

People's Democratic Republic of Algeria
الجمهورية الجزائرية الديمقراطية الشعبية
Ministry of Higher Education and Scientific Research
وزارة التعليم العالي و البحث العلمي



University Mohamed El Bachir El Ibrahimi of
Bordj Bou Arreridj
Mathematics and Computer Science Faculty



Thesis

Presented for obtaining the rank of **DOCTOR**

In : MATHEMATICS

Specialty : Operational Research

Presented by : Hemic Meriem

Theme:

Multi-objective Optimization for Supply Chain Management

*Defended publicly on **February 22, 2024** before the jury composed of*

PRESIDENT :	RAHMOUNE Azedine	Prof	University of Bordj Bou Arreridj
RAPPORTEUR :	ZOUACHE Djaafar	Prof	University of Bordj Bou Arreridj
CO-RAPPORTEUR :	BRAHMI Boualem	M.C.A	University of Bordj Bou Arreridj
EXAMINATORS :	MOUSSAOUI Abdelouahab	Prof	University of Ferhat Abbas Setif-1
	BRAHIMI Belkacem	M.C.A	University Mohamed Boudiaf of M'Sila
	ADDOUNE Smail	M.C.A	University of Bordj Bou Arreridj



Acknowledgements

I thank "Allah" for having given us the strength, the courage and the patience for the elaboration of this modest work.

It will be very difficult for me to thank everyone because it is thanks to the help of so many people that I was able to complete this thesis.

*I would like to express my sincere thankfulness and gratitude to my supervisor, **Dr. Djaafar Zouache**, for his dedication, patience, advice, recommendations, encouragement, and his constant help throughout this work.*

*I warmly thank my co-supervisor **Dr. Brahmi Boualem** for his wise advice, multiple encouragements, and his help which were preponderant for the success of this thesis.*

*I would like to thank all the members of the jury : **Pr. RAHMOUNE Azedine, Pr. MOUSSAOUI Abdelouahab, Pr. DRIAS Habiba, Dr. BRAHIMI Belkacem, and Dr. ADDOUNE Smail** for accepting to evaluate and examine this thesis.*

*I also thank all the people who contributed to improve the quality of the written manuscript throughout this thesis project, and to mitigate the imperfections to an acceptable rate. For this painstaking work, I particularly thank **Dr. Omar Mostefaoui** and **Pr. Hima Abdelkader** for their collaboration.*

I would like to thank all my family, my parents, my brothers and sisters for their great moral support, their encouragement, their patience and their trust.

Contents

1	General Introduction	7
1.1	Context	7
1.2	Plan of the thesis	8
1.3	Contributions	10
1.4	Publications	10
1.4.1	International journals	10
1.4.2	International Conferences	10
I	Preliminary notions	12
2	Background	13
2.1	Introduction	13
2.2	Combinatorial Optimization	14
2.2.1	Basic concepts	14
2.2.2	Complexity	14
2.2.2.1	Space complexity	14
2.2.2.2	Time complexity	14
2.2.2.3	Asymptotic notations	15
2.2.2.4	Classes of complexity	15
2.2.3	Methodologies for COP	16
2.2.3.1	Exact methods	16
2.2.3.2	Heuristics	17
2.2.3.3	Metaheuristics	17
2.3	Multi-Objectives Optimization	23
2.3.1	Basic concepts	23
2.3.2	Pareto Efficiency	23
2.3.3	Quality Metrics	26
2.3.4	Methodologies for Multi-objectives optimization problems	27
2.3.4.1	Multi-Objective Evolutionary Algorithms (MOEAs)	28
2.4	Conclusion	38
3	Supply Chain Management	39
3.1	Introduction	39
3.2	Basic concepts	39
3.2.1	Logistics	39
3.2.2	Supply chain (SC)	40
3.2.3	Supply chain management (SCM)	40
3.2.3.1	History	40

3.2.3.2	Example of Supply chain management	41
3.2.3.3	Definition of SCM	41
3.3	Decision phases in SCM	42
3.3.1	Supply Chain Strategy	42
3.3.2	Supply Chain Planning	42
3.3.3	Supply Chain Operations	42
3.4	Key Issues in SCM	43
3.4.1	Globalization	43
3.4.2	Fast-changing Markets	43
3.4.3	Quality and compliance	43
3.5	Drivers of SCM	43
3.5.1	Logistical drivers of SCM	44
3.5.1.1	Location/ Facilities	44
3.5.1.2	Inventory	44
3.5.1.3	Transportation	44
3.5.2	Cross-Functional drivers of SCM	44
3.5.2.1	Information	44
3.5.2.2	Sourcing	44
3.5.2.3	Pricing	44
3.6	Classic SCM problems	44
3.6.1	The Facility Location Problem (FLP)	44
3.6.1.1	Problem Description	45
3.6.1.2	Mathematical formulation	45
3.6.1.3	Classification and extension of FLP	45
3.6.2	The Vehicle Routing Problem (VRP)	46
3.6.2.1	Mathematical formulation	46
3.6.2.2	Classification and extension of VRP	47
3.6.3	The Inventory Problem (IP)	48
3.6.3.1	Problem Description	48
3.6.4	Integration problems	49
3.6.4.1	The Location Routing Problem (LRP)	49
3.6.4.2	The Inventory Routing Problem (IRP)	50
3.7	Financial supply chain management(FSCM)	50
3.7.1	Definition	50
3.8	Green supply chain management (GSCM)	50
3.8.1	Environment	50
3.8.2	Definition	51
3.9	Emergency Medical Services (EMS)	51
3.9.1	Definition	51
3.9.2	Planning problems	51
3.9.2.1	Ambulance Location Problem (ALP)	51
3.9.2.2	Ambulance Dispatching Problem (ADP)	51
3.9.2.3	Ambulance Relocation Problem (ARP)	52
3.10	Conclusion	52

II	Research works (Contributions)	53
4	G-MOEA/D-SA algorithm for the ambulance dispatching and relocation problem during COVID-19.	54
4.1	Introduction	54
4.2	Literature review	54
4.3	Mathematical model	55
4.3.1	Decision variable	56
4.3.2	Constraints	56
4.3.3	Objective functions	57
4.4	The proposed algorithm	57
4.4.1	G-MOEA/D-SA for ADRP	57
4.4.2	Outlines of G-MOEA/D-SA algorithm	57
4.4.3	Complexity of G-MOEA/D-SA	59
4.5	Experimental results	59
4.5.1	Dataset Description	59
4.5.2	Experimental Results and Analysis	62
4.5.2.1	The effect of simulated annealing (SA):	62
4.5.2.2	Comparisons with other multiobjective algorithms	66
4.5.2.3	Comparisons on performance computational time	73
4.6	Conclusion	78
5	VRPTW-MOEA algorithm for the Vehicle Routing Problem with Time Windows	80
5.1	Introduction	80
5.2	Literature review	80
5.3	Mathematical model	81
5.3.1	Parameters	81
5.3.2	Decision variable	82
5.3.3	Constraints	82
5.3.4	Objective functions	82
5.4	The proposed algorithm	83
5.4.1	Motivation	83
5.4.2	Outlines of VRPTW-MOEA algorithm	83
5.4.2.1	Solution representation	84
5.4.2.2	Selection operator	85
5.4.2.3	Multi-type Crossover operators	85
5.4.2.4	Mutation operator	86
5.4.2.5	Local search	87
5.4.2.6	Updating the parent population	88
5.5	Experimental study	89
5.5.1	Test problems	89
5.5.2	Parameter settings	89
5.5.3	Comparison with its variants	89
5.5.4	Comparison with previous multiobjective algorithms	91
5.6	Conclusion	91

6	EAG-NSGA-II algorithm for Multi-Depot Green Vehicle Routing Problem	93
6.1	Introduction	93
6.2	Literature review	93
6.3	Mathematical model	94
6.3.1	Decision variable	94
6.3.2	Constraints	94
6.3.3	Objective functions	95
6.4	The proposed algorithm	95
6.4.1	Outlines of EAG-NSGA-II algorithm	95
6.4.2	Solution representation	96
6.4.3	Crossover operator	97
6.4.4	Local search (LS)	97
6.4.5	Updating the archive	97
6.5	Experimental results and discussion	97
6.6	Conclusions	101
7	General conclusion	102

List of Figures

2.1	Decision space and objective space for MOP.	23
2.2	The dominance regions in bi-objective spaces that result from various dominance relations.	25
2.3	Most common forms of Pareto front in the case of two objectives.	26
2.4	A general MOEA framework.	29
2.5	Description of methods without archive and methods with archive.	30
2.6	Hypergrid to maintain diversity in the archive.	31
2.7	The steps of clustering methods in the optimization process.	32
2.8	Principle of crowding.	32
2.9	Diagram that shows the way in which the NSGA-II works.	34
3.1	An example of supply chain management-Coca-Cola.	41
4.1	A graphical summary of all emergency Covid-19 call locations (circles).	59
4.2	All cases of randomization of incoming COVID-19 calls over 144 periods.	61
4.3	Computational time for Datasets1.	74
4.4	Computational time for Datasets2.	74
4.5	Computational time for Datasets3.	75
4.6	Computational time for Datasets4.	75
4.7	Computational time for Datasets5.	76
4.8	Computational time for Datasets6.	76
4.9	Computational time for Datasets7.	77
4.10	Computational time for Datasets8.	77
4.11	Computational time for Datasets9.	78
4.12	Computational time for Datasets10.	78
5.1	An example of swap mutation operator.	87
5.2	An example of the reinsert route (RR).	88
5.3	An example of the reinsert customers (RC).	88
6.1	Pareto set obtained by the multi-objective algorithms on some instances.	98

List of Algorithms

1	Pseudo code of Simulated Annealing	18
2	Pseudo code of Tabu Search	19
3	Pseudo code of the ϵ -MOEA algorithm	35
4	Pseudo code of the MOEA/D algorithm	37
5	Pseudocode of G-MOEA/D-SA	58
6	Pseudocode of VRPTW-MOEA	84
7	Pseudocode of multi-crossover operators	85
8	Pseudocode of relocation mutation	86
9	Pseudocode of EAG-NSGA-II algorithm for MDGVRP	96

1

General Introduction

1.1 Context

THE combinatorial optimization is a subfield of mathematical optimization that is connected to operations research, algorithm theory, and computational complexity theory. It has major applications in a variety of domains, including artificial intelligence, machine learning, mathematics, auction theory, and software engineering.

