories C1 and C2. For categories R1, R2, and RC2, results from MOEA-RDT are between 4.3% and 6.3% above the best known, while for category RC1 and in total, they are no more than 2% higher.

Taking into account only the multi-objective studies, we see that MOEA-RDT ($\min D$) found solutions similar to those of Tan et al. [228] for all categories, in the sense that they have shorter travel distances, though with a slightly increased number of routes. In the same context, MOEA-RDT found solutions with corresponding smaller number of routes and shorter travel distance than those from the approach of Ombuki et al. [182] for categories R1 and RC1, and than the achieved by Ghoseiri and Ghannadpour [111] for all categories, except RC2, and in total.

These results show that, overall, MOEA-RDT maintains its level of performance, as it is still comparable or better in this respect than previously published multi-objective approaches.

5.3.3 Comparison with NSGA-II

In the same manner as in the bi-objective scenario, an additional series of experiments with MOEA-RDT set to use edit distance (MOEA-ERDT) was performed, and these results, along with those from MOEA-RDT, are compared to those from NSGA-II by means of the coverage, convergence, and hypervolume performance metrics. The process of computing $\mathsf{M_C}$, $\mathsf{M_D}$, and $\mathsf{M_H}$ was identical to the followed in that case, with the difference that now the three objectives are considered.

Figure 5.15 presents six series of box-and-whisker plots to display the resultant distributions of the $\mathsf{M_C}$ values. For each instance there are two boxes: the one on the left represents $\mathsf{M_C}$(NSGA-II, MOEA-RDT), and the one on the right $\mathsf{M_C}$(MOEA-RDT, NSGA-II). Both algorithms perform like in the bi-objective case, in the sense that

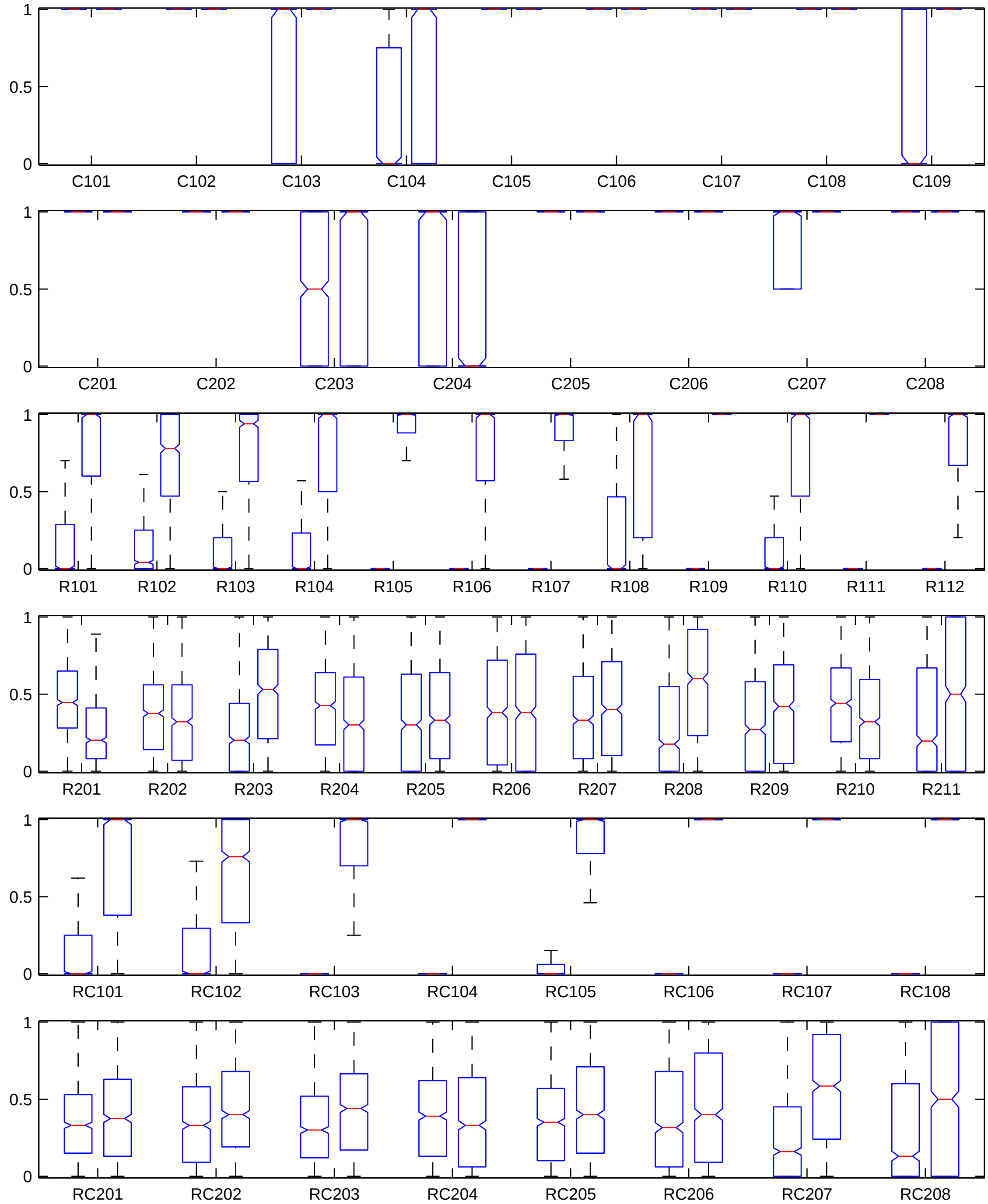


Figure 5.15: Box-and-whisker plots representing the distribution of the $\mathsf{M_C}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_C}$(NSGA-II, MOEA-*RDT*), and the one on the right $\mathsf{M_C}$(MOEA-*RDT*, NSGA-II).

they present similar coverage of each other for instances in categories C1 and C2, with the exception of three instances in which $\widetilde{\mathsf{M}}_\mathsf{C}(\text{MOEA-}RDT, \text{NSGA-II})$ is greater than $\widetilde{\mathsf{M}}_\mathsf{C}(\text{NSGA-II}, \text{MOEA-}RDT)$, and in one the latter is greater than the former.

Algorithm	Covers	C1	C2	R1	R2	RC1	RC2
NSGA-II	MOEA-ERDT	0.81 (0)	0.88 (2)	0.18 (0)	0.48 (10)	0.09 (0)	0.43 (5)
	MOEA-RDT	0.81 (0)	0.87 (2)	0.14 (0)	0.37 (4)	0.13 (0)	0.35 (0)
MOEA-ERDT	NSGA-II	0.94 (4)	0.87 (1)	0.72 (11)	0.32 (0)	0.85 (8)	0.36 (2)
	MOEA-RDT	0.91 (1)	0.87 (0)	0.38 (4)	0.31 (0)	0.51 (6)	0.32 (0)
MOEA-RDT	NSGA-II	0.93 (4)	0.88 (2)	0.78 (12)	0.41 (5)	0.80 (8)	0.45 (7)
	MOEA-ERDT	0.90 (2)	0.89 (2)	0.50 (8)	0.53 (11)	0.38 (2)	0.51 (7)

Table 5.14: Averages $\overline{\mathsf{M}}_{\mathsf{C}}$ over instance categories for the solutions obtained with NSGA-II, MOEA-ERDT, and MOEA-RDT. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

For categories R1 and RC1, solutions in the non-dominated sets found by MOEA-RDT widely cover those obtained by NSGA-II in all instances, and solutions from NSGA-II narrowly cover those of MOEA-RDT in very few instances. Categories R2 and RC2 will be analysed later since it is not visually clear what the difference in their performance is.

Table 5.14 presents the averages $\overline{\mathsf{M}}_{\mathsf{C}}$, over the instances within each problem category, of the coverage metric applied to every ordered pair of the algorithms NSGA-II, MOEA-ERDT, and MOEA-RDT, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. We can observe that, in all six categories, $\overline{\mathsf{M}}_{\mathsf{C}}(\text{MOEA-}RDT, \text{NSGA-II})$ is greater than $\overline{\mathsf{M}}_{\mathsf{C}}(\text{NSGA-II}, \text{MOEA-}RDT)$. Furthermore, the number of instances in which solutions from MOEA-RDT significantly cover those from NSGA-II is much higher than that for the inverse case. On the other hand, we see that in three categories, namely C1, R1, and RC1, $\overline{\mathsf{M}}_{\mathsf{C}}(\text{MOEA-E}RDT, \text{NSGA-II}) > \overline{\mathsf{M}}_{\mathsf{C}}(\text{NSGA-II}, \text{MOEA-E}RDT)$, for which solutions from MOEA-ERDT significantly cover those from NSGA-II, and the latter cover the former in the remaining three categories, i.e. C2, R2, and RC2. Solutions from MOEA-RDT significantly cover those from MOEA-ERDT in many of the instances in categories C2, R1, R2, and RC2, and the latter cover the former in more instances only in category RC1.

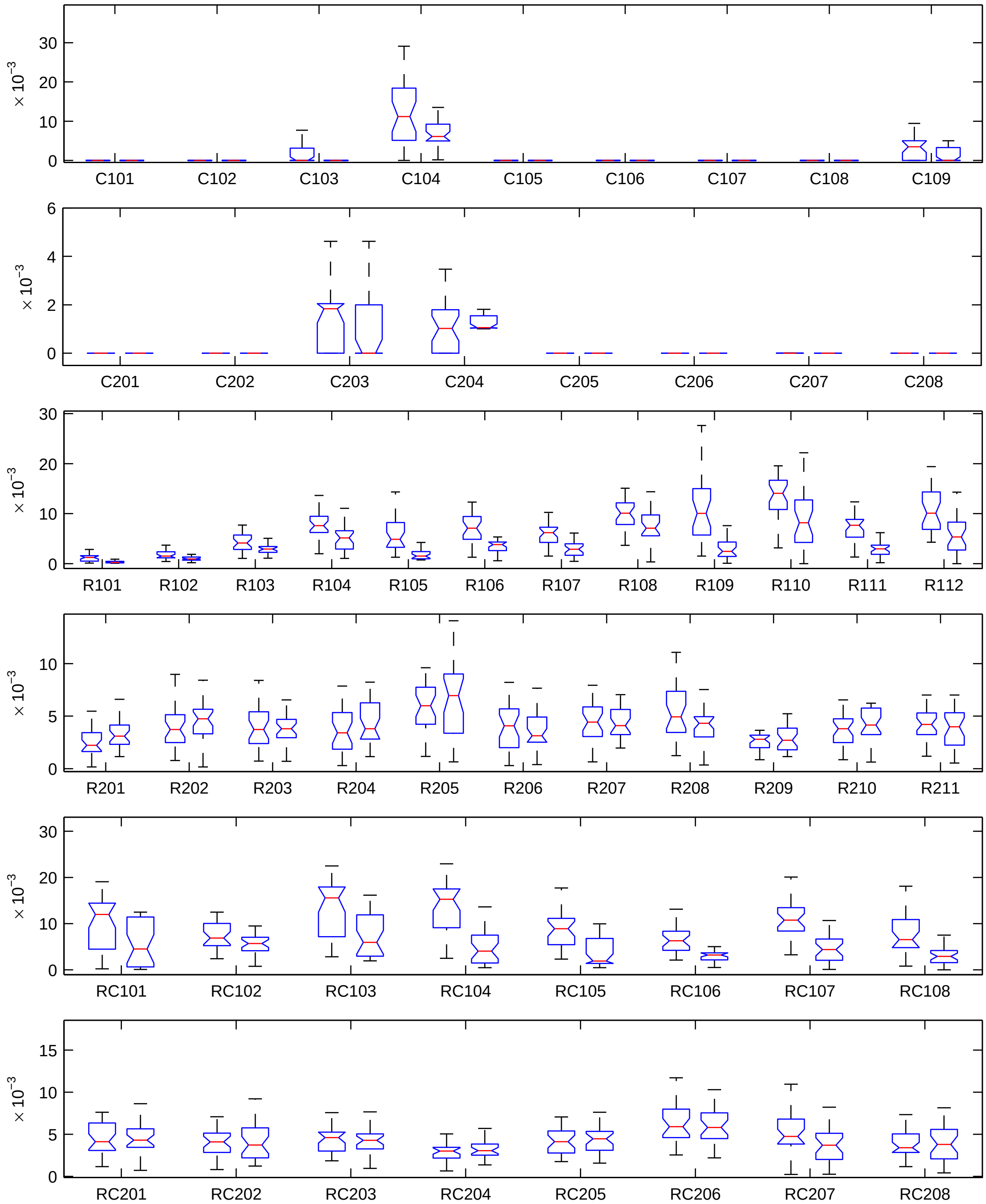


Figure 5.16: Box-and-whisker plots representing the distribution of the $\mathsf{M_D}$ values. For each instance are shown two bars: the one on the left depicting $\mathsf{M_D}(\text{NSGA-II}, \mathcal{R})$, and the one on the right $\mathsf{M_D}(\text{MOEA-}\mathit{RDT}, \mathcal{R})$.

Similarly to the analysis done in the bi-objective optimisation, Figure 5.16 shows the distribution of the $\mathsf{M_D}$ values. Again, both algorithms present similar performance for the clustered instances, with some exceptions where $\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{MOEA-}\mathit{RDT}, \mathcal{R}) <$

Algorithm	Compares	C1	C2	R1	R2	RC1	RC2
NSGA-II	MOEA-E*RDT*	1.97 (0)	0.33 (0)	7.33 (0)	4.06 (8)	9.90 (0)	4.48 (3)
	MOEA-*RDT*	(0)	(0)	(0)	(0)	(0)	(0)
MOEA-E*RDT*	NSGA-II	1.00 (2)	0.34 (0)	4.93 (9)	5.46 (0)	3.93 (8)	5.38 (0)
	MOEA-*RDT*	(0)	(0)	(0)	(2)	(0)	(0)
MOEA-*RDT*	NSGA-II	0.97 (3)	0.30 (0)	3.89 (12)	4.22 (1)	4.76 (8)	4.30 (1)
	MOEA-E*RDT*	(0)	(0)	(6)	(7)	(0)	(4)

Table 5.15: Averages $\overline{\mathsf{M}}_{\mathsf{D}}$ $(\times 10^{-3})$ over instance categories for the solutions obtained with NSGA-II, MOEA-ERDT, and MOEA-RDT. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

$\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{NSGA-II}, \mathcal{R})$ and the notches in the boxes do not overlap. Solutions from MOEA-RDT appear to be closer to $\mathcal{R}$ for the majority of the instances in sets R1 and RC1 than those from NSGA-II are, since $\widetilde{\mathsf{M}}_{\mathsf{D}}(\text{MOEA-}RDT, \mathcal{R}) < \widetilde{\mathsf{M}}_{\mathsf{D}}(\text{NSGA-II}, \mathcal{R})$ and the notches, and for some instances the complete boxes, do not overlap. Regarding set categories R2 and RC2, the plots do not show a clear visual difference in their performance.

Table 5.15 presents the averages $\overline{\mathsf{M}}_{\mathsf{D}}$, over the instances within each problem category, of the convergence metric applied to the solutions from NSGA-II, MOEA-ERDT, and MOEA-RDT, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. We observe that solutions from MOEA-ERDT and MOEA-RDT are significantly closer to the reference set $\mathcal{R}$ than those from NSGA-II for nearly all instances in categories C1, R1 and RC1, and solutions from NSGA-II are significantly closer to $\mathcal{R}$ than those from MOEA-ERDT for many instances in categories R2 and RC2, though never significantly closer than those from MOEA-RDT. Solutions from MOEA-RDT are significantly closer to the reference set than those from MOEA-ERDT for many instances in categories R1, R2, and RC2, and the latter are never significantly closer to $\mathcal{R}$ than the former.

Algorithm	Compares	C1	C2	R1	R2	RC1	RC2
NSGA-II	MOEA-E*RDT*	76.91 (0)	87.34 (0)	64.88 (1)	77.35 (6)	68.09 (0)	78.78 (3)
	MOEA-*RDT*	(0)	(0)	(0)	(1)	(0)	(0)
MOEA-E*RDT*	NSGA-II	77.00 (2)	87.34 (0)	65.27 (8)	77.02 (0)	68.79 (7)	78.54 (2)
	MOEA-*RDT*	(0)	(0)	(0)	(0)	(0)	(0)
MOEA-*RDT*	NSGA-II	77.00 (3)	87.34 (0)	65.37 (11)	77.30 (1)	68.75 (8)	78.78 (1)
	MOEA-E*RDT*	(0)	(0)	(4)	(4)	(1)	(1)

Table 5.16: Averages $\overline{\mathsf{M}}_{\mathsf{H}}$ ($\times 10^{-2}$) over instance categories for the solutions obtained with NSGA-II, MOEA-E*RDT*, and MOEA-*RDT*. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

The hypervolume metric was also computed in order to compare these three algorithms. Table 5.16 presents the average $\overline{\mathsf{M}}_{\mathsf{H}}$, over the instances within each set category, of the hypervolume metric applied to the solutions from algorithms NSGA-II, MOEA-E*RDT*, and MOEA-*RDT*, along with the numbers of instances, in brackets, for which there was significant improvement over the other approach. We observe that the average hypervolume delimited by the solutions from the three algorithms is very similar, however there are significant differences. MOEA-E*RDT* and MOEA-*RDT* present a significantly larger hypervolume than NSGA-II for many instances in categories C1, R1 and RC1, while that of NSGA-II is significantly larger that that of MOEA-E*RDT* in categories R2 and RC2. Solutions from MOEA-*RDT* define a significantly larger hypervolume than those from MOEA-E*RDT* for several instances in categories R1, R2, RC1, and RC2 .

To exemplify how the overall Pareto approximations are distributed in the objective space, Figure 5.17 presents these for four instances, one from each of the categories R1, R2, RC1, and RC2. We see that, in general, solutions from both algorithms are evenly distributed over the travel distance and delivery time dimensions, and over a small region in the number of routes. In the cases of the instances R105 and RC105, it is clear that solutions from MOEA-*RDT* dominate those from NSGA-II, and for

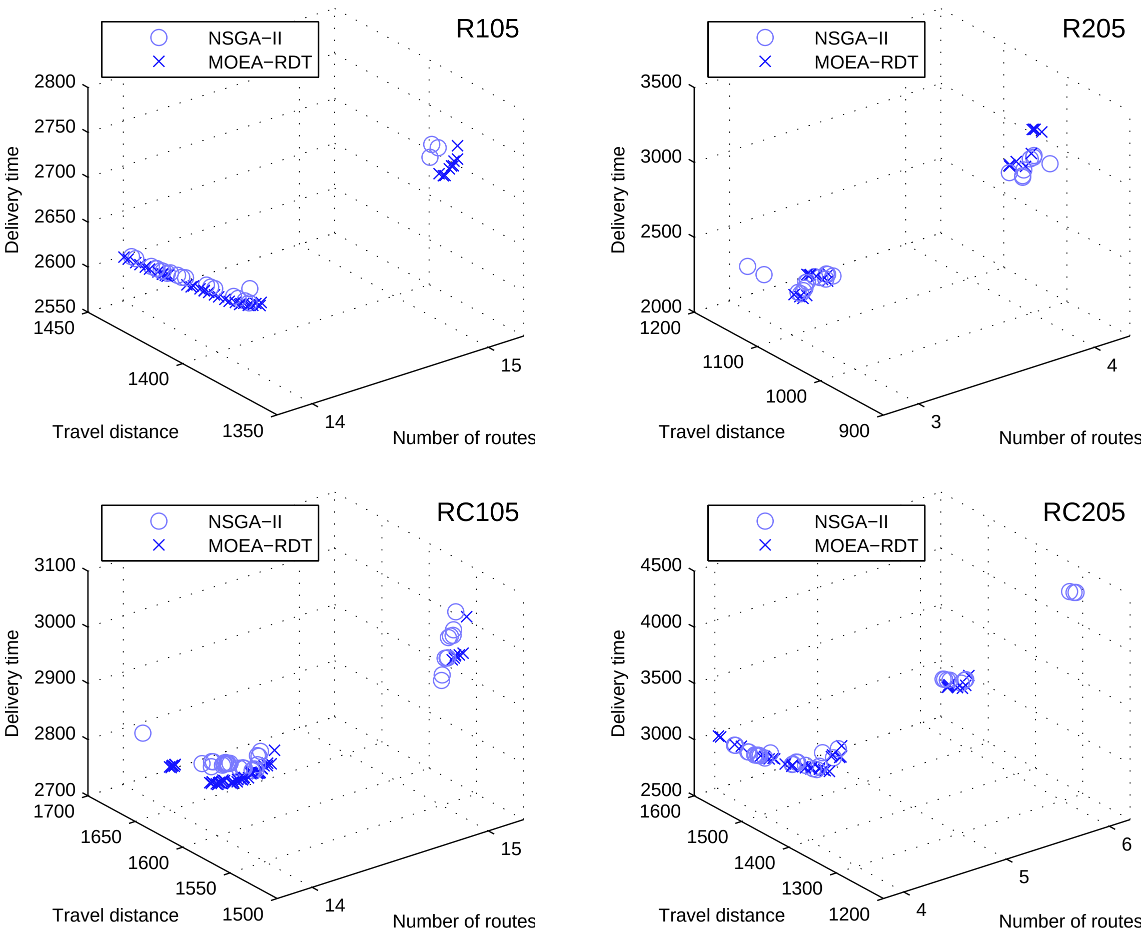


Figure 5.17: Overall Pareto approximations found by NSGA-II and MOEA-*RDT* for one instance in each of the categories R1, R2, RC1, and RC2.

instance RC205, NSGA-II found solutions with six routes, however, MOEA-*RDT* was not able to find solutions with that number of routes.

Additionally, Figure 5.18 presents the average population diversity over generations, along with the average hypervolume difference, for each instance category. Similarly to the bi-objective optimisation case, we observe that MOEA-*RDT* preserves a higher diversity for categories R1, R2, RC1 and RC2, for which also present a more gentle drop. We argue that this behaviour is due to that MOEA-*RDT* is performing a wider exploration of the search space, which is achieved by utilising the solution similarity measure. Moreover, the hypervolume difference between the two algorithms starts negative in all cases, which means that NSGA-II found better

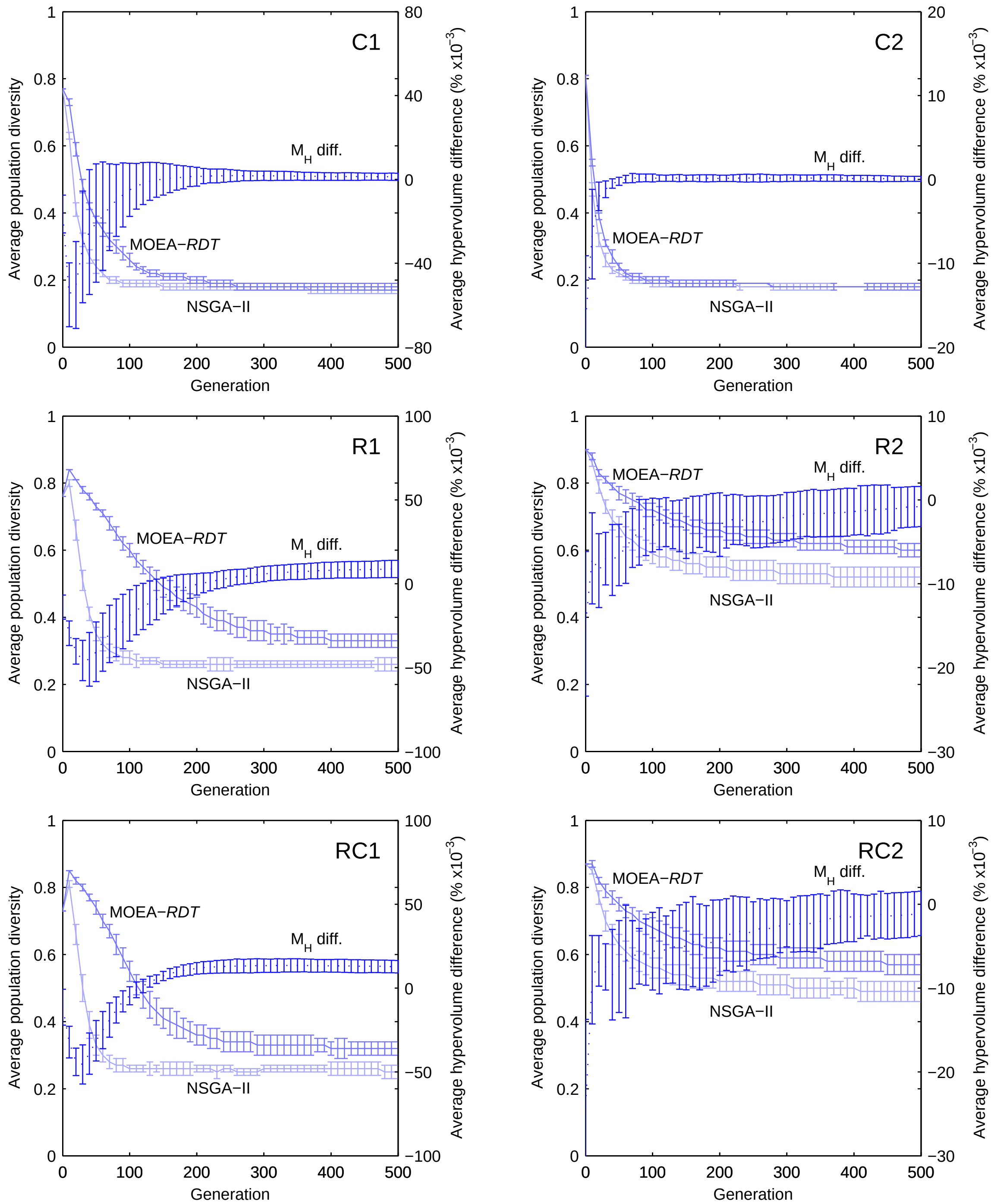


Figure 5.18: Average population diversity, over the instances in each set category, preserved by NSGA-II and MOEA-*RDT*, and average hypervolume difference between both algorithms.

solutions than those obtained by MOEA-*RDT*. However, this difference decreases in later generations, which means that MOEA-*RDT* is achieving better solutions and,

Algorithm	C1	C2	R1	R2	RC1	RC2
NSGA-II	1.99 (0)	1.44 (0)	18.06 (0)	17.85 (0)	18.75 (0)	19.09 (0)
MOEA-*RDT*	1.91 (0)	1.47 (0)	22.57 (4)	18.72 (0)	23.28 (3)	21.21 (1)

Table 5.17: Size of the Pareto approximations, averaged over instance categories, obtained with NSGA-II and MOEA-*RD*. Shown in brackets are the number of instances for which the result is significantly better than the other approach.

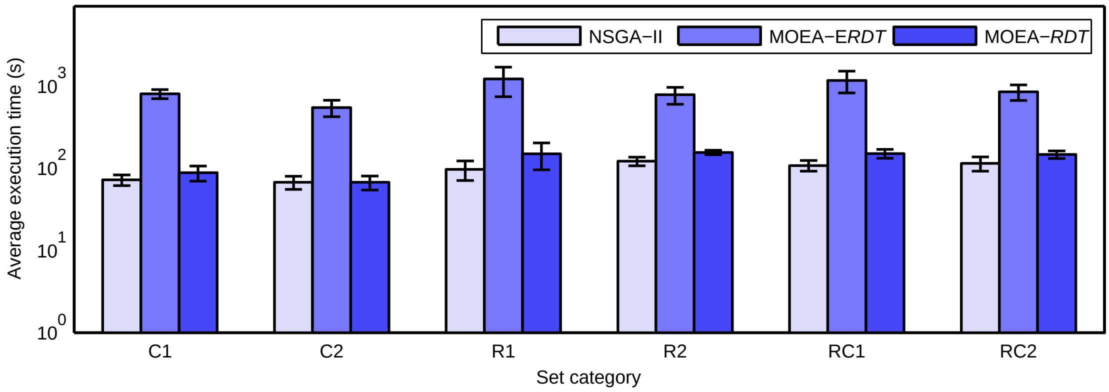


Figure 5.19: Execution time, averaged over instance categories, of the algorithms NSGA-II, MOEA-E*RDT*, and MOEA-*RDT*.

in the case of the categories R1 and RC1, finally improve those found by NSGA-II. In the case of categories R2 and RC2, the difference ended up being nearly zero.

Table 5.17 presents the size of the Pareto approximations, averaged over instance categories, and the number of instances, in brackets, where the difference in the sizes is significant. We can see that, for instances in categories R1, R2, RC1, and RC2, the average size of the Pareto approximations obtained with MOEA-*RDT* is larger than that of the non-dominated sets found by NSGA-II. Furthermore, the difference of the sizes is significant for some instances in categories R1, RC1, and RC2

To finalise the analysis of the tri-objective optimisation, the average execution time, over the instances in each set category, of NSGA-II, MOEA-E*RDT*, and MOEA-*RDT*, is depicted in Figure 5.19. Notice that the vertical axis is in logarithmic scale. We see that MOEA-*RDT* executes slower than NSGA-II, being this difference of approximately 30%. The execution time of MOEA-E*RDT* remians being the largest, with nearly one order of magnitud in this case. The difference in execution time between MOEA-RDT and NSGA-II is, as in the bi-objective case, due to the different time complexity in the survival selection stage, i.e. the $O(M \log M)$ of NSGA-II's AssignCrowdingDistance() and the $O(M^2)$ of ComputeJaccardSimilarity() in MOEA-RDT.

From this study we conclude that MOEA-RDT performs significantly better than NSGA-II for many of the Solomon's instances, particularly for all instances in categories R1 and RC1, where MOEA-RDT achieved a significantly improved performance. This is due to the wider exploration of the search space MOEA-RDT achieves. On the other hand, MOEA-RDT has a better performance when the Jaccard similarity is set instead of the edit distance, since, in the second case, the algorithm was surpassed by NSGA-II in two of the categories. Actually, when the edit distance is set, MOEA's performance appears to deteriorate while more objectives are to be optimised.

After having analysed the results from the bi-objective and tri-objective optimisations, we observe that MOEA showed a significantly improved performance over NSGA-II for many of the test instances used, particularly for all instances in categories R1 and RC1. Let us try to argue the reason for this behaviour.

The characteristics of the test instances used in this investigation were described in Section 3.2.1, and was stated that the attributes of the instances in categories R2 and RC2 make them have a wider search space than those in categories C1, C2, R1, and RC1. Thus, we conjecture that, although MOEA is preserving a higher population diversity, and exploring and exploiting a wider search space than NSGA-II, it is even harder for both algorithms to achieve improved solutions to instances in the former categories than in the latter. An investigation regarding this subject, which considers an in-depth analysis in order to find the reasons for this situation, is proposed as a key future research topic.

With these results and analysis, we claim that the research objectives regarding the multi-objective solution of the VRPTW, and its comparison with NSGA-II by means of appropriate quality indicators, is completely fulfilled. Additionally, the analysis carried out in this section constitute one of the main contributions of this thesis, which is that an appropriate multi-objective performance analysis of the two evolutionary approaches, concerning the three criteria optimisation of the problem, has been achieved.

The bi-objective and part of the tri-objective analyses were published as a journal paper in Computers & Operations Research [103]. The remaining tri-objective analysis was accepted for publication and presentation in another top conference, the 11th International Conference on Parallel Problem Solving From Nature (PPSN XI) [102].

5.4 Bi-objective optimisation of Capacitated VRPs

This section demonstrates how MOEA can be applied to different variants of the VRP, in this case, particularly to the CVRP. As was reviewed in Chapter 3.1, the objective of the Capacitated VRP is exclusively related to the minimisation of the number of routes, and for a given number of routes, the minimisation of the travel distance. In this section, the solution of this problem is considered from a bi-objective perspective, simultaneously optimising the number of routes, $f_1(\mathcal{R})$ in (3.4), and the travel distance, $f_2(\mathcal{R})$ in (3.5), subject to the capacity constraint (3.6) and the maximum route length constraint (3.7). Moreover, these objectives were considered to have the same priority.

MOEA-*RD* was tested on the Christofides et al.'s and Rochat and Taillard's benchmark sets, which instances were suitably modified so that customers could be visited at any time, as long as the length restriction, if any, is met. MOEA-*RD* was executed

30 times for every instance and the Pareto approximation was recorded after each run. The evolutionary parameters were kept as for the bi-objective and tri-objective optimisation of the VRPTW.

5.4.1 Solution of the Christofides et al.'s benchmark set

Results for the Christofides et al.'s test instances are shown in Table 5.18, where the best-known result, the difference between results from previous studies and the best-known, and the best and average best results from MOEA-*RD* are presented. The best-known results and from previous studies were taken from Table 3.3 (page 69). Along with the results from MOEA-*RD*, the percentage difference (%) between these and the best-known results are also shown. The last four rows show the averages of the percentage difference over all instances, and corresponding standard deviation, and over instances size $N \leq 120$, respectively.

Analysing this table, we can see that MOEA-*RD* was able to find the best-known solutions for three of the instances, namely vrnpc1, vrpnc6, and vrpnc7. We can also see that the best solutions obtained by MOEA-*RD* are no more than 0.53% higher than the best-known, considering instances of size $N \leq 120$, however, the overall difference increases up to 4.5%. Regarding the average of the best results over the 30 repetitions, the difference between them and the best-known is of 3.92% for instances of size $N \leq 120$, and 9.18% overall.

Excluding the instances for which MOEA-*RD* found the best-known solution, results from MOEA-*RD* are higher than those presented in previous studies, with the exception of those from Bullnheimer et al. [31], which, for many instances, are the highest. Additionally, MOEA-*RD* found that instance vrpnc14 has conflicting objectives, since the best-known result do not dominate the solutions obtained.

Instance	Best known	T	GHL	BHS	TV	P04	BA	MM	MB	NB	P09	MOEA-*RD*			
												Best	%	Avg.	%
	Best	T	GHL	BHS	TV	P	BA	MM	MB	NB	P				
vrpnc1	524.61	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	524.61	0.00	537.53	2.46
vrpnc2	835.26	0.00	0.01	4.23	0.40	0.00	0.43	0.40	0.00	0.00	0.00	843.92	1.04	862.25	3.23
vrpnc3	826.14	0.00	0.00	6.45	0.29	0.00	0.40	0.62	0.00	0.00	0.00	839.35	1.60	865.27	4.74
vrpnc4	1028.42	0.00	0.26	11.57	0.47	0.20	0.62	1.78	0.00	0.00	0.10	1117.71	8.68	1162.84	13.07
vrpnc5	1291.29	0.58	1.55	14.10	2.09	0.39	2.83	2.15	0.00	0.01	0.22	1559.87	20.80	1778.72	37.75
vrpnc6	555.43	0.00	0.00	1.35	0.00	0.00	0.00			0.00	0.00	555.43	0.00	560.74	0.96
vrpnc7	909.68	0.00	0.00	4.23	1.21	0.00	0.00			0.00	0.00	909.68	0.00	924.28	1.60
vrpnc8	865.94	0.00	0.00	2.34	0.41	0.00	0.20			0.00	0.00	866.87	0.11	882.33	1.89
vrpnc9	1162.55	0.00	0.03	3.39	0.91	0.00	0.32			0.00	0.00	1204.68	3.62	1251.65	7.66
vrpnc10	1395.85	0.15	0.64	7.80	2.86	0.49	2.11			0.00	0.40	1737.65	24.49	1826.93	30.88
vrpnc11	1042.11	0.00	0.00	2.91	0.07	0.00	0.46	0.00	0.00	0.00	0.00	1047.04	0.47	1151.20	10.47
vrpnc12	819.56	0.00	0.00	0.05	0.00	0.00	0.00		0.00	0.00	0.00	821.36	0.22	857.58	4.64
vrpnc13	1541.14	0.00	0.31	3.20	0.28	0.11	0.34			0.00	0.28	1558.80	1.15	1587.02	2.98
vrpnc14	866.35	0.00	0.00	0.41	0.00	0.00	0.09			0.00	0.00	872.95	0.76	920.05	6.20
Average		0.05	0.20	4.43	0.64	0.09	0.56	0.83	0.00	0.00	0.07		4.50		9.18
St. Dev.		0.16	0.43	4.24	0.87	0.16	0.85	0.92	0.00	0.00	0.13		8.06		11.27
Average		0	0.03	2.52	0.27	0.01	0.19	0.25	0.00	0.00	0.03		0.53		3.92
St. Dev.		0	0.1	2.11	0.37	0.04	0.2	0.31	0.00	0.00	0.09		0.57		2.81

Author: BA: Baker and Ayechew [11] BHS: Bullnheimer et al. [31] GHL: Gendreau et al. [106] MB: Mester and Bräysy [170] MM: Morgan and Mumford [176] NB: Nagata and Bräysy [179] P04: Prins [195] P09: Prins [196] T: Taillard [225] TV: Toth and Vigo [239]

Table 5.18: Best-known results, results from previous studies, and best results from MOEA-*RD* for the Christofides et al.'s benchmark set.