In most of the real-world problems, it is not a matter of optimizing only one objective but rather of simultaneously optimizing several objectives, which are generally conflicting. In the last decades, researchers showed an increasing interest in multiobjective optimization, which consists of optimizing several objectives simultaneously while looking for the best possible compromise, knowing that the improvement of one objective leads to the deterioration of another one. The notion of a unique optimal solution in mono-objective optimization disappears for multi-objective optimization problems (MOPs) in favor of the notion of a set of Pareto optimal solutions. The majority of these problems are qualified as difficult, especially in the continuous case, because their solution requires the use of advanced algorithms. We distinguish two main classes of methods: mathematical methods and methods based on meta-heuristics. Among the meta-heuristics, we find the family of evolutionary algorithms which have been very successful in multi-objective optimization. Over the decades, the economic market has undergone significant changes, which has necessitated a permanent revision of the adopted management methods to enable companies to adapt to these considerable changes. When supply was lower than demand, the main objective of companies was to produce in mass to reduce the unit production cost. With the increase in competition and the change in the supply-demand relationship, the importance of cost, quality, innovation, and delivery time has been realized. Nowadays, companies are increasingly focusing on their core business while delegating part of their work process to specialized service providers. As a result, supply chains have been created and we can now talk about supply chain management. Thus, competition is no longer between individual companies but between supply chain management in the current economic context. The term Supply Chain Management (SCM) first appeared in the 1980s and came into its own in the 1990s. The rise of the concept of SCM is mainly due to the fact that industrial and commercial companies want to respond in near-real-time to the demands

of their customers, while at the same time maintaining their place in the strategic arena. SCM has become a strategic axis for companies, especially in large multi-site and multi-national companies. The supply chain function has evolved from independent logistics, mainly synonymous with transport and warehousing management, to a global supply chain linking all flow from customer to customer to supplier to supplier. Thanks to the introduction of Electronic Data Interchange (EDI) techniques, and Enterprise Resource Planning (ERP) software, the SCM mode is moving from management by function to integrated management.

In the scientific works, throughout the last two decades, the various aspects of SCM are studied to face the new stakes, whether they are related to the social context or to the new strategies of the companies, as for example, the management of the risks in the supply chain, the tools of piloting of the supply chain, the inter-organizational collaboration of the supply chain, or the reverse supply chain, financial supply chain, and the green supply chain, etc.

There is a significant gap in SCM, particularly in healthcare, where emergency medical service (EMS) is one of the most researched sectors of healthcare where Operations Research (OR) is used. EMS are emergency medical services that treat illnesses and injuries which require an immediate medical response, treat them outside of the hospital, and transfer them to hospice care. Because ambulance response times can be critical in patient survival, ambulances are expected to reach as soon as possible at the location of reported occurrences, presenting fundamental considerations about where ambulances should be placed and how many should be dispatched.

Multi-objective supply chain management (MOSCM) is a complex and challenging discipline that involves optimizing multiple conflicting objectives simultaneously in the context of supply chain operations. Traditionally, SCM focused primarily on cost reduction and operational efficiency. However, as businesses have recognized the importance of other factors such as customer satisfaction, sustainability, and risk management, the need for a multi-objective approach has emerged.

In a multi-objective SCM problem, decision-makers must consider a range of objectives that may include minimizing costs, maximizing customer service levels, reducing lead times, optimizing inventory levels, improving sustainability, and enhancing supply chain resilience. These objectives often compete with each other, making it difficult to achieve optimal performance in all areas simultaneously. Balancing these objectives requires careful analysis and trade-off considerations.

Several methods and techniques can be applied to solve multi-objective SCM problems, and the choice of method depends on the complexity of the problem, the number of objectives, and the preferences of the decision-makers.

This thesis is devoted to the design of multi-objective optimization methods, more precisely, multi-objective evolutionary algorithms to handle with some variants of supply chain management problems. In the following sections, we present the organization of this document and the main contributions developed in this thesis.

1.2 Plan of the thesis

In this thesis, we are interested in solving three variants of a supply chain management problem using multiobjective evolutionary algorithms (MOEAs). The first variant studied is known as the ambulance dispatching and relocation problems (ADRP), with aim to ob-

tain optimal routes for ambulances to cover all emergency COVID-19 calls. The second is known as vehicle routing problem with time windows (VRPTW), with the aim to find the best routes for the vehicles to service all customers. The last variant is related to the multi-depot green vehicle routing problem (MDGVRP).

This Thesis is developed in two main parts with five chapters in total. The first part is devoted to the preliminary notions defining the context of the research work developed in this Thesis. This part essentially contains two main headings:

1. The second chapter is devoted to preliminary notions on combinatorial optimization problems in general, and their complexities and methodologies. In addition, this chapter addresses multi-objective optimization problems, and a list of multi-objective evolutionary algorithms known in the literature has been presented with particular emphasis on the methods used in our research work.
2. In the third chapter, we present the basic notions of the supply chain management problem in general. We will focus on the three problems of supply chain management (Facility Location Problem (FLP), Vehicle Routing Problem (VRP), Inventory Problem (IP), emergency medical services (EMS)) by giving a detailed definition with a mathematical formulation and by explaining the most known variants of each problem. We close this chapter with an overview of the different aspects of supply chain management.

The second part of this thesis contains the research works elaborated throughout this thesis project:

1. The global outbreak of the COVID-19 pandemic has had a significant impact, leading to a surge in emergency calls and posing significant challenges for emergency medical service centers (EMS) worldwide. This situation is particularly relevant in countries like Saudi Arabia, which experiences a large influx of pilgrims during pilgrimage seasons. Among the various challenges faced, the focus of the fourth chapter is on addressing the real-time problems of ambulance dispatching and relocation (real-time ADRP). To tackle this issue, we propose an enhanced MOEA/D algorithm called G-MOEA/D-SA, which incorporates Simulated Annealing. To validate the effectiveness of our approach, we conduct several experiments using real data collected during the Covid-19 pandemic in Saudi Arabia. These experiments aim to showcase the capabilities and advantages of the proposed method in handling real-time ADRP scenarios.
2. The fifth chapter is reserved for our second contribution to the vehicle routing problem with time windows (VRPTW). A new evolutionary algorithm has been developed by combining the ϵ -Domination based Multi-Objective Evolutionary Algorithm (ϵ -MOEA) and local search heuristic. An experimental study using famous 56 Solomons data sets was also conducted to highlight the proposed method.
3. In order to integrate environmental and sustainability thinking into supply chain management, the sixth chapter presents our third contribution which consists in studying the the Multi-depot Green Vehicle Routing Problem (MDGVRP). A new NSGA-II algorithm, called EAG-NSGA-II, has also been proposed to solve the designed constrained multi-objective MDGVRP model. Several experiments are conducted on 11 commonly adopted Cordeau's benchmarks to compare the proposed algorithm with three relevant state-of-art algorithms. At the end of this chapter, we present a discussion of the experimental results of the proposed algorithm.

1.3 Contributions

Concerning our contributions presented in this thesis, as already mentioned above, we have devoted the second part to the research work carried out throughout this thesis project. These contributions can be put in the context of a study on the solution of three variants of the supply chain management problem using various approaches to multi-objective evolutionary algorithms. This work can be summarized as follows:

1. Present a new variant of MOEA/D algorithm, namely G-MOEA/D-SA to handle the ambulance dispatching and relocation problems (ADRP). This research work was enhanced by a publication in 2023 in the international journal: Applied Soft Computing (Hemici et al., 2023).
2. Integrating the ϵ -domination based multi-objective evolutionary algorithm (ϵ -MOEA) with local search and multi-type crossover operators to solve the multiobjective vehicle routing problems with time windows (MOVRPTW).
3. Propose a novel variant of NSGA-II dubbed EAG-NSGA-II to address the multi-depot green vehicle routing problem (MDGVRP). This research work has been enhanced by a publication in the Artificial Intelligence and its Applications book (Hemici et al., 2021).

1.4 Publications

1.4.1 International journals

1. Meriem Hemici, Djaafar Zouache, Boualem Brahmi, Adel Got, and Habiba Drias. "A decomposition-based multiobjective evolutionary algorithm using Simulated Annealing for the ambulance dispatching and relocation problem during COVID-19". Journal Applied soft computing (2023): 110282. DOI: 10.1016/j.asoc.2023.110282. Impact Factor (IF): 8.263.
2. Meriem Hemici, Laith Abualigah, Djaafar Zouache, Boualem Brahmi, and Saad Talal Saad Alharbi. "A Guided Epsilon-Dominance Multiobjective Evolutionary Algorithm using Local Search Heuristics for the Vehicle Routing Problem with Time Windows". New Generation Computing (Major Revisions). Impact Factor (IF): 1.180.