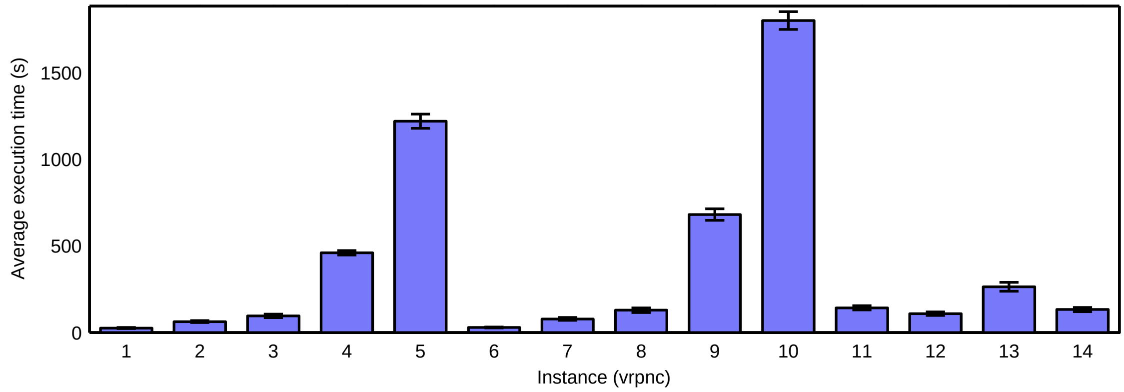


Figure 5.20: Execution time of MOEA-*RD* for the Christofides et al.'s benchmark set.

The average execution time of MOEA-*RD* for the Christofides et al. [37]'s instances is shown in Figure 5.20. Here we notice how the execution time increases for the different instance sizes. For example, instances vrpnc5 and vrpnc10 have 199 customers, for which MOEA-*RD* spends at least 4000% more time than for the smallest instances vrpnc1 and vrpnc6 with 50 customers.

5.4.2 Solution of the Rochat and Taillard's benchmark set

Results for the Rochat and Taillard's benchmark set are shown in Table 5.19, which has the same format as Table 5.18. The best-known and previous results were taken from Table 3.4 (page 69).

In this case, MOEA-*RD* was capable of finding the best-known solution for three out of the 12 instances, namely tai75a, tai75c, and tai75d, however, the average difference between the best results obtained and the best-known is of 1.28%, and if we consider only instances of size $N \leq 100$ the difference decreases down to 0.37%. Taking into account the average of the best solutions found in each of the 30 repetitions, these differences are of 3.01% and 1.81%, respectively.

Compared with the previous studies, we can observe that, with the exception of the instances size $N = 75$, MOEA-*RD* obtained solutions with the largest travel

Instance	Best known	RT	GTA	MB	MM	AD	NB	MOEA-*RD* Best	%	Avg.	%
tai75a	1618.36			0.00	0.00	0.00	0.00	1618.36	0.00	1629.24	0.67
tai75b	1344.62			0.00	0.00		0.00	1345.19	0.04	1358.78	1.05
tai75c	1291.01			0.00	0.00	0.00	0.00	1291.01	0.00	1325.31	2.66
tai75d	1365.42			0.00	0.00	0.00	0.00	1365.42	0.00	1380.22	1.08
tai100a	2041.34	0.32	0.00	0.00	0.65	0.32	0.00	2055.31	0.68	2079.16	1.85
tai100b	1939.90	0.04		0.00	0.02	0.02	0.00	1943.75	0.20	1969.25	1.51
tai100c	1406.20	0.09	0.00	0.00	0.00	0.39	0.00	1410.15	0.28	1430.81	1.75
tai100d	1580.46	0.05	0.05	0.05	0.69	0.24	0.00	1607.80	1.73	1643.49	3.99
tai150a	3055.23	0.51		0.00		0.04	0.00	3131.24	2.49	3201.61	4.79
tai150b	2656.47	2.90	0.00	2.68		2.87	2.66	2803.72	5.54	2872.58	8.14
tai150c	2341.84	0.96		0.05		0.95	0.72	2405.26	2.71	2469.76	5.46
tai150d	2645.39	0.67		0.00		0.35	0.00	2688.56	1.63	2730.47	3.22
Average		0.69	0.01	0.23	0.17	0.47	0.28		1.28		3.01
St. Dev.		0.95	0.02	0.77	0.31	0.85	0.78		1.68		2.23
Average		0.12	0.02	0.01	0.17	0.14	0.00		0.37		1.82
St. Dev.		0.13	0.03	0.02	0.31	0.17	0.00		0.60		1.07

Author: AD: Alba and Dorronsoro [5] MB: Mester and Bräysy [170] NB: Nagata and Bräysy [179]
GTA: Gambardella et al. [97] MM: Morgan and Mumford [176] RT: Rochat and Taillard [204]

Table 5.19: Best results from previous studies, and their percentage difference with the best-known results, for the Rochat and Taillard's benchmark set.

distances. Something important to mention is that, after analysing the Pareto approximations to each instance, it was discovered that they were comprised of one single solution, with the exception of those to instances tai75b and tai150c, which had two solutions.

Figure 5.21 presents the average execution time of MOEA-*RD* for the Rochat and Taillard [204]'s instances. We observe the same behaviour as for the Christofides et al. [37]'s test set, in the sense that for the largest instances, in this case tai150a through tai150d with 150 customers, MOEA-*RD* spends nearly 800% more time than for the smallest instances tai75a through tai75b with 75 customers.

In general, we conclude that the performance of MOEA-*RD* on instances of the CVRP is comparable with that of previous proposals for instances of up to 120 customers, for which the overall difference between the best results from MOEA-*RD* and the best-known results is, on average, 0.45%.

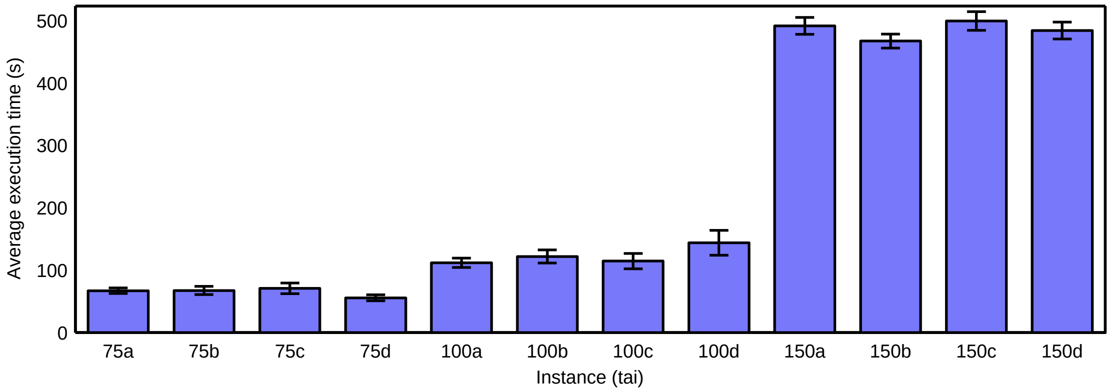


Figure 5.21: Execution time of MOEA-RD for the Rochat and Taillard's benchmark set.

This result and analysis achieve one the objectives of this investigation, which regards the design and development of a multi-objective EA for solving both the CVRP and VRPTW, and forms part of the main contributions of this research.

5.5 Summary

This chapter described the modifications made to the Bi-objective Evolutionary Algorithm (BiEA), presented in Chapter 4, in order to enhance its performance. These modifications are concentrated on the mutation stage: procedure and operators. A detailed analysis of the new algorithm's performance was also provided.

The mutation process in the new Multi-Objective Evolutionary Algorithm (MOEA) includes only three improved operators, namely Reallocate(), Exchange(), and Reposition(), which embrace the five operations previously used in BiEA. These operators make use of three basic functions: the stochastic SelectRoute() and SelectCustomer(), and the deterministic InsertCustomer().

The new MOEA was tested on benchmark sets for two VRP variants: the VRP with Time Windows and the Capacitated VRP. The former was solved, first, by optimising the two objectives number of routes (R) and travel distance (D), and then, an additional objective was considered, that is the delivery time (T). The results obtained by MOEA for the first case were analysed in several ways. Firstly, the outcome Pareto approximations showed to be actual bi-objective for 31 out of the 56 benchmark instances. In this respect, MOEA was compared with previous multi-objective studies which found conflicting objectives for fewer number of instances, and many of their overall Pareto approximations were dominated by those from MOEA. Secondly, the results were compared with those from previously published single-objective and bi-objective algorithms, showing that MOEA found solutions which travel distance was no more than 2% higher than the best-known, but with smaller or equal number of routes. Next, a comparison with the previously proposed BiEA was done, showing that, in fact, the modifications made to the mutation stage contributed to its performance enhancement. Finally, and perhaps most importantly, the new MOEA was evaluated using the coverage, convergence and hypervolume multi-objective performance metrics, showing significantly better results than the well-known multi-objective evolutionary optimiser NSGA-II for many test instances.

In respect of the tri-objective optimisation, the algorithm was tested with four different objective settings: first minimising pairs of objectives (RD, RT and DT), and then all three at once (RDT). The coverage, convergence and hypervolume performance metrics were used to evaluate the algorithm, showing that settings DT and RDT had a higher coverage of RD and RT, and that their solutions were closer to a composite reference set and delimited a larger hypervolume, indicating that the minimisation of the delivery time improves the algorithm's performance. Moreover, RDT significantly covered DT in more instances, which implied that even better results can be obtained by considering the minimisation of all three objectives at the same time. The non-dominated sets found by MOEA with objective setting RDT were compared with previous studies, and, although its results are not the overall best, they are better in some respects than those from previous multi-objective approaches. MOEA was compared against NSGA-II, by measuring the tri-objective coverage, convergence and hypervolume, showing that solutions found by MOEA are significantly better for many of the test instances.

With regard to the solution of the CVRP, MOEA was set to optimise the number of routes and travel distance only, since distance and time are not in conflict in this case. The results obtained showed that the outcome set had a proper bi-objective Pareto approximation for three test instances out of 26. On the other hand, although MOEA did not manage to achieve the best-known solution for all the instances in two benchmark sets, the difference between the best-known results and the best solutions found by MOEA-RD is very narrow, no more than 0.5%.

Chapter 6

Conclusions

The origin of the research presented in this thesis was, as in many other cases, curiosity. Curiosity to know, in the first instance, why some nature-inspired techniques perform suitably for certain kind of problems and not for others. In fact, the main topic of this thesis was selected while experimenting with different methods for solving different sorts of problems.

As a consequence of this experimentation, the answer to the very first research questions *why those studies were combining an EA with another technique* for solving VRPs, and *if the reason would be that EAs can not solve the problem by themselves* emerged: for certain classes of problems, the use of a *simple* Evolutionary Algorithm does not guarantee population diversity will be preserved, consequently it gets stuck in sub-optimum solutions and additional strategies have to be considered in order to escape from these regions and have a proper exploration of the search space.

At this stage, there were two possible directions for research: The first involved the consideration of another heuristic method as part of the evolutionary process, a topic that had already been the subject of extensive investigation. The second was to propose a method for measuring solution similarity and use this information to stimulate and preserve population diversity. The latter was chosen as the most promising way forward.

This chapter presents an evaluation of the proposed approach at three phases of enhancement, and the main contributions and achievements of this investigation. Finally, a number of topics for further research are considered.

6.1 Evaluation of the proposed approach

This section evaluates the proposed algorithm at its different stages of development, namely the preliminary multi-objective density-restricted Genetic Algorithm (drGA) and the Bi-objective Evolutionary Algorithm (BiEA), presented in Chapter 4, and the final Multi-Objective Evolutionary Algorithm (MOEA), described in Chapter 5.

6.1.1 Multi-objective density-restricted Genetic Algorithm

With the aim of overcoming the lack of population diversity, identified in the exploratory study, by means of using intrinsic information, the first approach to preserve diversity was a control to restrict the density of equal solutions in the population, which was incorporated as part of the proposed multi-objective density-restricted Genetic Algorithm (drGA) introduced in Section 4.2. This algorithm considered a recombination operator which intended preserving routes from both parents and a mutation process with six operators, which were applied proportionally to a weight they were pre-assigned: merge, insertion, and swap making inter-route changes, and inversion, shift, and split performing intra-route modifications. Then, the density control was used: if after the crossover and mutation stages the density of any solution had grown to more than the maximum allowed, further individuals were removed from the population until the density was rectified. Finally, fittest solutions were considered for the next generation, and in case the population size was compromised, the solutions with shortest travel distance were preferred.

This algorithm was tested on standard benchmark instances and, despite achieving better solutions than the exploratory approach, the conclusion was that the fact of having all distinct solutions in the population was not enough to escape from sub-optimum solutions, and a more sophisticated diversity preservation technique was necessary. Specifically, solutions from drGA were compared with those from past studies in a standard single-objective style, where the solutions with the shortest travel distance for each instance were considered. The results showed that solutions from drGA were between 5% and 12% larger, on average, than the best achieved by previous approaches. An interesting result from drGA was that, for some instance categories, it obtained solutions with shorter travel distances using more vehicles, when compared with those from previous studies, and vice versa for the accumulated results, which suggested the multi-objective nature of the VRPTW.

6.1.2 Bi-objective Evolutionary Algorithm

In order to obtain more reliable information about population diversity, the Jaccard similarity, based on the Jaccard's similarity coefficient, was designed and used to quantify the similarity between two solutions to the VRP, and this, in turn, to calculate the similarity between one solution and the rest of the population, and the population diversity. It is important to remark that this similarity measure is independent of the solution encoding and can be used in any variant of the VRP, since it uses the inherent information about the arcs forming the routes in the solution.

A similarity measure replaced the density control in drGA, which became the Bi-objective Evolutionary Algorithm (BiEA) described in Section 4.3, and was used to select one of the parents for the recombination process and to prevent individuals that were highly similar to others in the population being considered for the next generation. BiEA was set to use either Jaccard similarity or edit distance and tested on benchmark instances. The results of the experiments were submitted to the coverage, convergence and hypervolume multi-objective performance metrics, which showed that both obtained similar improved performance, however, the execution time of BiEA set to use edit distance was considerably longer, more than 300% on average, than when it was set to use Jaccard similarity.

Solutions from BiEA were compared, using the multi-objective performance metrics, to those from a version of the algorithm which did not consider the similarity measure for parent selection. The results of the performance metrics indicated that BiEA outstandingly and significantly surpassed the other algorithm in all instance categories. Furthermore, the population diversity preserved by both algorithms was analysed, confirming that the use of the similarity measure for selecting one of the parents did contribute to preserve a higher population diversity and to find higher quality solutions and larger non-dominated sets.

Results from BiEA showed a significant improvement over drGA, though a suitable comparison with previous studies using multi-objective performance metrics could not be made due to the fact that, unfortunately, they did not present their results in a proper multi-objective style. However, a traditional single-objective comparison was performed, where the best results from BiEA, i.e. solutions with the smallest number of routes and the shortest travel distance, were shown to be already better or similar to those from previous multi-objective studies, since they achieved the smallest average number of routes in all categories and, with respect to the average travel distance, they were the shortest in four of the six categories. Solutions from BiEA were also compared, in the same manner, with those from the versions of BiEA which only minimised one of the number of routes and travel distance objectives. This comparison remarkably reflected the importance of considering the simultaneous minimisation of both objectives, since BiEA obtained, for all instance categories, solutions with the smallest numbers of routes and the shortest travel distances.

In addition to the changes regarding the inclusion of the similarity measure, the new BiEA differed from drGA in the mutation process, which then only had five operators: the split operator was applied inversely proportional to the number of routes in the solution, then, one of the insertion and swap was applied, and finally, the inversion or shift was performed. In order to know what the contribution of the mutation operators was, an additional series of experiments was performed, setting different combinations of the mutation operators. After analysing the results with the coverage and convergence performance metrics, it was noticeable that the split operator was contributing more substantially to BiEA's performance, since, when this operator was excluded from the mutation process, BiEA obtained the worst results, and when BiEA was set to perform only this operator, the best results were obtained. The characteristic of this operator is that, after splitting a route, it attempts to reallocate the customers belonging to the shortest sub-route to other existing routes. This result suggested that the mutation process and operators could be enhanced.

6.1.3 Multi-Objective Evolutionary Algorithm

According to the analysis of the mutation operators, the mutation process was revised: the five mutation operations are now embraced in only three, namely reallocation, exchange and reposition of customers, and the process makes use of three basic functions, two of which are used to stochastically select routes and customers, while the other deterministically inserts a customer in a route. The final Multi-Objective Evolutionary Algorithm (MOEA), presented in Section 5.1, was analysed from several perspectives in the three different scenarios described below.

6.1.3.1 Bi-objective optimisation of the VRPTW

The first scenario involved the solution of the VRP with Time Windows regarding the minimisation of the number of routes and the travel distance. The analysis of the results confirmed that the overall Pareto approximation to 31 out of 56 benchmark instances had conflicting objectives, and actually represent the extent to which MOEA is finding the best-known Pareto approximation. Additionally, MOEA found multiple solution Pareto approximations for more instances than those found by recent multi-objective approaches.

When compared with previous studies, considering the solutions with the smallest number of routes and the shortest travel distance, although results from MOEA are not the overall best, they present an improvement when compared with previous multi-objective studies, since they have the smallest average number of routes for the six instance categories and the shortest average travel distance in four. In general, we conclude that MOEA presents a performance comparable or better than the previous multi-objective studies. Additionally, MOEA found the best-known solutions for 19 out of 56 instances, and for 22 instances MOEA achieved solutions with an increase of 2% in the travel distance and with equal or less number of routes compared with the best-known.

Results were also compared, by means of the coverage, convergence and hypervolume multi-objective performance metrics, with those from BiEA, confirming that the modifications made to the mutation stage contributed to find better solutions and, consequently, to the improvement of MOEA's performance, since solutions from MOEA significantly covered those from BiEA in the majority of the benchmark instances, for which they were also significantly closer to the reference set and delimited a significantly larger objective space.

Solutions were also compared in the same manner with those from NSGA-II, where it was noticeable that MOEA surpassed NSGA-II in many of the benchmark instances, since solutions from the former significantly covered those from the latter for the majority of the instances in five of the six categories, for which they were significantly closer to the reference set and defined a significantly larger objective space. According to the coverage metric, NSGA-II performed significantly better than MOEA in only two of the 56 instances. Moreover, the population diversity preserved by MOEA was higher that the preserved by NSGA-II for four out of the six instance categories, and the Pareto approximations were significantly larger in two of the categories.

Finally, MOEA was set to use edit distance instead of Jaccard similarity and the results were also analysed by means of the performance metrics, resulting in that this algorithm performed equally well as with Jaccard similarity, in the sense that it significantly surpassed NSGA-II in many of the benchmark instances and that both edit distance and Jaccard similarity significantly outperformed each other in approximately the same number of test instances.

6.1.3.2 Tri-objective optimisation of the VRPTW

The second scenario that was studied with MOEA took into account the optimisation of three objectives while solving the VRP with Time Windows: the standard number of routes (R) and travel distance (D), plus the delivery time (T). In this case, MOEA was tested with four different objective settings, first minimising pairs of objectives, i.e. RD, RT and DT, and then all three at once, that is RDT. The non-dominated solutions from each algorithm were submitted to the coverage, convergence and hypervolume performance metrics. Settings DT and RDT achieved significantly better solutions than settings RD and RT, in the sense that solutions from the former had a wider coverage of those from the latter, were closer to composite reference sets and delimited a larger objective space. RDT performed significantly better than DT in more instances, which implies that even better solutions can be obtained by considering the minimisation of all three objectives at the same time.

Solutions from MOEA were also compared to those from previous studies in the traditional single-objective style. This comparison indicated that MOEA could still find comparable solutions with respect to previous multi-objective studies, since, as in the bi-objective case, they achieved the smallest average number of routes in all six instance categories and the shortest average travel distance in three of them. These results showed that, overall, MOEA maintained its level of performance.

MOEA was compared against NSGA-II, by computing the tri-objective performance metrics, showing that, for the majority of the instances in the same five categories as in the bi-objective case, the non-dominated sets found by MOEA still significantly covered those from NSGA-II, were significantly closer to the reference set and defined a significantly larger hypervolume. This time, NSGA-II performed significantly better than MOEA in six of the 56 benchmark instances according to the coverage metric, and in one according to the hypervolume. Furthermore, the population diversity preserved by MOEA was still higher that the preserved by NSGA-II for the same four categories as in the bi-objective case, and the Pareto approximations were significantly larger in the two categories that were also larger in the bi-objective scenario.

Lastly, MOEA was set to use edit distance instead of Jaccard similarity. This algorithm significantly outperformed NSGA-II in three of the six categories, however, NSGA-II was significantly better in the other three. Moreover, solutions from this algorithm were significantly better than those from MOEA for instances in only one of the set categories, and those from MOEA were significantly better in three.

6.1.3.3 Bi-objective optimisation of the CVRP

The third and last scenario considered the solution of the Capacitated VRP concerning the minimisation of the number of routes and travel distance only, since distance and time are not in conflict in this case. MOEA was tested on 26 benchmark instances from two different sets, and, although MOEA managed to find the best-known solutions for only six of the instances, the overall average difference was of 3.1%. However, it is mportant to mention that, if only the instances size $N \leq 120$ are considered, the overall average difference decreases to 0.46%. Additionally, the solutions obtained by MOEA showed that the outcome set had a proper bi-objective Pareto approximation for only four of the instances, and those for all other contained one single solution.

6.1.4 Final discussion

In Section 5.2.3 we observed that MOEA obtained substantial savings in the number of routes for Solomon's instance categories R2 and RC2 with respect to the best-known results, however, the travel distance had been increased. Additionally, in Sections 5.2.4 and 5.3.3, it was clear that, although MOEA preserved a higher population diversity, and explored and exploited a wider search space than NSGA-II, solutions obtained by MOEA for instances in categories R2 and RC2 were not as good as those found for instances in the remaining categories. This situation might be due, to a certain degree, to an insufficient exploration and exploitation of regions in the search space where solutions with larger number of routes are located. Let us remind that instances in categories R2 and RC2 have an even larger search space than that of instance in categories C1, C2, R1, and RC1.

The facts above highlight the need for an adequate balance between exploration and exploitation of the search space, specially when its characteristics are similar to those defined by the R2 and RC2 instances. One of the objectives of this research was to use information regarding solution similarity in order to promote and preserve population diversity, and consequently provide the algorithm with more possibilities for exploration and exploitation of the search space. These objectives, as appropriately analysed in Chapter 5, were accomplished, however they were not enough to substantially succeed for instances in sets R2 and RC2.

This situation might be rectified by an adaptive mechanism for population diversity control, in order to improve the balance between exploration and exploitation of the search space. That is, at some point the algorithm could stop trying to increase population diversity and concentrate on the exploitation of promising regions.

On the other hand, MOEA and its predecessors were originally designed to solve the two variants studied in this thesis. However, the algorithm at its actual state of development could equally well solve other VRPs, particularly those variants that consider capacity and customer service time constraints. Actually, CVRP instances were slightly modified by incorporating *silly* infinite time windows. This procedure could be used, to some extent, in other variants of the VRP that could be handled by suitably modifying the instance customer precedence restrictions and by appropriately setting in the algorithm the operational constraints.

In general, VPRs can be seen as problems where a set of items have to be split into groups regarding some criteria and, for each group, items have to be sequenced according to some priorities, if any. This level of abstraction could lead to models, designs and implementations of a broad variety of combinatorial problems, which could potentially consider multiple objectives to be optimised.

6.2 Main contributions and achievements

The research carried out in this thesis made several contributions, which respond to the research questions initially raised and achieve the research objectives set. These contributions can be categorised according to the topics below.

6.2.1 Population diversity preservation

For the particular VRPs studied in this thesis, if one intends to solve them by means of an EA, special attention has to be given to preserve population diversity, since one needs not only to consider the diversity in the objective space, but also the intrinsic information of the solutions, e.g. customer assignment to routes and customer service sequence. In Section 4.2 it was proven than having all distinct solutions in the population, and all of them with different objective functions values, was not enough to escape from sub-optimum regions of the search space. Thus, a more sophisticated method than guaranteeing different solutions in the population is required to preserve diversity.

An appropriate measure, based on the Jaccard's similarity coefficient, was developed to quantify how similar two solutions to the VRP are, and this, in turn, is used to determine how similar one solution is to the rest of the population and the population diversity. It is important to say that this measure does not depend on how the solutions are represented, since it uses the inherent information about the arcs forming the designed routes, that it can be applied to any variant of the VRP, and that it has a linear time complexity.

In Section 4.3.2.2 it was confirmed that, when Jaccard similarity is incorporated in an Evolutionary Algorithm for solving the VRPTW, better and larger Pareto approximations are found and a higher population diversity is preserved than when similarity information is not considered. Furthermore, in Sections 5.2.4 and 5.3.3, this measure was proven to perform equally well or better than the edit distance when they are incorporated into the designed MOEA for solving the VRPTW regarding multiple objectives. It is worth noting that MOEA executes much quicker when Jaccard similarity is used than when it uses edit distance, since the latter has a quadratic time complexity.

The contributions stated above, respond to the research questions raised in the introductory chapter and achieve the research objective regarding population diversity preservation.

6.2.2 Solution of VRPs by means of an Evolutionary Algorithm

In order to design a multi-objective Evolutionary Algorithm for solving two variants of the VRP, a mutation process was developed to help better exploit the search space. This mutation process includes a set of three operators, which make modifications in the assignment of customers to routes and in the sequence of service within a route, and three basic functions, two of which are stochastic, to select routes and customers, and the other is deterministic, to insert a customer into a specific route.

Moreover, a Multi-Objective Evolutionary Algorithm (MOEA) has been formulated that effectively solves the Capacitated VRP and the VRP with Time Windows, regarding the optimisation of at least two objectives. This algorithm includes the above-mentioned Jaccard similarity and mutation process. In Chapter 5 it was confirmed that the good performance of this algorithm is, in part, due to the consideration of the information provided by the Jaccard similarity, since it helps the recombination phase to explore more suitably the search space, and, additionally, the mutation stage proved to be an effective technique for exploiting it. This combination leads to provide solutions that better represent the trade-offs between the objectives.

The solutions to the CVRP and VRPTW from the proposed algorithm were compared with those from the most successful of the previous studies from a single-objective point of view, showing that, although they are not the overall best, they are better than some of the previously published. However, this comparison is often misleading, since the best result for one objective does not necessarily represent the multi-objective performance of an optimiser.

The contributions exposed above accomplish the research objective related to the effective solution of the CVRP and VRPTW regarding multiple objectives.

6.2.3 Multi-objective performance analysis

The study presented in this thesis is one of the very few that have made a proper performance analysis of the optimisation of VRPs regarding multiple objectives, in the sense that appropriate multi-objective performance metrics were applied to the outcome non-dominated solutions from each algorithm that was considered.

In this respect, in Section 5.3.1, the developed MOEA was confirmed to find even better non-dominated solutions when it is set to optimise the delivery time, additionally to the number of routes and travel distance, since it obtained significantly better results from the performance metrics than when any pair of the three objectives was set.

In Sections 5.2.4 and 5.3.3, MOEA proved to be a significantly better method for solving many instances of the VRPTW when it is compared with the popular and successful NSGA-II, regarding both the bi-objective and tri-objective optimisations. This improved performance is the result of considering the information provided by the Jaccard similarity, in the sense that MOEA preserves a higher population diversity, which suggests a wider exploration of the search space, leading the algorithm to find better solutions.

Additionally, although MOEA found that many of the VRPTW benchmark instances and few of the CVRP have conflicting objectives, solutions lie in a narrow region of the number of routes dimension, which make Pareto approximations comprise a relatively small number of solutions. This fact highlights the need for proper multi-objective VRPs benchmark instances.

These contributions complete the research objective corresponding to the suitable multi-criterion analysis, by means of utilising proper multi-objective performance metrics, of the optimisation of the studied VRPs.

6.3 Directions for future research

There remains considerable scope for further development of the proposed MOEA approach and its components, which can be categorised as MOEA's enhancement and further study of VRPs.

6.3.1 Enhancement of the Multi-Objective Evolutionary Algorithm

Although the proposed Jaccard similarity used by MOEA has been shown to result in improved performance, it still only explicitly considers the arcs used in the routes in a solution and not the sequence of them nor the number of routes. Consequently, additional improvements might be possible by developing more refined similarity measures, which provide performance improvements that justify the inevitable increased computational costs.

In the same context, a more strict analysis of the edit distance may be carried out, in order to determine exactly what aspects of the tri-objective optimisation are weakening its performance and consider this for the further improvements of the Jaccard similarity.

An adequate study, including supplementary experimentation and analysis, regarding the parameters involved in the proposed MOEA is required in order to further support the conclusions stated here.

MOEA's performance for instances in categories R2 and RC2 was not as good as for those in C1, C2, R1 and RC1. Thus, an in-depth analysis of this behaviour is required in order to identify the actual reasons for this situation.

6.3.2 Further study of Vehicle Routing Problems

More rigorous comparisons of the results from MOEA with other evolutionary multi-criterion optimisation methods can be addressed, which may involve the utilisation of further Pareto compliant multi-objective quality indicators.

There is also the possibility to explore the extension of MOEA to the optimisation of supplementary objectives while solving CVRP and VRPTW, such as workload imbalance, makespan and waiting time, in order to verify if the algorithm maintains its good performance.

One more possible direction of research is the adaptation of MOEA for tackling further variants of the VRP, which may consider additional constraints and objectives to be optimised, with the aim of solving routing problems even closer to real-life applications.

It is evident the lack of multi-objective Vehicle Routing Problem benchmark instances, thus an effort might be done to design improved instances considering the multiple objectives that could be optimised.

Appendix A

Sample of Pareto approximations, performance metrics and statistics

Table A.1 presents the Pareto approximations found by MOEA-RDT to Solomon's instance R105. The first column shows the run number and the following columns present the number of routes (R), travel distance (D) and delivery time (T) associated with each solution in the approximation set.

run	R	D	T	R	D	T	R	D	T	R	D	T
1	14	1377.63	2627.67	14	1379.89	2627.31	14	1392.59	2626.43	14	1394.71	2622.15
	14	1396.96	2621.80	14	1397.44	2616.42	14	1399.56	2612.14	14	1401.82	2611.79
	14	1408.32	2611.15	14	1410.58	2610.80	14	1414.13	2610.49	14	1416.38	2610.13
	14	1420.31	2607.38	14	1422.57	2607.03	14	1427.73	2603.80	14	1429.99	2603.44
	14	1442.30	2602.14	14	1444.55	2601.79	15	1374.42	2755.31	15	1375.90	2739.59
	15	1377.53	2711.31									
2	14	1377.11	2631.56	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.91	2624.35
	14	1384.17	2624.00	14	1385.45	2622.33	14	1386.22	2620.76	14	1388.48	2620.40
	14	1396.36	2619.41	14	1397.13	2617.83	14	1399.39	2617.48	15	1369.51	2743.48
	15	1370.03	2739.59	15	1372.28	2739.23	15	1373.76	2727.81	15	1374.28	2723.91
	15	1376.54	2723.56									
3	14	1419.75	2688.88	14	1420.27	2679.34	14	1421.94	2677.76	14	1422.64	2672.15
	14	1424.90	2671.80	14	1427.51	2671.79	14	1427.63	2651.26	14	1428.15	2641.73
	14	1430.17	2633.15	14	1432.32	2626.56	14	1434.58	2626.21	14	1461.47	2619.77

continued on next page...

...continued from previous page

run	R	D	T	R	D	T	R	D	T	R	D	T
	14	1463.73	2619.42	15	1395.35	2711.39	15	1397.60	2711.04	15	1399.87	2707.08
	15	1402.13	2706.72	15	1403.44	2702.28	15	1404.63	2702.19	15	1405.70	2701.92
	15	1406.26	2692.10	15	1408.52	2691.74	15	1410.79	2687.78	15	1413.05	2687.43
4	14	1408.93	2641.80	14	1409.19	2630.82	14	1411.45	2630.47	14	1412.35	2630.23
	14	1414.60	2629.87	14	1420.44	2627.94	14	1422.70	2627.59	14	1423.59	2627.34
	14	1425.85	2626.99	14	1428.79	2626.47	14	1429.49	2626.40	14	1431.04	2626.11
	14	1431.75	2626.05	14	1435.02	2623.01	14	1435.19	2622.12	14	1436.13	2616.34
	14	1438.39	2615.98	14	1440.31	2615.29	14	1442.56	2614.93	14	1450.70	2612.13
	14	1452.95	2611.78	14	1460.20	2611.57	14	1462.46	2611.21	14	1468.54	2611.10
	15	1392.42	2727.94	15	1394.68	2727.59	15	1396.68	2712.27	15	1398.93	2711.91
	15	1401.56	2711.15	15	1403.82	2710.80	15	1406.50	2710.56	15	1407.40	2710.20
	15	1407.92	2709.38									
5	14	1388.94	2631.46	14	1391.20	2631.10	14	1394.32	2623.93	14	1396.58	2623.58
	14	1397.27	2616.36	14	1399.52	2616.00	14	1402.62	2613.08	14	1404.87	2612.72
	14	1435.46	2612.62	14	1438.33	2612.26	14	1452.26	2611.92	15	1375.50	2767.64
	15	1378.37	2767.28	15	1379.53	2742.99	15	1381.79	2742.64	15	1385.47	2737.19
	15	1387.73	2736.84									
6	14	1381.06	2628.08	14	1383.31	2627.72	14	1384.84	2622.26	14	1386.22	2620.76
	14	1388.48	2620.40	14	1405.70	2618.85	14	1407.96	2618.50	14	1409.11	2617.08
	14	1411.37	2616.73	14	1411.55	2612.99	14	1413.81	2612.64			
7	14	1386.01	2647.21	14	1386.66	2646.79	14	1388.92	2646.44	14	1390.94	2643.48
	14	1393.20	2643.13	14	1394.02	2641.52	14	1394.68	2641.11	14	1396.94	2640.76
	14	1397.03	2638.84	14	1398.96	2637.80	14	1399.40	2632.79	14	1401.66	2632.44
	14	1407.42	2627.11	14	1409.67	2626.76	14	1416.09	2623.47	14	1418.35	2623.11
	14	1424.11	2617.78	14	1426.37	2617.43						
8	14	1378.89	2627.67	14	1381.15	2627.31	14	1383.17	2624.35	14	1385.43	2624.00
	14	1389.19	2621.66	14	1391.45	2621.30	14	1392.86	2618.03	14	1394.98	2617.30
	14	1397.23	2616.94	14	1398.83	2616.25	14	1400.94	2615.52	14	1403.20	2615.16
	14	1408.13	2614.41	14	1410.24	2613.68	14	1412.50	2613.32	14	1414.09	2612.63
	14	1416.21	2611.90	14	1418.20	2611.66	14	1418.47	2611.54	14	1420.45	2611.31
	14	1421.21	2610.29	14	1423.47	2609.94	14	1426.77	2609.63	14	1429.03	2609.28
	14	1429.79	2608.26	14	1432.05	2607.91	14	1441.15	2606.08	14	1443.41	2605.73
	14	1444.17	2604.71	14	1446.43	2604.36						
9	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.91	2624.35	14	1383.94	2620.76
	14	1386.20	2620.40	14	1388.47	2616.43	14	1390.73	2616.07	14	1393.97	2613.66
	14	1396.23	2613.31	14	1418.20	2611.66	14	1418.55	2610.29	14	1420.81	2609.94
	14	1428.09	2608.72	14	1430.35	2608.36	14	1442.72	2607.64	14	1443.07	2606.28
	14	1445.33	2605.92									
10	14	1378.78	2628.08	14	1383.17	2622.33	14	1383.94	2620.76	14	1387.81	2618.18
	14	1393.78	2616.40	14	1411.37	2614.95	14	1417.34	2613.17	15	1368.28	2736.27
	15	1372.62	2730.31	15	1373.44	2728.95						

continued on next page...