1.4.2 International Conferences

1. Meriem Hemici, Djaafar Zouache, and Boualem Brahmi. "NSGA-II algorithm for multi-objective capacitated vehicle routing problem". Tunisian operational research society and decision aid society (TDAS/ TORS'2019). July 8-10, 2019. Zarzis, Tunisia.
2. Meriem Hemici, Djaafar Zouache, and Boualem Brahmi. "Epsilon- Domination based Multi-Objective Evolutionary Algorithm for Multi-objective optimization of Vehicle Routing Problem with Time Windows". International Conference on Networking Telecommunications, Biomedical Engineering and Applications (ICNTBA'19). November 4 - 5, 2019. Boumerdes, Algeria.
3. Meriem Hemici, Djaafar Zouache, Boualem Brahmi, and Kaouther Hemici. "External Archive Guided Manta Ray Foraging Optimization Algorithm for Multi-objective

Portfolio Optimization Problem". National Conference on Applied Computing and Smart Technologies (ACST'21). July th 10th, 2021. ESI-SBA, Algeria.

4. Meriem Hemici, Djaafar Zouache, Boualem Brahmi, and Kaouther Hemici. "An external archive guided NSGA-II algorithm for Multi-Depot Green Vehicle Routing Problem". International Conference on Artificial Intelligence and its Applications (AIAP'22). January 24-26, 2022. El Oued, Algeria. https://doi.org/10.1007/978-3-030-96311-8_47.

Part I

Preliminary nations

2

Background

2.1 Introduction

It is believed in some research literature that discrete optimization is an integer programming combined with combinatorial optimization (which is composed of optimization issues dealing with graph structures). Despite the fact that all of these areas have highly interconnected research literature, it frequently entails finding how to optimally allocate resources utilized to solve mathematical issues.

A lot of combinatorial problems call for simultaneous optimization of many goals. The majority of goals are expressed as incomparable units and are in some way antagonistic to one another (e.g., one goal cannot be improved without weakening at least another one). These are known as multi-objective problems. Consider a transportation company that wants to reduce overall route time in order to improve customer service. Furthermore, the corporation wishes to minimize the number of vehicles used in order to cut operational expenses. Clearly, both goals are at odds because adding additional trucks decreases route time while increasing operational expenses. Additionally, this challenge's goals are specified in a variety of measurement units.

In mono-objective optimization, it is feasible to assess if one solution is better than the other for each given pair of solutions. As a result, usually a unique solution (i.e., the global optimum) is obtained. However, there is no straightforward mechanism for judging if one solution is better than the others in multi-objective optimization. The most commonly used method in multi-objective optimization to compare solutions is the Pareto dominance relation, which leads to a set of solutions with varied trade-offs between the objectives rather than a single solution. These are known as Pareto optimum solutions or non-dominated solutions.

In this chapter, a review of evolutionary multi-objective optimization is presented. First, we provide basic concepts and terminologies in multi-objective optimization. These concepts are used repeatedly in several other chapters. Second, we will synthesize the methods used to solve multi-objective optimization problems. In the literature, the methods used to solve these problems are divided into two classes: exact methods and methods that use meta-heuristics. The difference between the exact methods and the metaheuristic methods is that the former find in most of the time exact solutions for the problem but consume

a huge computational time, and it can only find a solution for a multi-objective optimization problem, which is the opposite for the solving methods that use metaheuristics essentially the evolutionary algorithms that find an approximate set of Pareto solutions with an acceptable time.

2.2 Combinatorial Optimization

2.2.1 Basic concepts

Combinatorial optimization is an area of applied mathematics and theoretical computer science that focuses on choosing the best object from a constrained list of options. In these situations, a thorough search is frequently not possible. It resolves a number of optimization problems where the best solution must be found and the collection of likely solutions is discrete or may be reduced to discrete. The traveling salesman problem and the least spanning tree problem are among the most common combinatorial optimization problems (COPs). (Grotschel & Lovász, 1995)

Definition 1 *A combinatorial optimization problem (COP) consists in finding the minimum s^* of an application F , most often with integer or real values, on a finite set S such that $F(s^*) = \underset{s \in S}{\text{Minimize}} (F(s))$. Where F is an application of S in R^n .*

Remark 1 *The same definition can be given using the maximum knowing that $\underset{s \in S}{\text{Maximize}} (F(s)) = -\underset{s \in S}{\text{Minimize}} (F(s))$.*

2.2.2 Complexity

For COP, there are several algorithms that solve this problem. To choose the best one, we have to compare and analyze the performance of each algorithm. When analyzing an algorithm, we mainly consider the study of complexity. The complexity of an algorithm is the study of the amount of resources (time or space) needed to execute this algorithm.

2.2.2.1 Space complexity

The process of creating a formula to gauge how much memory space an algorithm will need to run efficiently is known as space complexity. There are two parts to the memory space needed by an algorithm: the fixed part, which contains the instruction space (i.e., code, simple variables, fixed components, constants, etc.), and the variable part, which contains the space needed by component variables whose size is dependent on the problem instance.

2.2.2.2 Time complexity

The process of formulating the entire amount of time needed to perform this algorithm is the definition of time complexity. The implementation specifics and programming language will not affect this calculation.

2.2.2.3 Asymptotic notations

An algorithm's complexity is mathematically represented using asymptotic notation. They are applied to derive inferences regarding the effectiveness of algorithms. These notations enable us to make approximate but significant assumptions about the complexity of time and space. The time complexity of algorithms is mostly represented by the three asymptotic notations that follow.

1. **Big-Oh notation:** The upper bound (worst case) of an algorithm's temporal complexity is expressed using the Big-Oh notation. As a result, Big-Oh notation represents the worst possible algorithm time complexity.
2. **Big-Omega notation:** The bottom bound (best case) of an algorithm's temporal complexity is expressed using the Big-Omega notation. Therefore, Big-Omega notation represents the optimal case of algorithm time complexity.
3. **Big-Theta notation:** The average bound (average case) of an algorithm's temporal complexity is expressed using Big-Theta notation. As a result, Big-Theta notation represents the typical case of algorithm time complexity.

2.2.2.4 Classes of complexity

This classification includes the fundamental well-known classes:

- **P(polynomial time):** Class P contains all relatively easy problems, i.e. those for which efficient algorithms are known. We can build a deterministic machine (e.g. a Turing machine) whose execution time is of polynomial complexity.
- **NP(Non-deterministic Polynomial time):** The problems of the NP class are decision problems for which we can construct a non-deterministic Turing machine whose execution time is of polynomial complexity.

Unlike deterministic machines that execute a well-defined sequence of instructions, non-deterministic machines have the remarkable ability to always choose the best sequence of instructions that leads to the correct answer when it exists. This abstract concept is in fact the basis of the whole theory of NP-completeness.

Among the set of problems belonging to NP, there is a subset that contains the most difficult problems: they are called NP-complete problems.

- **NP-complete:** Any problem in NP can be turned (reduced) into an NP-complete problem in polynomial time, according to this property. In other words, an issue is NP-complete if and only if it can be reduced to all other NP-related problems. If we discover a polynomial algorithm for an NP-complete issue, we subsequently discover polynomials for every problem in the class NP.
- **NP-hard:** An issue is NP-hard if it is more difficult than an NP-complete issue, or if there is an NP-complete issue that can be reduced to this issue using the Turing algorithm. The class NP is not always present in NP-hard issues. Particularly, all optimization issues with NP-complete "decision" problems fall under the category of NP-hard problems.

2.2.3 Methodologies for COP

COP can be particularly difficult to solve since the number of alternative solutions might grow exponentially with the complexity of the challenge. The resolution methods have been divided into two classes based on the quality of the solutions and the computational time required to find an optimal solution to the optimization problem: exact methods and approach methods, which include a large number of heuristic and metaheuristic algorithms.

2.2.3.1 Exact methods

These methods are called so because of the exact (optimal) solutions they provide. They allow us to obtain an optimal solution of the instances of the solved problems. Their general principle consists of an intelligent enumeration, as efficiently as possible, of all the solutions of the problem to extract an optimal solution.

1. **Branch & Bound methods:** [(Lawler & Wood, 1966), (Boyd & Mattingley, 2007)]

The Branch & Bound method is a generic method for solving COPs, which appeared in the middle of the 20th century. It enumerates in an intelligent way the set of solutions. To do this, it decomposes the solution space into smaller and smaller subsets, a good part of which is eliminated by means of bounds. This type of enumeration can therefore provide an optimal solution in a reduced time compared to a complete enumeration.

For combinatorial optimization problems, several Branch & Bound methods can be invented. However, they will have three common components:

- A rule for separating the solution set.
- A function for evaluating the solution sets.
- An exploration strategy.

2. **Branch & Cut methods:** [(Mitchell, 2002), (Belenguer et al., 2011)]

The previous techniques are no longer useful when the number of constraints is significant. The Branch & Cu approaches entail translating the constraints of the problem in stages, starting with the Cut phase of the solution space, for which the integrity requirement on the integer variables has been relaxed. When trying to solve a Linear Integer Programming (LIP), an Linear Programming (LP) solver is used to try and discover the best possible integer solution while still adhering to the problem's constraints. If not, the cutting phase must be continued on these sub-problems after a decomposition phase (also known as a branch) of the problem into two sub-problems is required.

3. **Dynamic Programming:** [(Bellman, 1966)]

This method follows Bellman's principle, which claims that "every solution to the initial problem of size N contains the optimal solution to the sub-problem of size $N-1$ ". In practice, we begin by solving a family of issues of size 1, then move on to the next phase, which is to solve a family of problems of size 2. After a fixed number of steps, one arrives to the starting problem of size N . At each stage, intermediate states must be examined and correspond to a family of problems to be solved. The approach must have a minimal number of intermediate stages and steps in order to be practical. It is generally known that only a very limited number of COP instances (10 to 25 customers) can be solved via dynamic programming.

2.2.3.2 Heuristics

Heuristics for COP is an algorithm that aims to find a feasible solution, taking into account the objective function, but without guaranteeing optimality. Exact methods have an exponential complexity on these problems, and only heuristics can solve large cases. In practice, we find many problems that are poorly formulated, or have many constraints, or have unknown complexity. Even if the problem is easy, adding new constraints can make NP-hard. Under these conditions, it is better from the start to use a heuristic approach, much easier to modify. There is a very large number of heuristics that depend on the problem to be treated, and it is easy to invent some. However, the following main types can be distinguished and will be described in detail.

1. **Local Search (LS)**[(Pirlot, 1996), (Crama et al., 2005)]

Many COPs are solved using LS techniques. It begins with an initial solution and develops a series of cheaper answers through subsequent changes. When the current solution can no longer be improved upon, the process ends. Since the search may end even if the algorithm's best result is not ideal, LS algorithms are frequently incomplete algorithms.

2. **Nearest Neighbors (NN)**[(Peterson, 2009), (Kramer & Kramer, 2013)]

The NN algorithm assumes that related objects are close together. In other words, related things are located close to one another. The concept of similarity, also known as distance, proximity, or closeness, is captured by NN in a way that is similar to some elementary school mathematics that we may have studied, such as calculating the distance between points on a graph.

3. **Greedy**[(Jungnickel & Jungnickel, 1999), (Jungnickel & Jungnickel, 2013)]

A greedy algorithm is a technique for solving problems which selects the locally optimal solution at each step in the hopes of achieving the global optimum. The two main objectives of this optimization procedure are the largest sum and the shortest path. Because this algorithm bases its choices on local information, it does not necessarily result in the global optimal solutions.

2.2.3.3 Metaheuristics

Metaheuristics are a set of methods used in operations research and artificial intelligence to optimize a wide range of different problems, without requiring major changes in the used algorithm. In other words: Meta-heuristics are a form of stochastic optimization algorithms, hybridized with a local search. The term meta is thus taken in the sense that the algorithms can combine several heuristics.

1. **Simulated Annealing (AS)**

SA was invented by (Kirkpatrick et al., 1983). They were able to solve 5000-vertex traveling salesman situations nearly optimally. The Metropolis simulation method in statistical mechanics inspired them. The analogy is based on the process of "annealing metals" in metallurgy. A metal that is cooled too quickly has many microscopic flaws, which is the equivalent of a combinatorial optimization problem's local minimum. Slow cooling causes the atoms to rearrange, the flaws to disappear, and the metal to have an ordered structure, which is equivalent to the global minimum for COP. A system's energy is represented by a real T ; the temperature.

Algorithm 1 Pseudo code of Simulated Annealing**Input:** x : Initial solution. T : The initial temperature. α : The parameter value between 0 and 1. IT_{max} : Number of iterations.

```

1:  $t = 1$ .
2: while  $(T > 0) \wedge (t \leq IT_{max})$  do
3:   Generate a solution  $x' \in N(x)$ . //  $N(x)$  the neighborhood function.
4:   Calculate  $\Delta f = f(x') - f(x)$ 
5:   if  $\Delta f > 0$  then
6:      $x := x'$ .
7:   else
8:     Generate  $r := random(0, 1)$  uniform distribution in the range.
9:     if  $r < exp(-\frac{\Delta f}{T})$  then
10:       $x := x'$ .
11:    end if
12:  end if
13:   $T = \alpha * T$ 
14:   $t = t + 1$ 
15: end while

```

Output: x : Best solution.**2. Tabu Search (TS)**

TS was invented by (Glover, 1986). They are of more recent conception than annealing which have no stochastic character and seem to be better for the same execution time. The primary mechanics of TS are inspired by human memory. TS uses a memory method to explore the search space while avoiding local optima. Thus, TS uses the memory mechanism to learn from previously explored paths.

Algorithm 2 presents the TS Pseudo-code given a collection of feasible solutions and $N(x)$ defend neighbors for each feasible solution. The goal of TS is to keep in a tabu list L , of a certain length l , the recently visited solutions. Each time we choose a new solution, it is inserted in the tabu list. If the tabu list is too large, we remove the oldest solution and it will no longer be tabu.

Algorithm 2 Pseudo code of Tabu Search**Input:** x : Initial solution. $T = \emptyset$: Tabu list which is initially empty. IT_{max} : Number of iterations.

```

1:  $t = 1$ 
2: while  $t \leq IT_{max}$  do
3:   Generate a solution  $x' \in N(x)$ . //  $N(x)$  the neighborhood function.
4:   Calculate  $f(x')$ .
5:   if  $x' \notin T$  then
6:      $x := x'$ .
7:     if  $f(x') < f(x)$  then
8:        $x := x'$ .
9:        $f(x) = f(x')$ .
10:       $T = T \cup \{x\}$ 
11:    end if
12:  end if
13:  if List  $T$  is full then
14:    Remove the first solution from the list  $T$ .
15:  end if
16:   $t = t + 1$ 
17: end while

```

Output: x : Best solution.**3. Evolutionary Algorithms (EAs)**

EAs are a type of algorithm that is influenced by Darwinian evolution theory to address various challenges. This vast class of algorithms involves the generation of a population of potential solutions to a given optimization problem.

At each iteration, the algorithm selects the best solutions, then randomly modifies them, in order to build a new population for the next iteration. The algorithm ends according to a criterion related to the convergence of the population, or to the computation time. The efficiency of the algorithm lies in the fact that the slight random variations on the solutions, allow to discover sufficiently and often better solutions which will be selected for the continuation. The process is progressively guided by the discovery of more satisfactory solutions. This is in line with Darwin's ideas and the theory of evolution of natural species, which postulate a constant and progressive adaptation of a species to its environment.

Three major families of algorithms were historically independently created in the mid-1960s and the 1970s. In order to handle problems involving continuous optimization, Rechenberg first suggested the Evolutionary Strategy in 1965. The next year, Fogel, Owens, and Walsh proposed evolutionary programming as a technique for creating finite state automata using artificial intelligence. The first genetic algorithm for combinatorial optimization was subsequently proposed by Holland in 1975.

- **Genetic Algorithm (GA)**

The oldest evolutionary algorithm is called GA. It takes its cues from Darwin's theory of evolution. Nowadays, the distinction between evolutionary and genetic algorithms is fuzzy.

Sometimes, the terms "genetic algorithm" and "evolutionary algorithm" are used interchangeably. We attempt to model the evolution of a population using genetic algorithms. We begin with a population of N solutions to the problem and each one of these solutions is represented by a single person. The parent population is the group that was selected at random. The value of the cost function $f(x)$, where x is the solution that the individual represents, expresses how well a person has adapted to their surroundings. If the cost of the solution a person represents is lower, we say the person is better fitted to their environment. Individuals develop through the employment of the recombination operators crossover and mutation. At each generation, parents are chosen at random to produce new children. Crossover is the main operator used to create new children (or offspring) by combining pieces of the selected parents. Mutation, on the other hand, is less common than crossover. Mutation is used to diversify populations and prevent GA from becoming too complex.

Natural Selection:

In biology, natural selection is one of the mechanisms that cause the evolution of species. This mechanism is particularly important because it explains the adaptation of species to environments over generations. The theory of natural selection helps to explain and understand how the environment influences the evolution of species and populations by selecting the most adapted individuals and is therefore a fundamental aspect of the theory of evolution.

Crossover:

Crossover is a genetic operator used to create new offspring solutions by combining genetic material from two or more parent solutions. It is inspired by biological reproduction and aims to explore and exploit the search space efficiently. Crossover typically involves selecting specific segments or parts of the parent solutions and exchanging or recombining them to create new solutions with potentially beneficial characteristics inherited from the parents. The specific mechanism of crossover varies depending on the type of representation used for the solutions (e.g., binary, real-valued, permutation).

Here are some commonly used crossover operators in evolutionary algorithms:

- Single-Point Crossover:** [(Poli & Langdon, 1998), (Poli & Langdon, 1997)] In this operator, a single crossover point is randomly chosen along the chromosome (or string of genes), and the genetic material beyond that point is exchanged between two parent solutions to create two offspring solutions.
- Multi-Point Crossover:** [(De Jong & Spears, 1990), (Umbarkar & Sheth, 2015)] Similar to single-point crossover, but instead of a single crossover point, multiple crossover points are selected. Genetic material between these points is exchanged to create offspring solutions.
- Uniform Crossover:** [(Falkenauer, 1999), (Hu & Di Paolo, 2009), (Williams & Crossley, 1998)] This operator works at the gene level. Each gene of the offspring is selected from one of the parents with equal probability. It provides a more diverse exploration of the search space.
- Arithmetic Crossover:** [(Yalcinoz et al., 2001)] Applicable to real-valued rep-

representations, this operator performs a weighted average of the parent values at each gene to create the corresponding offspring value. The weights are typically randomly generated.