...continued from previous page

run	R	D	T	R	D	T	R	D	T	R	D	T
11	14	1432.39	2646.42	14	1454.37	2642.67	14	1456.63	2642.32	14	1457.98	2639.45
	14	1458.40	2637.58	14	1462.88	2635.29	14	1465.12	2634.19	15	1398.16	2764.17
	15	1398.87	2745.39	15	1399.48	2745.34	15	1400.65	2723.81	15	1404.78	2708.14
	15	1409.72	2706.30	15	1412.33	2693.52	15	1414.59	2693.17	15	1415.48	2690.73
	15	1415.94	2690.30	15	1423.60	2688.84	15	1426.12	2683.76	15	1429.27	2680.97
	15	1429.73	2680.54	15	1431.99	2680.18						
12	14	1386.01	2647.21	14	1388.27	2646.85	14	1394.02	2641.52	14	1394.87	2640.37
	14	1397.13	2640.02	14	1399.69	2637.26	14	1401.04	2635.36	14	1402.89	2634.69
	14	1403.47	2620.92	14	1405.73	2620.57	14	1413.42	2620.46	14	1414.72	2615.96
	14	1416.98	2615.61	14	1419.24	2614.03	14	1421.50	2613.68	14	1425.16	2613.52
	14	1425.20	2612.25	14	1427.46	2611.90	14	1431.12	2611.74	14	1433.38	2611.38
	15	1369.69	2745.97	15	1378.76	2735.80	15	1381.01	2735.44	15	1382.03	2734.26
	15	1383.01	2720.12	15	1385.27	2719.77						
13	14	1400.12	2648.96	14	1401.51	2642.14	14	1403.63	2637.87	14	1405.89	2637.52
	14	1408.38	2636.09	14	1410.30	2633.84	14	1411.05	2633.67	14	1412.56	2633.49
	14	1432.19	2633.17	14	1432.22	2630.53	14	1434.47	2630.18	14	1434.77	2629.43
	14	1437.03	2629.08	15	1366.38	2755.49	15	1366.41	2740.59	15	1368.66	2740.24
	15	1370.64	2739.81	15	1370.66	2724.92	15	1372.92	2724.57	15	1375.80	2723.53
	15	1375.82	2717.60	15	1378.08	2717.25	15	1386.49	2715.96	15	1386.51	2710.02
	15	1388.77	2709.67	15	1398.07	2705.45						
14	14	1377.33	2641.23	14	1377.63	2627.67	14	1379.89	2627.31	14	1383.17	2624.35
	14	1384.84	2622.26	14	1386.22	2620.76	14	1388.48	2620.40	14	1397.44	2616.42
	14	1399.70	2616.07	14	1416.11	2613.99	14	1418.20	2611.66	14	1418.55	2610.29
	14	1420.81	2609.94	14	1427.88	2608.34	14	1428.23	2606.98	14	1430.49	2606.62
15	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.55	2623.38	14	1383.81	2623.03
	14	1386.71	2620.44	14	1388.97	2620.08	14	1392.05	2619.54	14	1393.37	2616.96
	14	1395.63	2616.61	14	1399.35	2616.06	14	1400.57	2612.45	14	1402.83	2612.10
	14	1405.91	2611.55	14	1408.17	2611.20	14	1412.95	2608.86	14	1415.21	2608.50
	14	1426.82	2603.86	14	1429.07	2603.51						
16	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.91	2624.35	14	1384.17	2624.00
	14	1392.86	2618.03	14	1394.98	2617.30	14	1397.23	2616.94	14	1397.44	2616.42
	14	1398.83	2616.25	14	1400.94	2615.52	14	1403.20	2615.16	14	1418.20	2611.66
	14	1420.31	2607.38	14	1422.57	2607.03	14	1440.31	2605.28	14	1442.42	2604.54
	14	1443.27	2601.80	14	1446.14	2601.45	14	1463.50	2600.66	14	1465.76	2600.31
	14	1469.61	2600.20	15	1377.53	2711.31						
17	14	1401.43	2626.29	14	1403.69	2625.94	14	1409.82	2616.86	14	1412.08	2616.51
	14	1423.26	2614.15	14	1425.52	2613.79	14	1430.41	2613.26	14	1430.58	2612.10
	14	1430.93	2610.73	14	1433.19	2610.38	14	1436.84	2609.94	14	1437.00	2608.79
	14	1437.36	2607.42	14	1439.62	2607.07	15	1394.23	2748.38	15	1394.27	2720.62
	15	1394.30	2714.03	15	1396.56	2713.68						
18	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.91	2624.35	14	1384.17	2624.00

continued on next page...

...continued from previous page												
run	R	D	T	R	D	T	R	D	T	R	D	T
	14	1405.80	2619.32	14	1408.06	2618.97	14	1410.08	2616.00	14	1412.33	2615.65
	14	1412.95	2608.86	14	1415.21	2608.50	14	1423.50	2608.37	14	1425.72	2607.38
	14	1426.07	2606.01	14	1428.33	2605.66	14	1441.12	2600.51	14	1443.38	2600.16
	14	1447.41	2599.35	14	1449.67	2598.99						
19	14	1391.29	2619.97	14	1393.13	2619.70	14	1395.39	2619.34	14	1398.14	2617.12
	14	1399.97	2614.76	14	1402.23	2614.41	14	1414.09	2606.69	14	1416.35	2606.33
	14	1436.07	2601.21	14	1438.33	2600.86	14	1446.18	2600.71	15	1376.58	2723.76
	15	1378.83	2723.40	15	1380.83	2708.08	15	1383.09	2707.73	15	1391.21	2680.00
20	14	1391.53	2638.15	14	1392.48	2636.78	14	1393.52	2634.97	14	1394.85	2633.71
	14	1396.97	2629.43	14	1399.23	2629.07	14	1400.02	2629.04	14	1401.65	2618.93
	14	1403.77	2618.20	14	1406.02	2617.84	14	1412.56	2616.01	14	1414.82	2615.66
	14	1429.36	2611.02	14	1429.82	2610.59	14	1431.47	2610.28	14	1431.93	2609.85
	14	1434.19	2609.50	15	1368.29	2747.00	15	1368.63	2730.72	15	1369.58	2729.35
	15	1371.84	2728.99	15	1372.89	2715.04	15	1373.84	2713.67	15	1376.09	2713.32
	15	1386.49	2698.39	15	1386.83	2682.11	15	1387.78	2680.74	15	1390.04	2680.39
21	14	1377.63	2627.67	14	1379.89	2627.31	14	1392.90	2624.05	14	1395.15	2623.69
	14	1396.03	2623.35	14	1397.44	2616.42	14	1399.56	2612.14	14	1401.82	2611.79
	14	1414.13	2610.49	14	1414.83	2608.04	14	1417.09	2607.68	14	1420.31	2607.38
	14	1422.57	2607.03	14	1429.39	2603.39	14	1431.65	2603.04	15	1371.17	2750.82
	15	1371.19	2744.88	15	1373.45	2744.53	15	1375.42	2735.14	15	1375.44	2729.21
22	14	1398.17	2629.89	14	1400.43	2629.54	14	1401.35	2618.60	14	1403.61	2618.25
	14	1404.88	2610.90	14	1407.14	2610.55	14	1407.20	2609.98	14	1409.45	2609.62
	14	1442.79	2606.52	14	1444.91	2605.78	14	1444.96	2604.97	14	1447.07	2604.23
	14	1448.06	2602.24	14	1450.23	2600.69	14	1452.48	2600.33			
23	14	1395.63	2646.21	14	1395.93	2633.13	14	1398.19	2632.78	14	1404.40	2623.70
	14	1406.66	2623.35	14	1417.70	2623.10	14	1418.77	2622.54	14	1418.83	2619.98
	14	1420.13	2618.61	14	1422.39	2618.26	14	1425.15	2617.13	14	1426.17	2613.67
	14	1427.30	2610.55	14	1428.60	2609.18	14	1430.86	2608.83	14	1442.97	2608.02
	14	1445.22	2607.67	15	1392.35	2715.91	15	1393.13	2709.20	15	1395.39	2708.85
24	14	1402.71	2649.17	14	1403.72	2634.35	14	1405.98	2634.00	14	1411.93	2632.60
	14	1411.94	2632.22	14	1414.64	2631.43	14	1416.89	2631.08	14	1420.60	2629.65
	14	1422.86	2629.30	14	1425.84	2618.51	14	1428.09	2618.16	14	1431.79	2617.11
	14	1431.80	2616.73	14	1434.06	2616.38	14	1434.94	2612.70	14	1437.20	2612.35
	15	1378.16	2740.13	15	1379.12	2739.78	15	1379.35	2738.76	15	1381.12	2724.46
	15	1383.38	2724.11	15	1383.61	2723.09	15	1385.86	2722.74	15	1389.71	2716.63
	15	1390.25	2716.31	15	1391.97	2716.28	15	1392.51	2715.95	15	1402.16	2715.45
	15	1402.17	2712.09									
25	14	1404.73	2630.27	14	1405.78	2616.91	14	1408.04	2616.56	14	1417.99	2612.26
	14	1418.09	2610.56	14	1420.35	2610.21	15	1380.32	2714.27	15	1382.57	2713.91
	15	1384.60	2710.95	15	1394.44	2706.20	15	1396.70	2705.84	15	1398.72	2702.88
	15	1400.98	2702.53	15	1402.31	2701.56						

continued on next page...

run	R	D	T	R	D	T	R	D	T	R	D	T
...continued from previous page												
26	14	1403.76	2653.70	14	1403.86	2627.06	14	1403.87	2622.78	14	1406.13	2622.43
	14	1409.42	2621.55	14	1411.68	2621.20	14	1423.48	2619.34	14	1424.62	2616.21
	14	1425.63	2612.76	14	1426.77	2609.63	14	1429.03	2609.28	14	1429.79	2608.26
	14	1432.05	2607.91	15	1380.84	2723.77	15	1382.08	2719.55	15	1384.34	2719.20
	15	1386.22	2716.25	15	1387.47	2712.03	15	1389.72	2711.67	15	1394.23	2710.57
	15	1395.48	2706.34	15	1397.74	2705.99	15	1400.18	2680.59	15	1400.18	2684.87
	15	1401.43	2676.37	15	1403.69	2676.01						
27	14	1403.13	2645.86	14	1405.39	2645.51	14	1407.19	2645.27	14	1407.43	2634.86
	14	1409.69	2634.51	14	1415.17	2631.89	14	1417.20	2624.57	14	1419.45	2624.22
	14	1419.80	2620.48	14	1422.05	2620.12	14	1426.10	2615.14	14	1428.36	2614.79
	14	1428.71	2611.05	14	1430.96	2610.69	15	1370.52	2741.17	15	1372.78	2740.81
	15	1374.78	2725.49	15	1377.04	2725.14	15	1379.94	2718.17	15	1382.20	2717.82
	15	1388.38	2710.45	15	1390.64	2710.10	15	1393.54	2703.13	15	1395.80	2702.78
	15	1396.85	2701.02	15	1399.11	2700.67	15	1400.82	2686.73	15	1401.06	2676.33
28	14	1377.63	2627.67	14	1379.89	2627.31	14	1381.91	2624.35	14	1384.17	2624.00
	14	1392.82	2621.43	14	1392.86	2618.03	14	1394.98	2617.30	14	1397.23	2616.94
	14	1398.83	2616.25	14	1400.94	2615.52	14	1403.20	2615.16	14	1403.78	2615.11
	14	1406.03	2614.76	14	1409.74	2613.33	14	1412.00	2612.98	14	1420.57	2610.12
	14	1421.03	2609.69	14	1422.68	2609.38	14	1423.15	2608.95	14	1425.40	2608.60
	14	1426.53	2608.34	14	1427.00	2607.91	14	1428.65	2607.60	14	1429.11	2607.17
	14	1431.37	2606.82									
29	14	1428.90	2629.26	14	1431.15	2628.91	14	1443.57	2626.60	14	1445.83	2626.24
	14	1448.58	2622.88	14	1450.84	2622.52	14	1456.02	2612.79	14	1458.28	2612.44
	15	1388.08	2750.66	15	1391.87	2742.60	15	1392.21	2734.98	15	1392.95	2732.99
	15	1396.00	2726.93	15	1397.39	2725.50	15	1398.24	2716.04	15	1399.83	2712.50
	15	1400.68	2703.05	15	1402.94	2702.69	15	1405.51	2700.05	15	1406.47	2698.95
	15	1412.55	2690.03	15	1414.80	2689.67						
30	14	1391.49	2623.16	14	1393.75	2622.81	14	1395.77	2619.85	14	1398.03	2619.49
	14	1422.60	2616.83	14	1423.99	2614.04	14	1424.00	2609.76	14	1426.26	2609.41
	14	1426.82	2603.86	14	1429.07	2603.51	14	1444.90	2602.75	14	1447.16	2602.39

Table A.1: Number of routes (R), travel distance (D) and delivery time (T) associated with the solutions in the Pareto approximations obtained by MOEA-*RDT* for instance R105.

Table A.2 presents, for Solomon's instance R105, the $\mathsf{M_D}$ values corresponding to the Pareto approximations result from each run of MOEA-*RDT*. These $\mathsf{M_D}$ values are normalised according to the reference point $\boldsymbol{z} = (100.00, 4989.42,\ 13017.71)$. The average of these 30 $\mathsf{M_D}$ values is 0.0019.

run	M_D	run	M_D	run	M_D	run	M_D	run	M_D	run	M_D
1	0.0009	6	0.0011	11	0.0069	16	0.0014	21	0.0010	26	0.0025
2	0.0007	7	0.0022	12	0.0018	17	0.0019	22	0.0015	27	0.0025
3	0.0052	8	0.0010	13	0.0021	18	0.0010	23	0.0019	28	0.0009
4	0.0033	9	0.0008	14	0.0009	19	0.0011	24	0.0024	29	0.0042
5	0.0018	10	0.0008	15	0.0007	20	0.0015	25	0.0024	30	0.0011

Table A.2: M_D values corresponding to the Pareto approximations, result from each run of MOEA-*RDT*, to Solomon's instance R105.

Similarly, Table A.3 presents, for Solomon's instance R105, the M_H values corresponding to the Pareto approximations result from each run of MOEA-*RDT*. These M_D values are normalised according to the reference point $\boldsymbol{z} = (100.00, 4989.42, 13017.71)$. The average of these 30 M_H values is 0.6096.

run	M_H	run	M_H	run	M_H	run	M_H	run	M_H	run	M_H
1	0.6072	6	0.6124	11	0.6072	16	0.6067	21	0.6068	26	0.6090
2	0.6074	7	0.6120	12	0.6079	17	0.6047	22	0.6119	27	0.6106
3	0.6065	8	0.6147	13	0.6097	18	0.6154	23	0.6052	28	0.6141
4	0.6057	9	0.6148	14	0.6141	19	0.6090	24	0.6077	29	0.6076
5	0.6063	10	0.6072	15	0.6142	20	0.6102	25	0.6078	30	0.6127

Table A.3: M_H values corresponding to the Pareto approximations, result from each run of MOEA-*RDT*, to Solomon's instance R105.

Table A.4 presents the Pareto approximations found by NSGA-II to Solomon's instance R105. The first column shows the run number and the following columns present the number of routes (R), travel distance (D) and delivery time (T) associated with each solution in the approximation set.

run	R	D	T	R	D	T	R	D	T	R	D	T
1	14	1479.23	2657.82	14	1479.30	2656.32	14	1481.47	2654.90	14	1481.54	2653.40
	14	1483.80	2653.05	15	1419.63	2725.36	15	1421.74	2721.08	15	1423.99	2718.16
	15	1426.24	2717.80	15	1429.76	2716.20	15	1432.02	2715.85	15	1434.98	2715.14
	15	1435.71	2714.37	15	1437.22	2712.22	15	1437.95	2711.45	15	1440.21	2711.09
	15	1443.00	2710.27	15	1443.72	2709.49	15	1445.98	2709.14	15	1447.05	2707.05
	15	1449.30	2704.13	15	1451.56	2703.78	15	1455.07	2702.18	15	1457.33	2701.83
	15	1461.02	2700.35	15	1463.26	2697.43	15	1465.52	2697.07	15	1469.04	2695.47

continued on next page...

...continued from previous page

run	R	D	T	R	D	T	R	D	T	R	D	T
	15	1471.29	2695.12	15	1477.20	2693.53						
2	14	1421.24	2655.62	14	1422.94	2655.26	14	1423.72	2646.03	14	1423.95	2636.66
	14	1425.65	2635.82	14	1427.90	2635.47	14	1434.86	2634.01	14	1436.56	2633.17
	14	1438.82	2632.81	15	1399.21	2763.26	15	1399.44	2753.89	15	1401.14	2753.04
	15	1403.40	2752.69	15	1406.72	2734.99	15	1406.94	2725.62	15	1408.64	2724.77
	15	1410.90	2724.42	15	1420.68	2721.34						
3	15	1387.98	2743.18	15	1390.08	2733.66	15	1392.24	2727.51	15	1394.49	2727.15
	15	1396.52	2724.19	15	1398.77	2723.84	15	1402.87	2712.36	15	1405.13	2712.01
	15	1407.15	2709.04	15	1409.40	2708.69	15	1413.46	2708.21	15	1415.72	2707.86
	15	1416.99	2703.28	15	1419.24	2702.93	15	1421.27	2699.97	15	1423.52	2699.61
	15	1437.74	2695.08	15	1439.99	2694.72	15	1449.96	2693.70	15	1452.22	2693.35
4	15	1412.56	2724.19	15	1415.00	2714.03	15	1418.03	2711.19	15	1420.99	2704.91
	15	1423.48	2704.89	15	1424.01	2702.08	15	1425.92	2694.73	15	1428.94	2691.90
	15	1434.08	2689.28	15	1437.10	2686.45	15	1437.21	2683.72	15	1439.32	2682.98
	15	1440.23	2680.89	15	1442.34	2680.15						
5	14	1407.05	2637.89	14	1411.33	2634.57	14	1414.34	2633.56	14	1418.62	2630.24
	14	1421.73	2628.12	14	1428.02	2625.25	14	1429.02	2623.79	14	1435.31	2620.92
	14	1438.42	2618.79	14	1445.71	2614.46	15	1393.60	2752.83	15	1397.88	2749.52
	15	1399.26	2742.53	15	1400.60	2730.05	15	1401.33	2723.34	15	1405.61	2720.02
6	14	1391.02	2642.05	14	1391.50	2640.64	14	1391.53	2638.15	14	1392.02	2636.75
	14	1394.27	2636.40	14	1398.09	2636.26	14	1399.14	2636.10	14	1400.20	2635.52
	14	1400.82	2634.50	14	1401.13	2622.34	14	1401.65	2618.93	14	1403.77	2618.20
	14	1406.02	2617.84	14	1423.88	2616.59	14	1424.39	2615.99	14	1426.65	2615.63
	14	1429.86	2613.23	14	1430.38	2612.62	14	1432.63	2612.27			
7	14	1421.21	2630.05	14	1421.22	2625.77	14	1423.47	2625.41	14	1425.73	2624.89
	14	1425.74	2620.61	14	1428.00	2620.26	15	1390.21	2756.56	15	1390.96	2755.72
	15	1391.53	2753.09	15	1392.28	2752.25	15	1394.47	2740.88	15	1395.22	2740.05
	15	1395.79	2737.41	15	1396.54	2736.58	15	1398.80	2736.22	15	1399.24	2732.90
	15	1400.31	2732.26	15	1400.56	2729.43	15	1402.01	2725.74	15	1402.77	2724.91
	15	1403.34	2722.27	15	1404.09	2721.44	15	1406.35	2721.08	15	1407.86	2717.12
	15	1408.61	2716.28	15	1410.05	2714.29	15	1411.84	2700.20	15	1411.84	2704.48
	15	1412.60	2699.36	15	1414.85	2699.01	15	1416.37	2695.05	15	1416.62	2692.22
	15	1418.88	2691.86	15	1419.39	2685.06	15	1419.39	2689.34	15	1420.15	2684.22
8	14	1405.56	2631.64	14	1407.67	2630.90	14	1407.81	2630.44	14	1409.93	2629.71
	14	1416.47	2628.72	14	1418.73	2627.52	14	1426.94	2626.85	14	1427.43	2626.67
	14	1429.69	2625.48	15	1397.95	2743.56	15	1400.21	2742.36	15	1402.21	2727.88
	15	1404.47	2726.69									
9	14	1452.30	2649.21	14	1457.42	2645.84	15	1432.32	2744.71	15	1433.08	2736.72
	15	1433.86	2729.01	15	1438.57	2727.97	15	1443.64	2703.02	15	1445.76	2699.34
	15	1448.42	2698.33	15	1450.46	2698.30						
10	15	1401.92	2756.50	15	1403.17	2752.28	15	1406.76	2749.71	15	1408.36	2746.09

continued on next page...