- (e). **Order Crossover (OX):**[(Davis, 1985)] Primarily used in permutation-based problems, OX selects a random segment from one parent and preserves the order of the genes within that segment while filling the remaining positions in the offspring with the genes from the other parent, respecting their order.
- (f). **Partially Mapped Crossover (PMX):**[(Ting et al., 2010)] Also used in permutation-based problems, PMX selects two crossover points and preserves the order of the genes within that segment in one offspring. The remaining genes are filled based on the mappings of genes between the parents.

Mutation:

Mutation is a genetic operator used to introduce diversity and explore new regions of the search space by making random changes to the genetic material (e.g., genes) of individual solutions (i.e., individuals) in the population.

Mutation is typically applied with a low probability to individual solutions selected from the current population. The mutation operation randomly alters the genetic material of the individual solution, resulting in a new solution in the search space. The specific mutation mechanism depends on the representation used for the solutions (e.g., binary, real-valued, permutation).[(Forrest, 1996), (Deep & Thakur, 2007), (Deep & Mebrahtu, 2011)]

Here are some commonly used mutation operators in evolutionary algorithms:

- (a). **Bit Flip Mutation:** In this operator, a random bit in the binary representation of the solution is flipped from 0 to 1 or from 1 to 0.
- (b). **Gaussian Mutation:** Applicable to real-valued representations, this operator adds a random value drawn from a Gaussian distribution with a mean of 0 and a standard deviation to each gene of the solution.
- (c). **Swap Mutation:** Primarily used in permutation-based problems, swap mutation selects two random positions in the solution and swaps the elements at those positions.
- (d). **Inversion Mutation:** Another permutation-based operator, inversion mutation selects a random segment of genes in the solution and reverses the order of the genes in that segment.
- (e). **Random Resetting:** In this operator, a gene is randomly assigned a new value from its allowable range.
- (f). **Boundary Mutation:** Applicable to real-valued representations, this operator perturbs the boundary of the search space by randomly selecting a dimension and setting the corresponding gene value to the boundary value.

4. Swarm Intelligence-Based Algorithms

- **Particle Swarm Optimization (PSO)**

PSO is inspired by the Social Behavior of Birds flocking, was proposed by (Kennedy & Eberhart, 1995). PSO is a computational method for problem optimization. By updating generations, PSO hunts for optima. The solution of the problem is represented by particles in PSO. Particles communicate with one another directly or indirectly by employing search directions (gradients). During the PSO

iteration, each particle updates its position based on its prior experience as well as the experience of its neighbors. Particles are made up of three vectors:

- (a). X-vector represents the particle's current position in the search space.
- (b). P-vector (Pbest) represents the position of the particle's best solution determined thus far.
- (c). V-vector contains a gradient (direction) in which a particle will travel if left alone.

- **Gray Wolf Optimization (GWO)**

GWO, proposed by (Mirjalili et al., 2014), is a metaheuristic inspired by the social structure and hunting behavior of grey wolves. It emulates the hierarchical leadership and search mechanism observed in grey wolf packs. The algorithm assigns three wolf types, namely alpha, beta, and gamma, to simulate the hierarchical structure. GWO strikes a balance between exploration and exploitation by exploring the search space for promising solutions and exploiting them to improve the overall solution quality. Each individual in the population represents a potential solution to the problem at hand. The fitness function is computed for all solutions, and the position update equations are utilized to adjust the position of each wolf.

$$\begin{aligned}\vec{D} &= |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \\ \vec{X}(t+1) &= \vec{X}_p(t) - \vec{A} \cdot \vec{D}\end{aligned}\quad (2.1)$$

Where t refers to the current iteration. \vec{X}_p is the prey position. \vec{X} is the grey wolf position. \vec{A} and \vec{D} are coefficient vectors, are defined as:

$$\begin{aligned}\vec{A} &= 2 \cdot \vec{a} \cdot \vec{r}_1 - \vec{a} \\ \vec{C} &= 2 \cdot \vec{r}_2\end{aligned}\quad (2.2)$$

Where a components of \vec{a} gradually decreases from 2 to 0 over the iteration period. Random values within the range of $[0, 1]$ are assigned to the variables r_1 and r_2 . When the entire pack converges on the prey, their positions are updated based on the best positions of the alpha, beta, and delta wolves using the following equations:

$$\begin{aligned}\vec{D}_\alpha &= |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}| \\ \vec{D}_\beta &= |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}| \\ \vec{D}_\gamma &= |\vec{C}_3 \cdot \vec{X}_\gamma - \vec{X}| \\ \vec{X}_1 &= \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha \\ \vec{X}_2 &= \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta \\ \vec{X}_3 &= \vec{X}_\gamma - \vec{A}_3 \cdot \vec{D}_\gamma \\ \vec{X}(t+1) &= \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3}\end{aligned}\quad (2.3)$$

2.3 Multi-Objectives Optimization

2.3.1 Basic concepts

A multi-objective optimization problem (MOP) involves simultaneously optimizing multiple objective functions that may be conflicting in nature. It can be formulated as follows:

$$\begin{aligned} \text{Minimize/Maximize } F(x) &= (f_1(x), \dots, f_m(x)) \\ \text{s.t. } x &\in S \end{aligned} \quad (2.4)$$

Here, m represents the total number of objectives, and f_i denotes the i -th objective. The decision vector is denoted by x , and S represents the set of feasible solutions that satisfy the constraints of the optimization problem. The mapping of the decision space is denoted as $F(x) \in R^m$ and referred to as the objective space. The elements of $F(x)$, given by $(f_1(x), f_2(x), \dots, f_m(x))$, are known as objective vectors and consist of real-valued objective functions. Based on these definitions, the following key definitions are provided:

Definition 2 (Decision space) The space R^n in which the set of feasible solutions $S \subseteq R^n$ is located is called the decision space.

Definition 3 (Objective space) The Space R^m in which the image of S in R^n by the application $F(S)$.

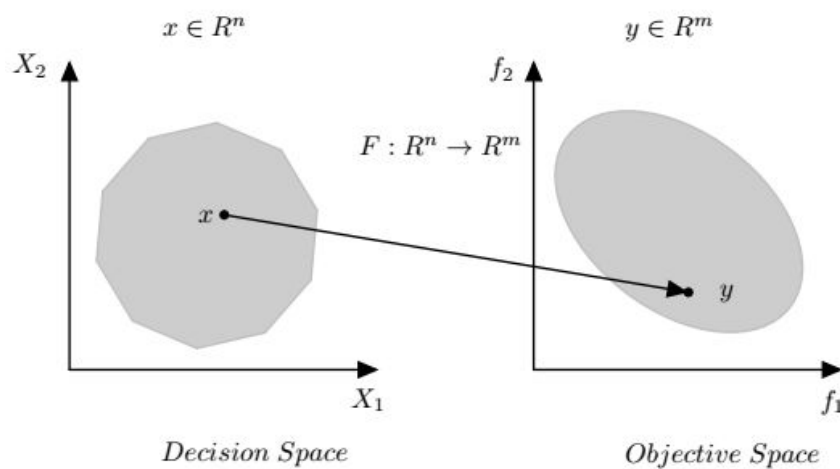


Figure 2.1 – Decision space and objective space for MOP.

2.3.2 Pareto Efficiency

In this part, we will introduce the notions related to dominance relation which was proposed by the Italian economist Pareto(1876) and which allows to compare two objective vectors for MOPs.

Definition 4 (Pareto-Dominance, denoted by \prec) A solution x_1 is said to dominate a solution x_2 , if and only if the following conditions are met:

$$\forall i \in \{1, \dots, m\}; f_i(x_1) \leq f_i(x_2) \quad \wedge \quad \exists j \in \{1, \dots, m\}; f_j(x_1) < f_j(x_2). \quad (2.5)$$

Definition 5 (ϵ -Dominance (Laumanns et al., 2002), denoted by \prec_ϵ) Given a relaxed vector $\epsilon = (\epsilon_1, \epsilon_2, \dots, \epsilon_m)$ with $\epsilon_i > 0, i = 1, 2, \dots, m$, a solution x_1 is said to ϵ -dominate a solution x_2 , if and only if the following conditions are met:

$$\forall i \in \{1, \dots, m\}; f_i(x_1) - \epsilon_i \leq f_i(x_2) \wedge \exists j \in \{1, \dots, m\}; f_j(x_1) - \epsilon_j < f_j(x_2). \quad (2.6)$$

Definition 6 (Grid-Dominance (Yang et al., 2013), denoted by \prec_G) A feasible solution x_1 is said to grid-dominate another feasible solution x_2 , if and only if:

$$\forall i \in \{1, \dots, m\}; G_i(x_1) \leq G_i(x_2) \wedge \exists j \in \{1, \dots, m\}; G_j(x_1) < G_j(x_2). \quad (2.7)$$

Where $G(\cdot)$ is the grid coordinates of the solution and can be calculated by:

$$G_i(x_1) = \lfloor \frac{(f_i(x_1) - lb_i)}{d_i} \rfloor \quad (2.8)$$

$$d_i = \frac{(ub_i - lb_i)}{div} \quad (2.9)$$

$$lb_i = \min f_i - \frac{(\max f_i - \min f_i)}{2 * div} \quad (2.10)$$

$$ub_i = \max f_i + \frac{(\max f_i - \min f_i)}{2 * div} \quad (2.11)$$

Where div denotes the number of divisions of the objective space in each dimension.

Definition 7 (θ -Dominance (Yuan et al., 2015), denoted by \prec_θ) A solution x_1 is said to θ -dominate a solution x_2 , if and only if the following conditions are met:

$$\forall i \in \{1, \dots, m\}; F_i(x_1) < F_i(x_2) \quad (2.12)$$

Where $F(\cdot)$ is the weight vector and can be calculated by:

$$\begin{aligned} F_i(x_1) &= d_i^1(x_1) + \theta d_i^2(x_1) \\ d_i^1(x_1) &= \frac{\|f(x_1)^T \cdot \lambda_i\|}{\|\lambda_i\|} \\ d_i^2(x_1) &= \|f(x_1) - d_i^1(x_1) \times (\frac{\lambda_i}{\|\lambda_i\|})\| \end{aligned} \quad (2.13)$$

Where λ is the weight vector and θ is a penalty parameter. $d_i^1(x_1)$ and $d_i^2(x_1)$ are computed in the normalized objective space.

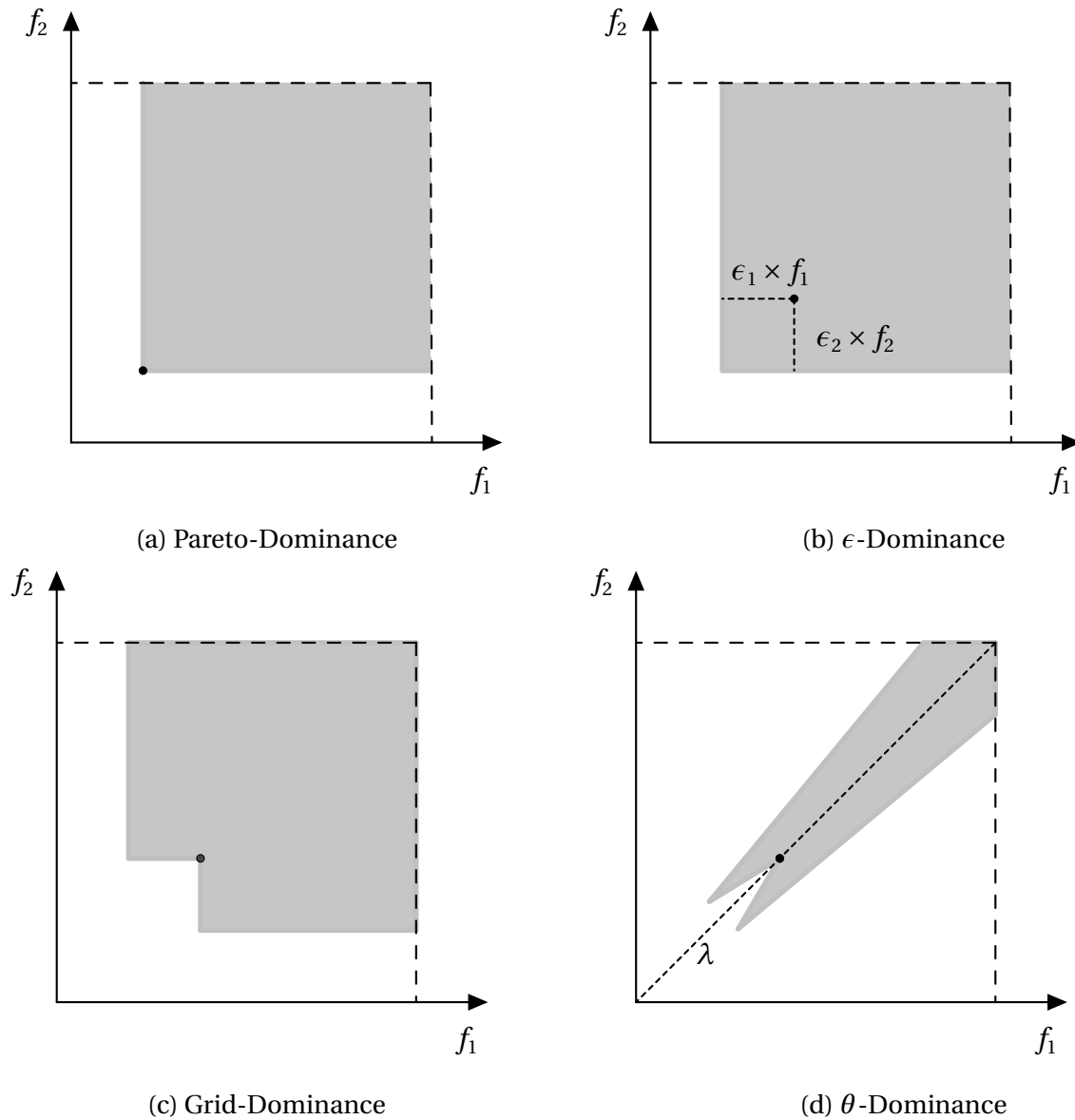


Figure 2.2 – The dominance regions in bi-objective spaces that result from various dominance relations.

Definition 8 (Pareto optimality) A given solution x_1 is referred to as a Pareto optimum solution if it is not dominated by any other solution in the whole feasible search space:

$$\nexists x_2 \in S \mid x_1 \prec x_2. \quad (2.14)$$

Definition 9 (Pareto optimal set) Pareto optimal solution set is the set of all Pareto optimal solutions, denoted as:

$$PS = \{x_1 \in S \mid \nexists x_2 \in S, x_1 \prec x_2\}. \quad (2.15)$$

Definition 10 (Pareto front) Pareto front (PF) is the objective space projection of the Pareto optimum set (PS):

$$PF = \{F(x) \mid x \in PS\}. \quad (2.16)$$

Remark 2 Depending on the type of problem we are trying to solve, we obtain the Pareto front. The most common forms of Pareto fronts are shown in figure 2.3.

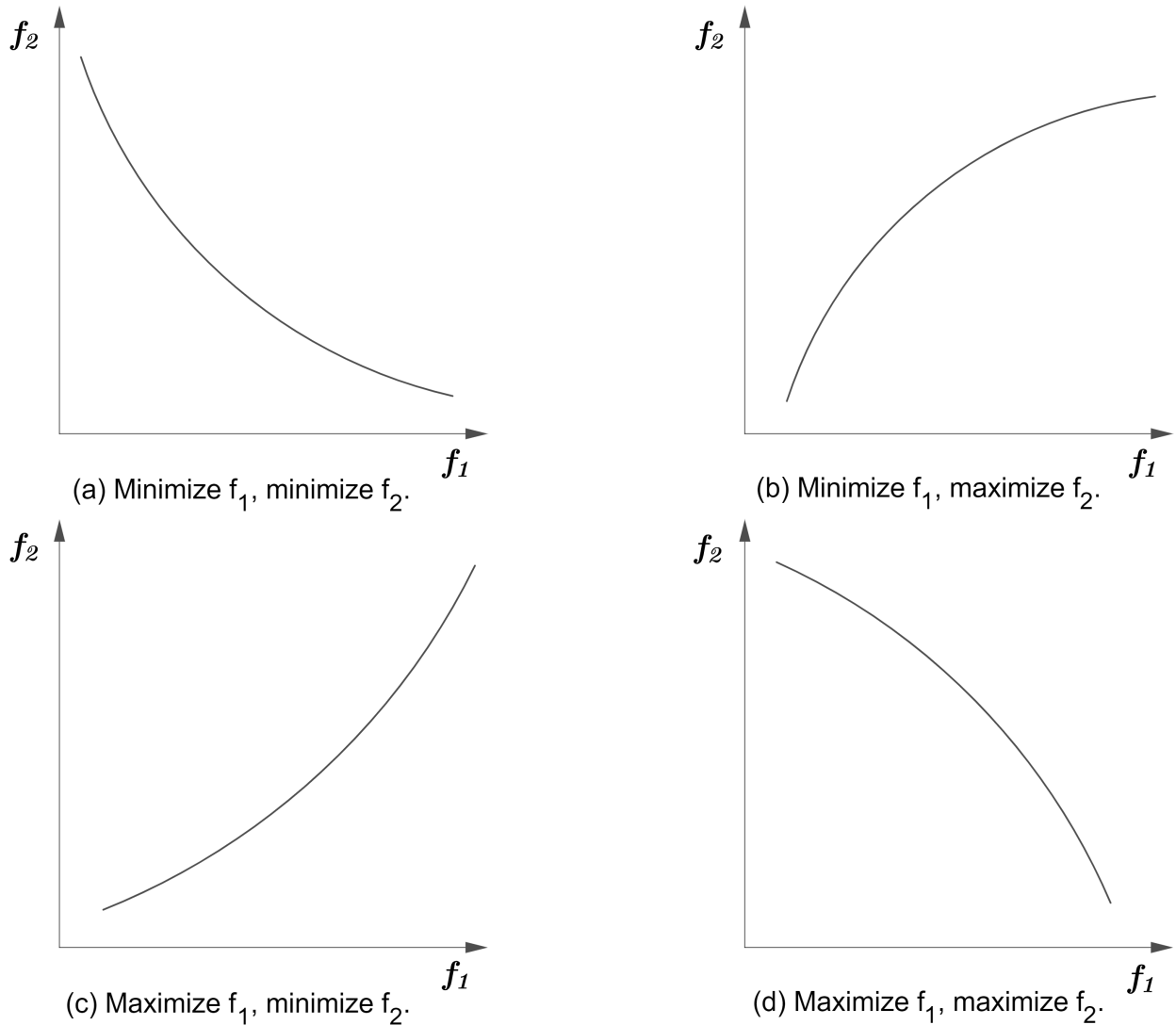


Figure 2.3 – Most common forms of Pareto front in the case of two objectives.