...continued from previous page												
run	R	D	T	R	D	T	R	D	T	R	D	T
	15	1411.88	2746.08	15	1411.95	2743.53	15	1412.39	2742.54	15	1413.12	2741.86
	15	1413.64	2738.32	15	1414.51	2738.26	15	1415.24	2737.58	15	1415.75	2734.04
	15	1418.78	2732.20	15	1419.35	2732.08	15	1422.38	2729.64	15	1431.59	2728.69
	15	1433.71	2727.21	15	1435.15	2725.22						
11	14	1421.63	2662.23	14	1423.41	2656.92	14	1423.84	2649.84	14	1425.62	2644.54
	14	1427.02	2638.55	14	1428.80	2633.25	14	1431.06	2632.89	14	1436.35	2626.91
	14	1438.14	2622.50	14	1438.75	2622.45	14	1440.39	2622.15	14	1441.28	2621.06
	14	1441.74	2620.62	14	1444.00	2620.27						
12	14	1403.98	2655.30	14	1405.85	2649.16	14	1408.11	2648.80	14	1414.12	2644.74
	14	1416.38	2644.39	14	1419.41	2636.16	14	1421.67	2635.81	14	1422.37	2635.07
	14	1424.63	2634.71	14	1430.65	2634.35	14	1432.90	2634.00	14	1439.25	2633.70
	14	1439.40	2623.74	15	1400.13	2734.61	15	1402.39	2734.25			
13	14	1386.97	2652.05	14	1397.88	2647.77	14	1400.38	2637.98	14	1402.50	2633.70
	14	1405.46	2632.60	14	1416.60	2630.76	14	1417.07	2629.05	14	1420.03	2627.95
	14	1423.66	2626.13	14	1426.62	2625.03	14	1430.21	2624.30	14	1430.67	2623.86
	14	1433.17	2623.20	14	1433.63	2622.77	14	1444.30	2621.36	14	1444.77	2620.93
	14	1444.77	2619.65	14	1445.23	2619.22	14	1447.73	2618.55	14	1448.19	2618.12
	15	1376.64	2741.70	15	1379.94	2741.61	15	1380.89	2726.02	15	1384.19	2725.94
	15	1385.82	2715.85									
14	14	1402.93	2621.40	14	1405.19	2621.05	14	1408.90	2619.62	14	1411.16	2619.27
	14	1417.06	2613.33	14	1419.09	2611.44	14	1421.35	2611.09	14	1425.05	2609.66
	14	1427.31	2609.31	14	1439.92	2608.74	14	1441.95	2606.85	14	1444.21	2606.50
	14	1447.92	2605.07	14	1450.18	2604.72						
15	14	1417.56	2661.39	14	1418.80	2657.17	14	1421.96	2656.89	14	1422.94	2653.87
	14	1424.18	2649.65	14	1425.88	2646.29	14	1429.03	2646.01	14	1430.76	2644.62
	14	1431.23	2643.01	14	1434.38	2642.73	14	1436.11	2641.33	14	1444.97	2640.29
	14	1448.12	2640.01	14	1449.85	2638.61	15	1409.15	2747.51	15	1410.39	2743.29
16	14	1451.61	2632.57	14	1453.87	2628.03	14	1457.01	2625.24	15	1423.56	2744.40
	15	1425.81	2739.86	15	1428.96	2737.07	15	1431.68	2728.65	15	1432.53	2721.95
	15	1434.79	2717.41	15	1437.94	2714.62	15	1441.53	2684.48	15	1443.79	2679.94
	15	1446.94	2677.15									
17	14	1396.29	2636.22	14	1398.40	2635.48	14	1398.54	2635.02	14	1400.66	2634.28
	14	1404.37	2633.70	14	1404.51	2633.24	14	1406.63	2632.51	14	1407.20	2631.94
	14	1409.46	2630.74	14	1413.17	2630.16	14	1415.42	2628.96	14	1423.99	2625.92
	14	1426.11	2625.19	14	1428.37	2624.88	14	1428.83	2624.45	14	1429.96	2624.14
	14	1432.08	2623.41	14	1434.33	2623.10	14	1434.79	2622.67	15	1387.01	2742.02
	15	1389.27	2740.83	15	1392.97	2740.24	15	1395.23	2739.05			
18	15	1383.38	2756.88	15	1383.40	2750.94	15	1385.66	2750.59	15	1388.35	2748.15
	15	1388.88	2746.97	15	1389.00	2742.86	15	1391.26	2742.51	15	1393.95	2740.07
	15	1394.24	2732.55	15	1394.26	2726.61	15	1396.52	2726.26	15	1398.09	2720.46
	15	1400.35	2720.11	15	1404.66	2714.47	15	1404.68	2708.53	15	1406.94	2708.18

continued on next page...

...continued from previous page												
run	R	D	T	R	D	T	R	D	T	R	D	T
	15	1407.83	2705.74	15	1408.29	2705.31	15	1410.55	2704.96	15	1415.73	2703.91
	15	1417.99	2703.56	15	1419.31	2701.16	15	1421.57	2700.81			
19	15	1444.58	2737.51	15	1444.83	2731.96	15	1447.99	2731.12	15	1450.69	2725.98
	15	1450.94	2720.43	15	1451.61	2719.53	15	1453.37	2718.45	15	1454.05	2717.55
	15	1463.64	2717.06	15	1465.41	2715.97	15	1466.08	2715.07	15	1466.38	2712.76
	15	1467.06	2711.86	15	1468.82	2710.78	15	1469.50	2709.88			
20	15	1418.75	2772.63	15	1420.27	2760.46	15	1420.29	2758.49	15	1422.55	2758.14
	15	1422.88	2756.96	15	1424.40	2744.79	15	1424.42	2742.82	15	1426.68	2742.46
	15	1439.79	2732.49	15	1439.81	2730.52	15	1442.07	2730.17	15	1442.23	2727.77
	15	1444.49	2727.42									
21	14	1396.79	2658.93	14	1398.90	2654.65	14	1402.03	2638.83	14	1404.14	2634.55
	14	1409.61	2632.49	15	1389.18	2770.85	15	1392.78	2768.28			
22	15	1381.13	2737.81	15	1383.25	2737.08	15	1385.39	2722.14	15	1387.65	2721.79
	15	1390.55	2714.82	15	1392.81	2714.47	15	1400.83	2708.65	15	1400.98	2707.54
	15	1403.23	2707.19	15	1406.05	2707.18	15	1406.14	2700.22	15	1408.40	2699.87
	15	1411.21	2699.86	15	1412.02	2696.87	15	1414.27	2696.52	15	1417.34	2696.18
	15	1419.59	2695.83	15	1422.05	2689.35	15	1424.31	2689.00			
23	14	1383.05	2640.19	14	1383.39	2623.91	14	1385.65	2623.55	14	1389.36	2622.13
	14	1391.61	2621.77	15	1381.44	2742.99	15	1381.77	2726.71			
24	14	1447.51	2705.72	14	1448.80	2652.50	14	1452.21	2647.22	14	1452.23	2646.28
	14	1453.50	2645.03	14	1454.31	2639.00	14	1471.48	2636.19	15	1433.23	2792.94
	15	1433.89	2792.68	15	1434.52	2739.72	15	1435.18	2739.46	15	1440.11	2735.16
	15	1440.76	2734.90	15	1440.80	2730.21	15	1443.87	2727.91	15	1445.85	2727.72
	15	1446.51	2727.46	15	1446.72	2726.52						
25	14	1425.91	2660.41	14	1426.61	2653.62	14	1427.95	2650.77	14	1428.66	2643.97
	14	1432.27	2643.17	14	1433.61	2641.18	14	1437.22	2640.38	14	1438.49	2639.43
	14	1443.44	2636.64	14	1453.53	2635.99	15	1387.79	2744.29	15	1388.50	2737.49
	15	1393.00	2736.78	15	1393.03	2730.71	15	1393.71	2729.98	15	1396.51	2729.28
	15	1397.22	2722.48	15	1401.74	2715.70	15	1406.70	2712.91	15	1415.34	2712.83
	15	1419.22	2708.33	15	1419.86	2706.05	15	1424.17	2705.54	15	1424.81	2703.26
26	14	1415.06	2667.71	14	1421.12	2661.30	14	1424.24	2658.88	14	1431.11	2656.48
	14	1433.22	2653.61	14	1438.34	2652.15	14	1438.38	2651.03	14	1441.49	2648.60
	14	1464.25	2640.36									
27	14	1414.31	2649.53	14	1416.57	2649.17	14	1418.25	2641.36	14	1418.90	2638.86
	14	1420.69	2635.49	14	1421.34	2633.00	14	1422.13	2619.82	14	1424.28	2613.23
	14	1426.54	2612.88	14	1426.72	2607.37	14	1428.98	2607.01	14	1429.98	2606.45
	14	1432.23	2606.10	14	1432.86	2605.80	14	1435.12	2605.44	14	1443.41	2604.37
	14	1445.67	2604.02	15	1397.94	2730.80	15	1400.22	2717.80	15	1402.47	2717.45
	15	1404.34	2714.28	15	1405.60	2710.28	15	1405.79	2703.19	15	1408.05	2702.84
28	14	1409.21	2655.47	14	1411.46	2655.12	14	1412.37	2645.25	14	1414.63	2644.89
	14	1417.07	2644.81	14	1420.01	2639.95	14	1422.27	2639.60	14	1423.11	2639.04

continued on next page...

...continued from previous page												
run	R	D	T	R	D	T	R	D	T	R	D	T
	14	1423.34	2636.88	14	1425.59	2636.53	14	1428.03	2636.44	15	1400.50	2768.57
	15	1401.96	2762.34	15	1404.22	2761.98	15	1404.33	2756.29	15	1405.13	2753.36
	15	1407.39	2753.00	15	1407.50	2747.31						
29	14	1421.77	2630.72	14	1421.80	2628.68	14	1432.76	2627.31	14	1437.93	2627.15
	14	1437.95	2625.11	14	1448.91	2623.73	15	1392.01	2741.28	15	1392.03	2739.24
	15	1399.97	2736.24	15	1399.99	2734.20	15	1402.08	2732.56	15	1402.11	2730.52
	15	1404.10	2729.37	15	1404.12	2727.33	15	1409.57	2726.33	15	1412.05	2725.38
	15	1415.82	2718.42	15	1415.85	2716.38	15	1421.66	2714.42			
30	14	1409.11	2649.41	14	1409.13	2641.14	14	1411.25	2640.40	14	1413.90	2636.67
	14	1416.02	2635.93	14	1419.86	2634.89	14	1420.02	2629.52	14	1422.14	2628.78
	14	1425.98	2627.74	14	1428.10	2627.00	15	1399.83	2755.22	15	1399.86	2746.94
	15	1404.09	2739.55	15	1404.11	2731.27	15	1408.88	2726.80			

Table A.4: Number of routes (R), travel distance (D) and delivery time (T) associated with the solutions in the Pareto approximations obtained by NSGA-II for instance R105.

Table A.5 presents, for Solomon's instance R105, the $\mathsf{M_D}$ values corresponding to the Pareto approximations result from each run of MOEA-RDT. These $\mathsf{M_D}$ values are normalised according to the reference point $\boldsymbol{z} = (100.00, 4989.42,\ 13017.71)$. The average of these 30 $\mathsf{M_D}$ values is 0.0057.

run	$\mathsf{M_D}$	run	$\mathsf{M_D}$	run	$\mathsf{M_D}$	run	$\mathsf{M_D}$	run	$\mathsf{M_D}$	run	$\mathsf{M_D}$
1	0.0119	6	0.0018	11	0.0030	16	0.0096	21	0.0036	26	0.0048
2	0.0049	7	0.0054	12	0.0035	17	0.0025	22	0.0055	27	0.0029
3	0.0073	8	0.0033	13	0.0023	18	0.0052	23	0.0013	28	0.0047
4	0.0102	9	0.0109	14	0.0015	19	0.0143	24	0.0102	29	0.0049
5	0.0034	10	0.0082	15	0.0044	20	0.0110	25	0.0050	30	0.0039

Table A.5: $\mathsf{M_D}$ values corresponding to the Pareto approximations, result from each run of NSGA-II, to Solomon's instance R105.

Similarly, Table A.6 presents, for Solomon's instance R105, the $\mathsf{M_H}$ values corresponding to the Pareto approximations result from each run of NSGA-II. These $\mathsf{M_H}$ values are normalised according to the reference point $\boldsymbol{z} = (100.00, 4989.42,$ $13017.71)$. The average of these 30 $\mathsf{M_H}$ values is 0.6035.

run	$\mathsf{M_H}$	run	$\mathsf{M_H}$	run	$\mathsf{M_H}$	run	$\mathsf{M_H}$	run	$\mathsf{M_H}$	run	$\mathsf{M_H}$
1	0.6050	6	0.6117	11	0.6066	16	0.6037	21	0.6027	26	0.6077
2	0.6045	7	0.6078	12	0.6030	17	0.6046	22	0.6022	27	0.6051
3	0.6021	8	0.6038	13	0.6070	18	0.6011	23	0.6055	28	0.6025
4	0.5982	9	0.6014	14	0.6108	19	0.5927	24	0.5998	29	0.6061
5	0.6049	10	0.5975	15	0.6014	20	0.5951	25	0.6075	30	0.6037

Table A.6: $\mathsf{M_H}$ values corresponding to the Pareto approximations, result from each run of NSGA-II, to Solomon's instance R105.

The result of the two-tailed t-test for two samples with unequal variance applied to the values in Tables A.2 and A.5 is 0.0, which indicates that the true means of both series of values do differ. The same result is obtained when the t-test is applied to the values in Tables A.3 and A.6.

Bibliography

[1] E. Aarts and J. K. Lenstra, editors. *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[2] E. Aarts and J. K. Lenstra. Introduction. In E. Aarts and J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[3] D. H. Ackley. *A connectionist machine for genetic hillclimbing*. Kluwer Academic Publishers, 1987.

[4] H. E. Aguirre and K. Tanaka. Working principles, behavior, and performance of MOEAs on MNK-landscapes. *Eur. J. Oper. Res.*, 181(3):1670 – 1690, 2007.

[5] E. Alba and B. Dorronsoro. Computing nine new best-so-far solutions for capacitated VRP with a cellular genetic algorithm. *Inform. Process. Lett.*, 98 (6):225–230, 2006.

[6] E. Alba and B. Dorronsoro. A hybrid cellular genetic algorithm for the capacitated vehicle routing problem. In A. Abraham, C. Grosan, and W. Pedrycz, editors, *Engineering Evolutionary Intelligent Systems*, volume 82 of *SCI*, pages 379–422. Springer, 2008.

[7] G. B. Alvarenga, G. R. Mateus, and G. de Tomi. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Comput. Oper. Res*, 34(6):1561–1584, 2007.

[8] C. Archetti and M. G. Speranza. An overview on the split delivery vehicle routing problem. In K.-H. Waldmann and U. M. Stocker, editors, *Operations Research*, pages 123–127. Springer, 2006.

[9] C. Archetti and M. G. Speranza. The split delivery vehicle routing problem: A survey. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 103–122. Springer, 2008.

[10] T. Bäck. Binary strings. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, 2000.

[11] B. M. Baker and M. A. Ayechew. A genetic algorithm for the vehicle routing problem. *Comput. Oper. Res.*, 30(5):787–800, 2003.

[12] E. Balas and M. W. Padberg. Set partitioning: A survey. *SIAM Rev.*, 18(4): 710–760, 1976.

[13] R. Baldacci, P. Toth, and D. Vigo. Recent advances in vehicle routing exact algorithms. *4OR-Q. J. Oper. Res.*, 5(4):269–298, 2007.

[14] R. Baldacci, M. Battarra, and D. Vigo. Routing a heterogeneous fleet of vehicles. In B. L. Golden, S. Raghavan, and E. A. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 3–27. Springer, 2008.

[15] R. Baldacci, P. Toth, and D. Vigo. Exact algorithms for routing problems under vehicle capacity constraints. *Ann. Oper. Res.*, 175(1):213–245, 2010.

[16] A. Beham. Parallel tabu search and the multiobjective vehicle routing problem with time windows. In *21th International Parallel and Distributed Processing Symposium*, pages 1–8. IEEE Computer Society, 2007.

[17] R. Bent and P. Van Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows. *Transport. Sci.*, 38(4):515–530, 2004.

[18] J. J. Bentley. Fast algorithms for geometric traveling salesman problems. *INFORMS J. Comput.*, 4(4):387–411, 1992.

[19] J. Berger and M. Barkaoui. A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Comput. Oper. Res.*, 31(12):2037–2053, 2004.

[20] J. Berger, M. Barkaoui, and O. Bräysi. A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR*, 41(2): 179–194, 2003.

[21] D. J. Bertsimas. A vehicle routing problem with stochastic demand. *Oper. Res.*, 40(3):574–585, 1992.

[22] T. Bickle. Tournament selection. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, 2000.

[23] P. Bienert. *Aufbau einer Optimierungsautomatik für drei Parameter*. PhD thesis, Technical University of Berlin, Institute of Measurement and Control Technology, Berlin, Germany, 1967.

[24] F. Bock. An algorithm for solving 'traveling salesman' and related network optimization problems. In *14th National Meeting of ORSA*, 1958.

[25] G. E. P. Box. Evolutionary operation: A method for increasing industrial productivity. *J. Roy. Stat. Soc. C-App.*, 6(2):81–101, 1957.