2.3.3 Quality Metrics

The evaluation of the performance of MOEAs can have several aspects, such as the quality of the results obtained or the computation time required. We only focus on the first aspect since the execution time of the algorithms becomes negligible when it comes to a real application where the evaluation of the objectives can last from a few minutes to a few days.

Definition 11 (*Set coverage metric*) This metric is intended to be used for comparing two non dominated Pareto sets gained by different algorithms (Zitzler & Thiele, 1998).

$$C(A, B) = \frac{|\{b \in B \mid \exists a \in A: a \text{ dominates } b\}|}{|B|} \quad (2.17)$$

The first and second algorithms' non-dominated solutions are divided into sets A and B , respectively. The equation $C(A, B) = 1$ shows that set A dominates all solutions in set B . No solution in set B is dominated by set A , as shown by the value $C(A, B) = 0$. Additionally, $1 - C(B, A)$ does not fundamentally equal $C(A, B)$.

Definition 12 (ϵ -indicator (I_ϵ)) Given two non dominated Pareto sets gained by different algorithms (Zitzler et al., 2003).

$$I_\epsilon(A, B) = \max_{b \in B} \min_{a \in A} \max_{1 \leq i \leq m} f_i(a) - f_i(b). \quad (2.18)$$

Definition 13 (**Inverted generation distance (IGD)**): Given a set of reference points R and a set of obtained solutions S , the IGD measures the quality of the solutions in S compared to the reference points in R (Sun et al., 2018). Let $d(x, R)$ represent the distance between a solution x and the set of reference points R . The IGD is calculated as the average of the distances between each solution in S and the nearest reference point in R :

$$IGD = \frac{\sum_{x \in S} dist(x, R)}{|S|} \quad (2.19)$$

Where $|S|$ is the cardinality of set S , and the sum is taken over all solutions x in S . The smaller the value of IGD, the better the quality of the solutions in S with respect to the reference points in R .

Definition 14 (**Hypervolume (HV)**) The HV is a mathematical formulation used to measure the quality of a set of solutions in multi-objective optimization. It quantifies the extent of the objective space covered by the solutions. The HV value represents the volume of the objective space dominated by the solutions with respect to a reference point (Zitzler et al., 2003).

Let R be the reference point, which is a point in the objective space that represents the worst possible values for each objective. The HV of a set of solutions S is calculated as the volume of the hypervolume dominated by S and bounded by the reference point R .

$$HV = Volume(\cup_{x \in S} \{[f_1(x), r_1] \times \dots [f_m(x), r_m]\}). \quad (2.20)$$

The HV can be calculated using various algorithms, such as incremental computation or Monte Carlo sampling. The higher the HV value, the better the coverage and diversity of the solutions in the objective space.

2.3.4 Methodologies for Multi-objectives optimization problems

The best feasible trade-offs between the objectives are represented by a group of solutions in MOPs, as opposed to a single solution that reduces or maximizes all objectives simultaneously. When solving a MOP, the goal is typically to discover not only this collection of tradeoff solutions, but also that these answers are evenly spread across the Pareto front.

EAs have long been renowned for their ability to solve MOPs by identifying a representative collection of Pareto optimal solutions in a single run. Furthermore, compared to traditional mathematical programming techniques, EAs are less sensitive to the shape or continuity of the Pareto front.

2.3.4.1 Multi-Objective Evolutionary Algorithms (MOEAs)

MOPs are a specific type of EAs called MOEAs. The structure of MOEAs and evolutionary algorithms for combinatorial optimization problems is similar. Due to the fact that MOEA uses m -dimensional fitness vectors ($m > 2$), the fitness assignment process is the main distinction. Finding a Pareto front approximation is, as various publications have noted, a bi-objective task in and of itself with the following goals:

- Reduce the distance between the generated vectors and the Pareto optimal front.
- Maximize the diversity of the found Pareto front approximation.

1. General framework

Figure 2.4 depicts the general framework of a MOEA. To begin, a population P is created by randomly producing N individuals. However, if we know the qualities of a good solution, we should use that knowledge to populate the population. Second, fitness assignment necessitates ranking individuals based on a preference connection and then providing a scalar fitness value to each individual based on this rank. Third, mating selection is used for reproduction, with individuals with higher quality becoming parents of the following generation to push for quality improvement. Fourth, to create children, variation operators (Crossover and Mutation) are used to these parents. Fifth, environmental selection affects whether solutions (i.e., the next-generation population) survive from the existing population and children. This evolutionary process continues until a terminating condition is met (for example, the number of generations surpasses a predetermined upper bound).

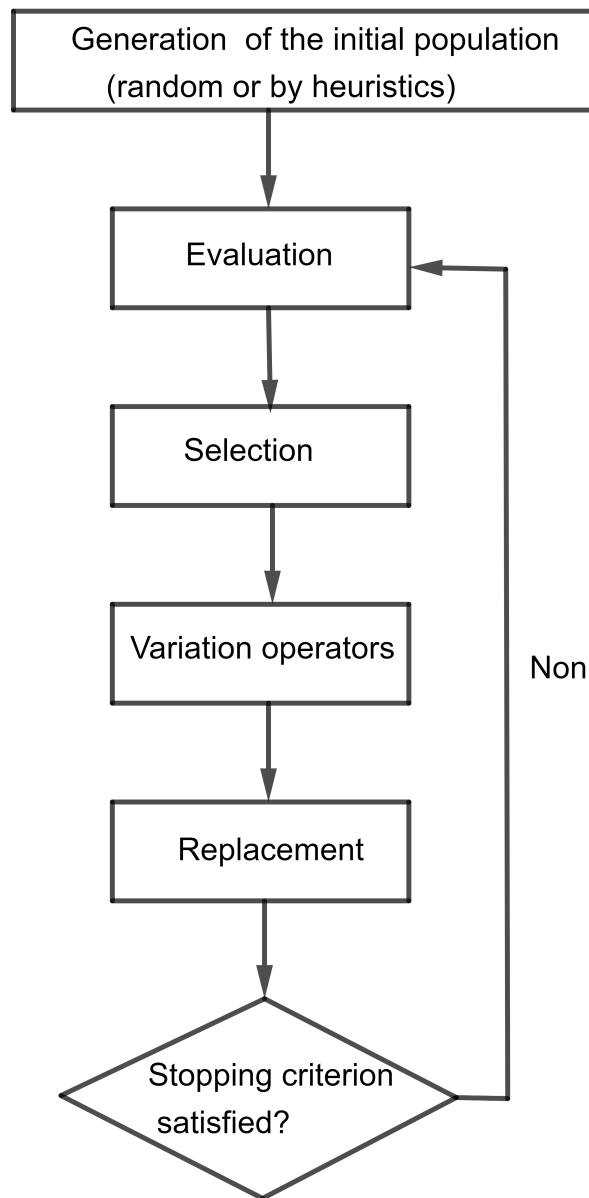


Figure 2.4 – A general MOEA framework.

2. Key Elements

The elements that must be considered when designing MOEAs are covered in detail in this part.

- (a). **Fitness Assignment:** A partial order of the population in the objective functional space is created using a preference relation. Then, based on how each solution stacks up against the others, a scalar score (rank) is given to it. For instance, dominance ranking systems keep track of how many people a certain person is controlled by. According to dominance counting methods, an individual's fitness is correlated with the number of other people it dominates. The most often used preference relation in MOEAs is Pareto dominance, as we saw in the preceding section.

Aggregation-based: This approach aggregates or combines the objective func-

tions into a single scalar value. Different components of the Pareto optimum set are produced by methodically changing the parameters during the optimization process. Although an aggregation-based strategy can be expressed as a preference relation, it should be highlighted that the solutions are not contrasted in the objective space. In other words, before the comparison, the vectors are converted from R^m to R .

Preference-based: This strategy involves inducing a partial order of the population in the objective space via a preference relation. Then, based on how each solution stacks up against the others, a scalar score (rank) is given to it. For instance, approaches based on dominance rank count the number of persons over which a certain individual is dominant. In dominance-based methods, a person's fitness is measured by how many other people they can dominate. The most popular preference relationship in MOEAs is Pareto dominance.

(b). **Elitism:**

Elitism is a method that prevents the loss of the best solutions obtained during the search owing to stochastic factors. This notion is very important in MOEAs. Elitism is more difficult to apply in multi-objective optimization than in single-objective optimization. Due to restricted memory resources, if more non-dominated solutions emerge than can be kept, which solutions should be discarded? As a result, the elitist method used influences whether or not the fitness is globally convergent. Currently, we can differentiate primarily two methods of implementing elitism. One approach is to merge the old and new populations and then use selection to maintain the best solutions in the next generation (Deb et al., 2002). The alternative approach is to keep up with an external set of individuals known as an archive that stores the non-dominated solutions found during the search process. These two approaches are illustrated in Figure 2.5.