[26] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Transport. Sci.*, 39(1):104–118, 2005.

[27] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part II: Metaheuristics. *Transport. Sci.*, 39(1):119–139, 2005.

[28] O. Bräysy, W. Dullaert, and M. Gendreau. Evolutionary algorithms for the vehicle routing problem with time windows. *J. Heuristics*, 10(6):587–611, 2004.

[29] H.-J. Bremermann. Optimization through evolution and recombination. In M. C. Yovits, G. T. Jacobi, and G. D. Goldstein, editors, *Self-organizing systems*, pages 93–106. Spartan Books, 1962.

[30] B. Bullnheimer, R. F. Hartl, and C. Strauss. Applying the ant system to the vehicle routing problem. In *2nd Metaheuristics International Conference*, pages 1–12, 1997.

[31] B. Bullnheimer, R. F. Hartl, and C. Strauss. An improved ant system algorithm for the vehicle routing problem. *Ann. Oper. Res.*, 89(1):319–328, 1999.

[32] R. Burkard and S. M. Mauro Dell'Amico. *Assignment Problems*. SIAM, 2009.

[33] E. K. Burke, S. Gustafson, and G. Kendall. Diversity in genetic programming: an analysis of measures and correlation with fitness. *IEEE T. Evolut. Comput.*, 8(1):47–62, 2004.

[34] J. P. Castro Gutiérrez, D. Landa-Silva, and J. A. Moreno-Pérez. Exploring feasible and infeasible regions in the vehicle routing problem with time windows using a multi-objective particle swarm optimization approach. In N. Krasnogor, B. Melián-Batista, J. A. Moreno-Pérez, J. M. Moreno-Vega, and D. A. Pelta, editors, *Nature Inspired Cooperative Strategies for Optimization*, volume 236 of *SCI*, pages 103–114. Springer, 2009.

[35] N. Christofides. The travelling salesman problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, *Combinatorial Optimization*, pages 131–149. John Wiley & Sons, 1979.

[36] N. Christofides and S. Eilon. Algorithms for large-scale travelling salesman problems. *Oper. Res Quart*, 23(4):511–518, 1972.

[37] N. Christofides, A. Mingozzi, and P. Toth. The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, *Combinatorial Optimization*, pages 315–338. John Wiley & Sons, 1979.

[38] G. Clarke and J. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Oper. Res.*, 12(4):568–581, 1964.

[39] C. A. Coello Coello. A short tutorial on evolutionary multiobjective optimization. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, volume 1993 of *LNCS*, pages 21–40. Springer, 2001.

[40] C. A. Coello Coello. 20 years of evolutionary multi-objective optimization: What has been done and what remains to be done. In G. Yen and D. Fogel, editors, *Computational Intelligence: Principles and Practice*, pages 73–88. IEEE Press, 2006.

[41] C. A. Coello Coello and M. Salazar Lechuga. MOPSO: A proposal for multiple objective particle swarm optimization. In *2002 IEEE Congress on Evolutionary Computation*, pages 1051–1056. IEEE Press, 2002.

[42] C. A. Coello Coello, G. L. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2007.

[43] E. G. Coffman, G. Galambos, S. Martello, and D. Vigo. Bin packing approximation algorithms: Combinatorial analysis. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*. Kluwer Academic Publishers, 1998.

[44] Y. Collette and P. Siarry. *Multiobjective Optimization: Principles and Case Studies*. Springer, August 2003.

[45] A. Corberán, E. Fernández, M. Laguna, and R. Martí. Heuristic solutions to the problem of routing school buses with multiple objectives. *J. Oper. Res. Soc.*, 53(4):427–435, 2002.

[46] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon, and F. Soumis. VRP with time windows. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 157–193. SIAM, 2001.

[47] J.-F. Cordeau, G. Laporte, and A. Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *J. Oper. Res. Soc.*, 52(8): 928–936, 2001.

[48] J.-F. Cordeau, M. Gendreau, G. Laporte, J.-Y. Potvin, and F. Semet. A guide to vehicle routing heuristics. *J. Oper. Res. Soc.*, 53(5):512–522, 2002.

[49] J.-F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.-S. Sormany. New heuristics for the vehicle routing problem. In A. Langevin and D. Riopel, editors, *Logistics Systems: Design and Optimization*, pages 279–297. Springer, 2005.

[50] R. Cordone and R. Wolfler Calvo. A heuristic for the vehicle routing problem with time windows. *J. Heuristics*, 7(2):107–129, 2001.

[51] G. A. Croes. A method for solving traveling-salesman problems. *Oper. Res.*, 6(6):791–812, 1958.

[52] Z. J. Czech and P. Czarnas. Parallel simulated annealing for the vehicle routing problem with time windows. In *10th Euromicro Workshop on Parallel, Distributed, and Network-Based Processing*, pages 376–383. IEEE Computer Society, 2002.

[53] P. Czyzak and A. Jaszkiewicz. Pareto simulated annealing - a metaheuristic technique for multiple-objective combinatorial optimization. *J. Multi-Criteria*

Decis. Anal., 7(1):34–47, 1998.

[54] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Manage. Sci.*, 6(1):80–91, 1959.

[55] I. Das and J. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Struct. Optimization*, 14(1):63–69, 1997.

[56] B. De Backer, V. Furnon, P. Kilby, P. Prosser, and P. Shaw. Local search in constraint programming: Application to the vehicle routing problem. In *1997 Workshop on Industrial Constraint Directed Scheduling*, pages 1–15, 1997.

[57] K. De Jong. *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, University of Michigan, Ann Arbor, Michigan, USA, 1975.

[58] K. De Jong. *Evolutionary computation: a unified approach*. MIT Press, 2006.

[59] K. De Jong. Parameter setting in EAs: a 30 year perspective. In F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms*, pages 1–18. Springer, 2007.

[60] K. De Jong, D. B. Fogel, and H.-P. Schwefel. A history of evolutionary computation. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators*, pages 40–58. Institute of Physics Publishing, 2000.

[61] K. Deb. Introduction to representations. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, 2000.

[62] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[63] K. Deb. Evolutionary multi-objective optimization without additional parameters. In F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors, *Parameter setting in evolutionary algorithms*, pages 241–257. Springer, 2007.

[64] K. Deb. Current trends in evolutionary multi-objective optimization. *Int. J. Simul. Multidiscip. Design Opt.*, 1(1):1–8, 2007.

[65] K. Deb and S. Jain. Running performance metrics for evolutionary multi-objective optimization. In L. Wang, K. C. Tan, T. Furuhashi, J.-H. Kim, and X. Yao, editors, *Fourth Asia-Pacific Conference on Simulated Evolution and Learning*, pages 13–20, 2002.

[66] K. Deb and S. Tiwari. Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization. *Eur. J. Oper. Res.*, 185(3):1062 – 1087, 2008.

[67] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE T. on Evolut. Comput.*, 6(2): 182–197, 2002.

[68] K. Deb, M. Mohan, and S. Mishra. Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. *Evolut. Comput.*, 13(4):501–525, 2005.

[69] G. Desaulniers, J. Desrosiers, A. Erdmann, M. M. Solomon, and F. Soumis. VRP with pickup and delivery. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 225–242. SIAM, 2001.

[70] M. Desrochers, J. K. Lenstra, and M. W. P. Savelsbergh. A classification scheme for vehicle routing and scheduling problems. *Eur. J. Oper. Res.*, 46 (3):322–332, 1990.

[71] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.*, 40(2):342–354, 1992.

[72] M. Desrochers, C. V. Jones, J. K. Lenstra, M. W. P. Savelsbergh, and L. Stougie. Towards a model and algorithm management system for vehicle routing and scheduling problems. *Decis. Support Syst.*, 25(2):109–133, 1999.

[73] M. M. Deza and E. Deza. *Encyclopedia of Distances*. Springer, 2009.

[74] K. F. Doerner and R. F. Hartl. Health care logistics, emergency preparedness, and disaster relief: New challenges for routing problems with a focus on the austrian situation. In B. L. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 527–550. Springer, 2008.

[75] K. F. Doerner, A. Focke, and W. J. Gutjahr. Multicriteria tour planning for mobile healthcare facilities in a developing country. *Eur. J. Oper. Res.*, 179 (3):1078–1096, 2007.

[76] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, 2004.

[77] M. Dror and P. Trudeaut. Savings by split delivery routing. *Transport. Sci.*, 23(2):141, 1989.

[78] M. Dror, G. Laporte, and P. Trudeau. Vehicle routing with split deliveries. *Discrete Appl. Math.*, 50(3):239 – 254, 1994.

[79] C. Duhamel, J.-Y. Potvin, and J.-M. Rousseau. A tabu search heuristic for the vehicle routing problem with backhauls and time windows. *Transport. Sci.*, 31(1):49–59, 1997.

[80] W. Dullaert and O. Bräysy. Routing relatively few customers per route. *TOP*, 2(11):325–336, 2002.

[81] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.

[82] A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J. E. Smith. Parameter control in evolutionary algorithms. In F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors, *Parameter setting in evolutionary algorithms*, pages 19–46. Springer, 2007.

[83] B. Eksioglu, A. V. Vural, and A. Reisman. The vehicle routing problem: A taxonomic review. *Comput. Ind. Eng.*, 57(4):1472–1483, 2009.

[84] H. Esbensen and E. Kuh. Design space exploration using the genetic algorithm. In *1996 IEEE International Symposium on Circuits and Systems*, volume 4, pages 500–503. IEEE Press, 1996.

[85] E. Falkenauer. A hybrid grouping genetic algorithm for bin packing. *J. Heuristics*, 2(1):5–30, 1996.

[86] T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *J. Global Optim.*, 6(2):109–133, 1995.

[87] M. L. Fisher and R. Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124, 1981.

[88] M. M. Flood. The traveling-salesman problem. *Oper. Res.*, 4(1):61–75, 1956.

[89] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, 1966.

[90] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest, editor, *5th International Conference on Genetic Algorithms*, pages 416–423. Morgan Kaufmann, 1993.

[91] B. A. Foster and D. M. Ryan. An integer programming approach to the vehicle scheduling problem. *Oper. Res. Quart.*, 27(2):367–384, 1976.

[92] P. M. Francis, K. R. Smilowitz, and M. Tzur. The period vehicle routing problem and its extensions. In B. Golden, S. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 73–102. Springer, 2008.

[93] J. Franks. *A (Terse) Introduction to Lebesgue Integration*. AMS, 2009.

[94] A. S. Fraser. Simulation of genetic systems by automatic digital computers. *Aust. J. Biol. Sci.*, 10:484–491, 1957.

[95] R. M. Friedberg. A learning machine: Part I. *IBM J. Res. Dev.*, 2(1):2–13, 1958.

[96] R. M. Friedberg, B. Dunham, and J. H. North. A learning machine: Part II. *IBM J. Res. Dev.*, 3(3):282–287, 1959.

[97] L. M. Gambardella, E. Taillard, and G. Agazzi. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, and K. V. Price, editors, *New Ideas in Optimization*, pages 63–76. McGraw-Hill, 1999.

[98] A. Garcia-Najera. Preserving population diversity for the multi-objective vehicle routing problem with time windows. In F. Rothlauf, editor, *Genetic and Evolutionary Computation Conference 2009*, pages 2689–2692. ACM, 2009.

[99] A. Garcia-Najera and J. A. Bullinaria. A multi-objective density restricted genetic algorithm for the vehicle routing problem with time windows. In *2008 UK Workshop on Computational Intelligence*, pages 47–52, 2008.

[100] A. Garcia-Najera and J. A. Bullinaria. Bi-objective optimization for the vehicle routing problem with time windows: Using route similarity to enhance performance. In M. Ehrgott, C. Fonseca, X. Gandibleux, J. K. Hao, and M. Sevaux, editors, *5th International Conference on Evolutionary Multi-Criterion Optimization*, volume 5467 of *LNCS*, pages 275–289. Springer, 2009.

[101] A. Garcia-Najera and J. A. Bullinaria. Comparison of similarity measures for the multi-objective vehicle routing problem with time windows. In F. Rothlauf, editor, *Genetic and Evolutionary Computation Conference 2009*, pages 579–586. ACM, 2009.

[102] A. Garcia-Najera and J. A. Bullinaria. Optimizing delivery time in multi-objective vehicle routing problems with time windows. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *11th International Conference on Parallel Problem Solving From Nature*, volume 6239 part II of *LNCS*, pages 51–60. Springer, 2010.

[103] A. Garcia-Najera and J. A. Bullinaria. An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Comput. Oper. Res.*, 38(1):287–300, 2011.

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

[105] M. Gendreau and C. D. Tarantilis. Solving large-scale vehicle routing problems with time windows: The state-of-the-art. Technical Report CIRRELT-2010-04, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Montreal, Canada, 2010.

[106] M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Manage. Sci.*, 40(10):1276–1290, 1994.

[107] M. Gendreau, G. Laporte, and R. Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transport. Sci.*, 29 (2):143–155, 1995.

[108] M. Gendreau, G. Laporte, and R. Séguin. Stochastic vehicle routing. *Eur. J. Oper. Res.*, 88(1):3–12, 1996.

[109] M. Gendreau, G. Laporte, and J.-Y. Potvin. Metaheuristics for the capacitated VRP. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 129–

154. SIAM, 2001.

[110] M. Gendreau, G. Laporte, and J.-Y. Potvin. Vehicle routing: modern heuristics. In E. Aarts and J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[111] K. Ghoseiri and S. F. Ghannadpour. Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Appl. Soft. Comput.*, 10(4):1096–1107, 2010.

[112] I. Giannikos. A multiobjective programming model for locating treatment sites and routing hazardous wastes. *Eur. J. Oper. Res.*, 104(2):333–342, 1998.

[113] B. E. Gillett and L. R. Miller. A heuristic algorithm for the vehicle-dispatch problem. *Oper. Res.*, 22(2):340–349, 1974.

[114] I. D. Giosa, I. L. Tansini, and I. O. Viera. New assignment algorithms for the multi-depot vehicle routing problem. *J. Oper. Res. Soc*, 53(9):977–984, 2002.

[115] F. Glover. Tabu Search–Part I. *INFORMS J. Comput.*, 1(3):190–206, 1989.

[116] F. Glover. Tabu Search–Part II. *INFORMS J. Comput.*, 2(1):4–32, 1990.

[117] F. Glover and M. Laguna. *Tabu Search*. Springer, 1998.

[118] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[119] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlins, editor, *First Workshop on Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[120] D. E. Goldberg and R. L. Jr. Alleles, loci, and the traveling salesman problem. In J. J. Grefenstette, editor, *1st International Conference on Genetic Algorithms and their Applications*, pages 10–19. Lawrence Erlbaum Associates, 1985.

[121] B. L. Golden, S. Raghavan, and E. Wasil, editors. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, 2008.

[122] J. Grefenstette. Proportional selection and sampling algorithms. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, 2000.

[123] G. Gutin and A. Punnen. *Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.

[124] E. Hadjiconstantinou and D. Roberts. Routing under uncertainty: an application in the scheduling of field service engineers. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 331–352. SIAM, 2001.

[125] R. W. Hamming. Error detecting and error correcting codes. *Bell Syst. Tech. J.*, 26(2):147–160, 1950.

[126] J. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, 1975.

[127] J. Homberger. *Verteilt-Parallele Metaheuristiken zur Tourenplanung*. PhD thesis, FernUniversitt, Hagen, Germany, 2000.

[128] J. Homberger and H. Gehring. Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR*, 37(3):297 – 318, 1999.

[129] J. Homberger and H. Gehring. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *Eur. J. Oper. Res.*, 162(1):220–238, 2005.

[130] T. Ibaraki, S. Imahori, M. Kubo, T. Masuda, T. Uno, and M. Yagiura. Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transport. Sci.*, 39(2):206–232, 2005.

[131] P. Jaccard. Etude comparative de la distribution florale dans une portion des alpes et des jura. *Bull. Soc. Vaudoise Sci. Nat.*, 37:547–579, 1901.

[132] D. S. Johnson and L. A. McGeoch. The traveling salesman problem: A case study in local optimization. In E. Aarts and J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[133] N. Jozefowiez, F. Semet, and E.-G. Talbi. Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem. In J. J. M. Guervós, P. Adamidis, H.-G. Beyer, J. L. F.-V. Martín, and H.-P. Schwefel, editors, *7th International Conference on Parallel Problem Solving from Nature*, volume 2439 of *LNCS*, pages 271–280. Springer, 2002.

[134] N. Jozefowiez, F. Semet, and E.-G. Talbi. Enhancements of NSGA II and its application to the vehicle routing problem with route balancing. In E.-

G. Talbi, P. Liardet, P. Collet, E. Lutton, and M. Schoenauer, editors, *7th International Conference on Artificial Evolution*, volume 3871 of *LNCS*, pages 131–142, 2006.

[135] N. Jozefowiez, F. Semet, and E.-G. Talbi. Multi-objective vehicle routing problems. *Eur. J. Oper. Res.*, 189(2):293–309, 2008.

[136] N. Jozefowiez, F. Semet, and E.-G. Talbi. An evolutionary algorithm for the vehicle routing problem with route balancing. *Eur. J. Oper. Res.*, 195(3): 761–769, 2009.

[137] S. Jung and B. R. Moon. A hybrid genetic algorithm for the vehicle routing problem with time windows. In *Genetic and Evolutionary Computation Conference 2002*, pages 1309–1316. Morgan Kaufmann, 2002.

[138] B. Kallehauge. Formulations and exact algorithms for the vehicle routing problem with time windows. *Comput. Oper. Res.*, 35(7):2307–2330, 2008.

[139] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.

[140] J. Kennedy, R. C. Eberhart, and Y. Shi. *Swarm Intelligence*. Morgan Kaufmann, 2001.

[141] V. R. Khare, X. Yao, and K. Deb. Performance scaling of multi-objective evolutionary algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *2nd International Conference on Evolutionary Multi-Criterion Optimization*, volume 2632 of *LNCS*, pages 376–390. Springer, 2003.

[142] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[143] J. Knowles, L. Thiele, and E. Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical Report TIK 214, Computer Engineering and Networks Laboratory (TIK), ETH, Zurich, Switzerland, 2006.

[144] J. D. Knowles and D. W. Corne. The Pareto Archived Evolution Strategy: A new baseline algorithm for Pareto multiobjective optimisation. In *1999 IEEE Congress on Evolutionary Computation*, pages 98–105. IEEE Press, 1999.

[145] J. D. Knowles and D. W. Corne. Approximating the nondominated front using the Pareto Archived Evolution Strategy. *Evol. Comput.*, 8(2):149–172, 2000.

[146] J. B. Kollat and P. M. Reed. Comparing state-of-the-art evolutionary multi-objective algorithms for long-term groundwater monitoring design. *Adv. Water Resour.*, 29(6):792–807, 2006.

[147] A. Konak, D. W. Coit, and A. E. Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliab. Eng. Syst. Safe.*, 91(9):992–1007, 2006.

[148] G. Kontoravdis and J. F. Bard. A GRASP for the vehicle routing problem with time windows. *INFORMS J. Comput.*, 7(1):10–23, 1995.

[149] E. F. Krause. *Taxicab geometry: an adventure in non-Euclidean geometry*. Dover Publications, 1986.

[150] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[151] H. W. Kuhn. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.*, 2(1):83–97, 1955.

[152] F. Kursawe. A variant of evolution strategies for vector optimization. In H.-P. Schwefel and R. Männer, editors, *1st Workshop on Parallel Problem Solving from Nature*, volume 496 of *LNCS*, pages 193–197. Springer, 1991.

[153] G. Laporte. Fifty years of vehicle routing. *Transport. Sci.*, 43(4):408–416, 2009.

[154] G. Laporte and F. Semet. Classical heuristics for the capacitated VRP. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 109–128. SIAM, 2001.

[155] G. Laporte, F. Louveaux, and H. Mercure. The vehicle routing problem with stochastic travel times. *Transport. Sci.*, 26(3):161–170, 1992.

[156] G. Laporte, M. Gendreau, J.-Y. Potvin, and F. Semet. Classical and modern heuristics for the vehicle routing problem. *Int. Trans. Oper. Res.*, 7(4):285–300, 2000.

[157] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolut. Comput.*, 10(3): 263–282, 2002.

[158] A. Le Bouthillier and T. G. Crainic. A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. *Comput. Oper. Res.*, 32(7): 1685–1708, 2005.

[159] J. K. Lenstra and A. H. G. R. Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.

[160] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Doklady*, 10(8):707–710, 1966.

[161] H. Li and A. Lim. Local search with annealing-like restarts to solve the vrptw. *Eur. J. Oper. Res.*, 150(1):115–127, 2003.

[162] S. Lin. Computer solutions of the traveling salesman problem. *Bell Syst. Tech. J.*, 44(10):2245–2269, 1965.

[163] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.*, 21(2):498–516, 1973.

[164] F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors. *Parameter Setting in Evolutionary Algorithms*. Springer, 2007.

[165] Y. Marinakis and A. Migdalas. Annotated bibliography in vehicle routing. *Oper. Res. Int. J.*, 7(1):27–46, 2007.

[166] R. Martí, M. Laguna, and V. Campos. Scatter search vs. genetic algorithms: An experimental evaluation with permutation problems. In C. Rego and B. Alidaee, editors, *Metaheuristic Optimization Via Adaptive Memory and Evolution: Tabu Search and Scatter Search*, pages 263–282. Kluwer Academic Publishers, 2005.

[167] R. McGill, J. W. Tukey, and W. A. Larsen. Variations of box plots. *Am. Stat.*, 32(1):12–16, 1978.

[168] Q. Meng, D.-H. Lee, and R. L. Cheu. Multiobjective vehicle routing and scheduling problem with time window constraints in hazardous material transportation. *J. Transp. Eng.*, 131(9):699–707, 2005.

[169] D. Mester. An evolutionary strategies algorithm for large scale vehicle routing problem with capacitate and time windows restrictions. Working paper. Institute of Evolution, University of Haifa, Israel, 2002.

[170] D. Mester and O. Bräysy. Active guided evolution strategies for large-scale vehicle routing problems with time windows. *Comput. Oper. Res.*, 32(6):1593–1614, 2005.

[171] D. Mester, O. Bräysy, and W. Dullaert. A multi-parametric evolution strategies algorithm for vehicle routing problems. *Expert Syst. Appl.*, 32(2):508–517, 2007.

[172] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, 1996.

[173] Z. Michalewicz and M. Schmidt. Parameter control in practice. In F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors, *Parameter setting in evolutionary algorithms*, pages 277–294. Springer, 2007.

[174] M. Minsky. Steps toward artificial intelligence. In E. A. Feigenbaum and J. Feldman, editors, *Computers and Thought*. McGraw-Hill, 1963.

[175] R. H. Mole and S. R. Jameson. A sequential route-building algorithm employing a generalized savings criterion. *Oper. Res. Quart.*, 27(1):503–511, 1976.

[176] M. J. Morgan and C. L. Mumford. Capacitated vehicle routing: perturbing the landscape to fool an algorithm. In D. Corne, Z. Michalewicz, B. McKay, G. Eiben, D. Fogel, C. Fonseca, G. Greenwood, G. Raidl, K. C. Tan, and A. Zalzala, editors, *2005 IEEE Congress on Evolutionary Computation*, pages 2271–2277. IEEE Press, 2005.

[177] T. Murata and R. Itai. Multi-objective vehicle routing problems using two-fold EMO algorithms to enhance solution similarity on non-dominated solutions. In C. A. Coello Coello, A. Hernandez, and E. Zitzler, editors, *3rd International Conference on Evolutionary Multi-Criteria Optimization*, volume 3410 of *LNCS*, pages 885–896. Springer, 2005.

[178] Y. Nagata. Edge assembly crossover for the capacitated vehicle routing problem. In *7th European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 142–153. Springer, 2007.

[179] Y. Nagata and O. Bräysy. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. *Networks*, 54(4):205–215, 2009.

[180] Y. Nagata and S. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the travelling salesman problem. In T. Bäck, editor, *7th International Conference on Genetic Algorithms*, pages 450–457. Morgan Kaufmann, 1997.

[181] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the traveling salesman problem. In J. J. Grefenstette, editor, *Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 224–230. Lawrence Erlbaum Associates, 1987.

[182] B. Ombuki, B. J. Ross, and F. Hanshar. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Appd. Intel.*, 24(1):17–30, 2006.

[183] I. Or. *Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking*. PhD thesis, Northwestern University, 1976.

[184] I. H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems. *Ann. Oper. Res.*, 41(4):421–451, 1993.

[185] J. Pacheco and R. Martí. Tabu search for a multi-objective routing problem. *J. Oper. Res. Soc.*, 57(1):29–37, 2006.

[186] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1998.

[187] J. M. Pasia, K. F. Doerner, R. F. Hartl, and M. Reimann. A population-based local search for solving a bi-objective vehicle routing problem. In C. Cotta and J. I. van Hemert, editors, *7th European Conference on Evolutionary Computation in Combinatorial Optimization*, volume 4446 of *LNCS*, pages 166–175. Springer, 2007.

[188] F. B. Pereira and J. Tavares, editors. *Bio-inspired Algorithms for the Vehicle Routing Problem*. Springer, 2008.

[189] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Comput. Oper. Res.*, 34(8):2403–2435, 2007.

[190] J.-Y. Potvin. A review of bio-inspired algorithms for vehicle routing. In F. B. Pereira and J. Tavares, editors, *Bio-inspired Algorithms for the Vehicle Routing Problem*, volume 161 of *SCI*, pages 1–34. Springer, 2009.

[191] J.-Y. Potvin. Evolutionary algorithms for vehicle routing. *INFORMS J. Comput.*, 21(4):518–548, 2009.

[192] J.-Y. Potvin and S. Bengio. The vehicle routing problem with time windows — part II: Genetic search. *INFORMS J. Comput.*, 8(2):165–172, 1996.

[193] J.-Y. Potvin and J.-M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *Eur. J. Oper. Res.*, 66(3):331–340, 1993.

[194] J.-Y. Potvin, T. Kervahut, B.-L. Garcia, and J.-M. Rousseau. The vehicle routing problem with time windows — part I: Tabu search. *INFORMS J. Comput.*, 8(2):158–164, 1996.

[195] C. Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Comput. Oper. Res.*, 31(12):1985–2002, 2004.

[196] C. Prins. A GRASP × evolutionary local search hybrid for the vehicle routing problem. In F. B. Pereira and J. Tavares, editors, *Bio-inspired Algorithms for the Vehicle Routing Problem*, volume 161 of *SCI*, pages 35–53. Springer, 2009.

[197] M. Rahoual, B. Kitoun, M.-H. Mabed, V. Bachelet, and F. Benameur. Multicriteria genetic algorithms for the vehicle routing problem with time windows. In *4th Metaheuristics International Conference*, pages 527–532, 2001.

[198] I. Rechenberg. Cybernetic solution path of an experimental problem. Technical report, Royal Aircraft Establishment, 1965.

[199] I. Rechenberg. *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog, 1973.

[200] C. R. Reeves and J. E. Beasley. Introduction. In C. R. Reeves, editor, *Modern heuristic techniques for combinatorial problems*, pages 1–19. John Wiley & Sons, 1993.

[201] J. Renaud, F. F. Boctor, and G. Laporte. An improved petal heuristic for the vehicle routeing problem. *J. Oper. Res. Soc.*, 47(2):329–336, 1996.

[202] J. Renaud, G. Laporte, and F. F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Comput. Oper. Res.*, 23(3):229–235, 1996.

[203] P. P. Repoussis, C. D. Tarantilis, and G. Ioannou. Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE Trans. Evolut. Comput.*, 13(3):624–647, 2009.

[204] Y. Rochat and E. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *J. Heuristics*, 1(1):147–167, 1995.

[205] S. Ronald. More distance functions for order-based encodings. In *1998 IEEE International Conference on Evolutionary Computation*, pages 558–563. IEEE Press, 1998.

[206] S. Ropke and D. Pisinger. A unified heuristic for a large class of vehicle routing problems with backhauls. *Eur. J. Oper. Res.*, 171(3):750–775, 2006.

[207] L.-M. Rousseau, M. Gendreau, and G. Pesant. Using constraint-based operators to solve the vehicle routing problem with time windows. *J. Heuristics*, 8 (1):43–58, 2002.

[208] D. M. Ryan, C. Hjorring, and F. Glover. Extensions of the petal method for vehicle routeing. *J. Oper. Res. Soc.*, 44(3):289–296, 1993.

[209] F. E. Satterthwaite. Random balance experimentation. *Technometrics*, 1(2): 111–137, 1959.

[210] M. W. P. Savelsbergh. The vehicle routing problem with time windows: Minimizing route duration. *INFORMS J. Comput.*, 4(2):146–154, 1992.

[211] M. W. P. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transport. Sci.*, 29(1):17–29, 1995.

[212] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, *First International Conference on Genetic Algorithms and their Applications*, pages 93–100. Lawrence Erlbaum Associates, 1985.

[213] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *J. Comput. Phys.*, 159(2):139–171, 2000.

[214] H.-P. Schwefel. *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik*. PhD thesis, Technical University of Berlin, Hermann Föttinger Institute for Hydrodynamics, Berlin, Germany, 1965.

[215] P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. Working paper, 1997. University of Strathclyde, Glasgow, United Kingdom.

[216] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In M. Maher and J.-F. Puget, editors, *4th International Conference on Principles and Practice of Constraint Programming*, volume 1520 of *LNCS*, pages 417–431. Springer, 1998.

[217] B. W. Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, 1986.

[218] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.*, 35(2):254–265, 1987.

[219] K. Sörensen. Distance measures based on the edit distance for permutation-type representations. In *Workshop on Analysis and Design of Representations and Operators*, pages 15–21, 2003.

[220] K. Sörensen. Distance measures based on the edit distance for permutation-type representations. *J. Heuristics*, 13(1):35–47, 2007.

[221] K. Sörensen and M. Sevaux. MA|PM: memetic algorithms with population management. *Comput. Oper. Res.*, 33(5):1214–1225, 2006.

[222] W. Spendley, G. R. Hext, and F. R. Himsworth. Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, 4 (4):441–461, 1962.

[223] N. Srinivas and K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evol. Comput.*, 2(3):221–248, 1994.

[224] G. Syswerda. Uniform crossover in genetic algorithms. In J. D. Schaffer, editor, *3rd International Conference in Genetic Algorithms*, pages 2–9. Morgarn Kaufmann, 1989.

[225] E. Taillard. Parallel iterative search methods for vehicle routing problems. *Networks*, 23(8):661–673, 1993.

[226] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows. *Transport. Sci.*, 31(2):170–186, 1997.

[227] C. Tan and J. Beasley. A heuristic algorithm for the period vehicle routing problem. *Omega*, 12(5):497–504, 1984.

[228] K. C. Tan, Y. H. Chew, and L. H. Lee. A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Comput. Optim. and Appl.*, 34(1):115–151, 2006.

[229] K. C. Tan, Y. H. Chew, and L. H. Lee. A hybrid multiobjective evolutionary algorithm for solving truck and trailer vehicle routing problems. *Eur. J. Oper. Res.*, 172(3):855–885, 2006.

[230] K. C. Tan, C. K. Goh, Y. J. Yang, and T. H. Lee. Evolving better population distribution and exploration in evolutionary multi-objective optimization. *Eur. J. Oper. Res.*, 171(2):463–495, 2006.

[231] K. C. Tan, C. Y. Cheong, and C. K. Goh. Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *Eur. J. Oper. Res.*, 177(2):813–839, 2007.

[232] A. Toffolo and E. Benini. Genetic diversity as an objective in multi-objective evolutionary algorithms. *Evol. Comput.*, 11(2):151–167, 2003.

[233] P. Toth and D. Vigo. An exact algorithm for the vehicle routing problem with backhauls. *Transport. Sci.*, 31(4):372–385, 1997.

[234] P. Toth and D. Vigo. Exact solution of the vehicle routing problem. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistic*, pages 1–31. Kluwer Academic Publisher, 1998.

[235] P. Toth and D. Vigo, editors. *The vehicle routing problem*. SIAM, 2001.

[236] P. Toth and D. Vigo. An overview of vehicle routing problems. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 1–26. SIAM, 2001.

[237] P. Toth and D. Vigo. VRP with backhauls. In P. Toth and D. Vigo, editors, *The vehicle routing problem*, pages 195–224. SIAM, 2001.

[238] P. Toth and D. Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Appl. Math.*, 123(1):487–512, 2002.

[239] P. Toth and D. Vigo. The granular tabu search and its application to the vehicle routing problem. *INFORMS J. Comput.*, 15(4):333–346, 2003.

[240] B. Ulungu, J. Teghem, P. Fortemps, and D. Tuyttens. Mosa method: a tool for solving multi-objective combinatorial optimization problems. *J. Multi-Criteria Decis. Anal.*, 8:221–236, 1999.

[241] P. J. M. van Laarhoven and E. H. L. Aarts, editors. *Simulated annealing: theory and applications.* Kluwer Academic Publishers, 1987.

[242] D. A. Van Veldhuizen. *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations.* PhD thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, USA, 1999.

[243] N. Velasco, P. Castagliola, P. Dejax, C. Guéret, and C. Prins. A memetic algorithm for a pick-up and delivery problem by helicopter. In F. B. Pereira and J. Tavares, editors, *Bio-inspired Algorithms for the Vehicle Routing Problem*, pages 173–190. Springer, 2009.

[244] R. M. Vicente Campos, Manuel Laguna. Context-independent scatter and tabu search for permutation problems. *INFORMS J. Comput.*, 17(1):111–122, 2005.

[245] T. Volgenant and R. Jonker. The symmetric traveling salesman problem and edge exchanges in minimal 1-trees. *Eur. J. Oper. Res.*, 12(4):394–403, 1983.

[246] C. Voudouris and E. Tsang. Guided local search and its application to the traveling salesman problem. *Eur. J. Oper. Res.*, 113(2):469–499, 1999.

[247] R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *J. Assoc. Comput. Mach.*, 21(1):168–173, 1974.

[248] S. Watanabe and K. Sakakibara. A multiobjectivization approach for vehicle routing problems. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *4th International Conference on Evolutionary Multi-criterion Optimization*, volume 4403, pages 660–672. Springer, 2007.

[249] L. D. Whitley. Permutations. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Evolutionary Computation 1: Basic Algorithms and Operators.* Institute of Physics Publishing, 2000.

[250] L. D. Whitley, T. Starkweather, and D. Fuquay. Scheduling problems and traveling salesmen: The genetic edge recombination operator. In J. D. Schaffer, editor, *3rd International Conference on Genetic Algorithms*, pages 133–140.

Morgan Kaufmann, 1989.

[251] S. Wolf and P. Merz. Evolutionary local search for the super-peer selection problem and the p-hub median problem. In *4th International Conference on Hybrid Metaheuristics*, pages 1–15. Springer, 2007.

[252] A. Wren. *Computers in transport planning and operation*. Ian Allen, 1971.

[253] A. Wren and A. Holliday. Computer scheduling of vehicles from one or more depots to a number of delivery points. *Oper. Res. Quart.*, 23(3):333–344, 1972.

[254] T. Yu and L. Davis. An introduction to evolutionary computation in practice. In T. Yu, L. Davis, C. M. Baydar, and R. Roy, editors, *Evolutionary Computation in Practice*, volume 88 of *SCI*, pages 1–8. Springer, 2008.

[255] T. Yu, L. Davis, C. M. Baydar, and R. Roy, editors. *Evolutionary Computation in Practice*, volume 88 of *SCI*. Springer, 2008.

[256] K. Q. Zhu. A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows. In *15th IEEE International Conference on Tools with Artificial Intelligence*, pages 176–183. IEEE Computer Society, 2003.

[257] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, ETH, Zurich, Switzerland, 1999.

[258] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms – a comparative case study. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *5th International Conference on Parallel Problem Solving from Nature V*, volume 1498 of *LNCS*, pages 292–304. Springer, 1998.

[259] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the Strenght Pareto approach. *IEEE T. Evolut. Comput.*, 3(4):257–271, 1999.

[260] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.*, 8(2):173–195, 2000.

[261] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report TIK 103, Computer Engineering and Networks Laboratory (TIK), ETH, Zurich, Switzerland, 2001.

[262] E. Zitzler, M. Laumanns, and L. Thiele. SPEA 2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization. In K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, and T. Fogarty, editors, *Evolutionary Methods for Design, Optimisation and Control*, pages 19–26. CIMNE, 2002.

[263] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assesment of multiobjective optimizers: An analysis and review. *IEEE T. Evolut. Comput.*, 7(2):117–132, 2003.

[264] E. Zitzler, M. Laumanns, and S. Bleuler. A tutorial on evolutionary multiobjective optimization. In X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'kindt, editors, *Metaheuristics for Multiobjective Optimisation*, volume 535 of *LNEMS*, pages 3–38. Springer, 2004.

[265] E. Zitzler, J. D. Knowles, and L. Thiele. Quality assessment of pareto set approximations. In J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *LNCS*, pages 373–404. Springer, 2008.