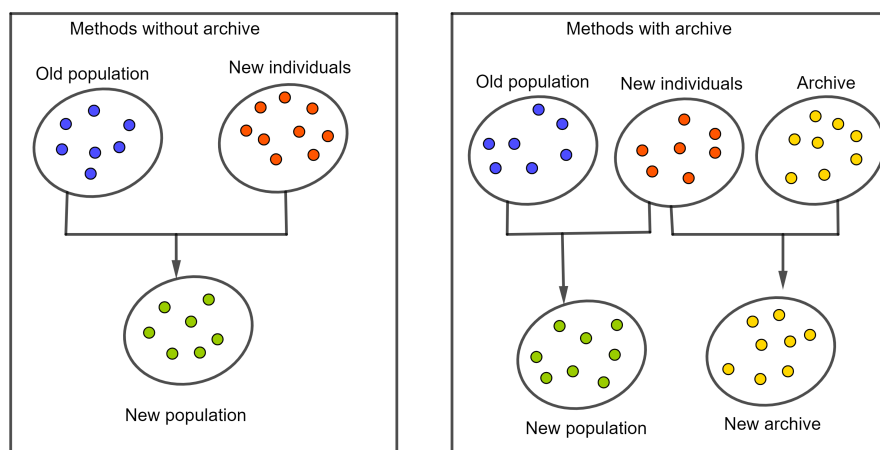


Figure 2.5 – Description of methods without archive and methods with archive.

(c). **Density Estimators**

MOEAs aim to produce a set of solutions that are evenly spread throughout the Pareto front and are not dominated. The options for preserving population variety will be discussed in the paragraphs that follow.

- (d). **Fitness sharing:** Creating and maintaining niche subpopulations (niches) distributed throughout the objective space is the aim of fitness sharing. The idea is to view fitness as a resource that needs to be shared by others who have similar interests. Due to the higher level of competitiveness in the specialty, each participant obtains less fitness points. Formally, the shared fitness f_{si} of the person i is defined as follows:

$$f_{si} = \frac{f_i}{\sum_{j=1}^N \phi(d_{ij})} \quad (2.21)$$

In this case, the fitness of individual i is f_i , and the partition function is $\phi(d_{ij})$, which is defined as follows:

$$\phi(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{share}}\right) & d_{ij} < \sigma_{share} \\ 0 & \text{Otherwise.} \end{cases} \quad (2.22)$$

Where σ_{share} represents the radius of the niche. The distance between people i and j is denoted by d_{ij} .

Hypergrid: A hypergrid creates hypercube-like divisions in the objective space. Figure 2.6 illustrates how each non-dominated solution takes up a hypercube. Only non-dominated solutions that belong to a sparsely populated hypercube are to be accepted. During the search process, the position and extension of the grid can be altered, even though the hypergrid's number of divisions in each dimension remains constant.

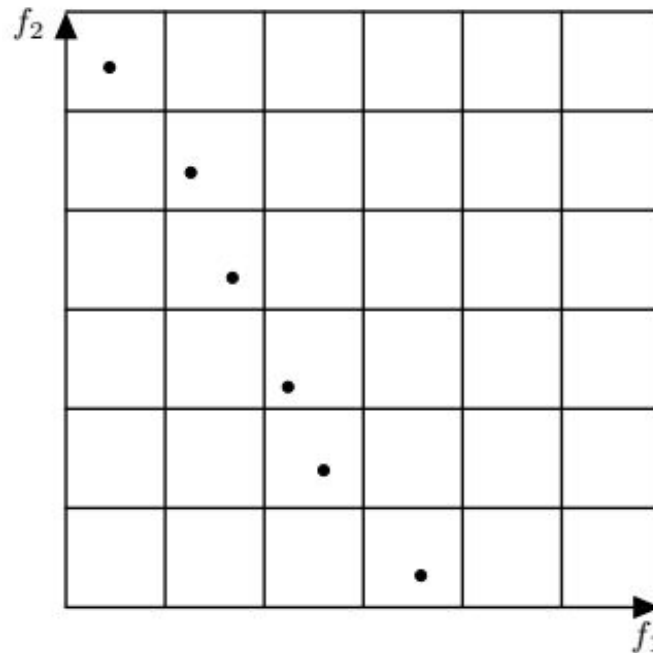


Figure 2.6 – Hypergrid to maintain diversity in the archive.

Clustering: The objective of a clustering algorithm is to group a set of points that are both quite distinct from one another and reasonably close to one another. Clustering is used in a MOEA to maintain the archive's diversity while reducing its size. Figure 2.7 depicts the three stages of this technique. The archive is divided into subsets using a clustering algorithm, and a representative is chosen from each subset before the remaining members of the cluster are discarded.

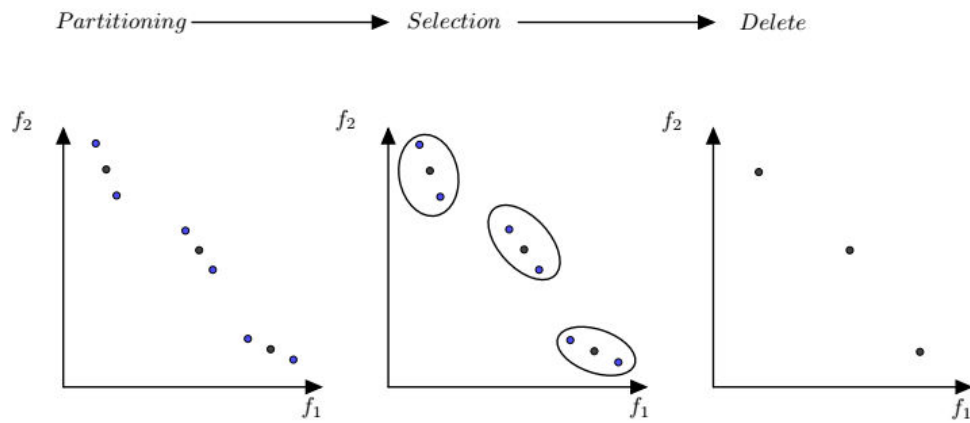


Figure 2.7 – The steps of clustering methods in the optimization process.

(e). **Crowding distance (CD)**

CD is a selection operator defined in the search space, used to estimate the density in the neighborhood of an individual i . It computes the average distance on each objective, between the two closest points located on both sides of the solution i . This distance noted $CD(i)$ serves as an estimator of the size of the largest hypercube including point i without including another point of the population and formed by the solutions of the same Pareto front closest to i (see Figure 2.8).

$CD(i)$ is computed as a function of the perimeter of the hypercube having as vertices the points closest to i on each objective. Figure 2.8 is represented the two-dimensional hypercube associated with point i .

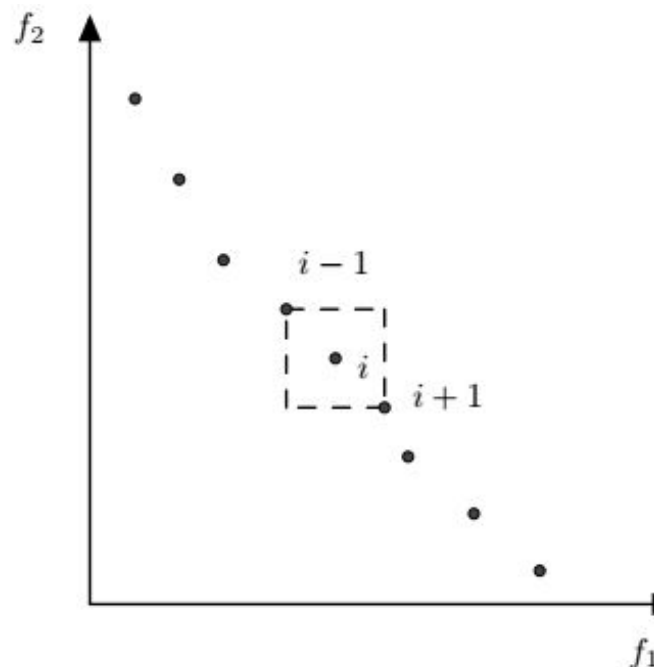


Figure 2.8 – Principle of crowding.

3. Classification

MOEAs have been used in a variety of ways to solve MOPs.

(a). Pareto-based Algorithms

The first class of MOEAs primarily uses the notion of Pareto dominance to compare individuals to each other. Furthermore, in the elitist replacement, these individuals are ranked using this dominance relationship and only the best are retained. In this part, we present algorithms that are based on the concept of Pareto dominance. Among them, we have selected the two algorithms that are considered, in the literature, the most efficient (NSGA-II and ϵ -MOEA). These approaches are relatively recent and still represent research topics.

Nondominated Sorting Genetic Algorithm II (NSGA-II):

NSGA-II was proposed by (Deb et al., 2002) and is classified as one of the benchmark algorithms in the field of multi-objective evolutionary optimization. It takes its name from the NSGA algorithm that was previously proposed by (Srinivas & Deb, 1994). In this second version of NSGA, the author tries to solve all the criticisms made on NSGA: complexity, non-elitism, and use of sharing. NSGA-II integrates a selection operator, based on a crowding distance calculation, very different from NSGA. NSGA-II is an elitist algorithm that does not save the elite in an external archive. To combat elitism, NSGA-II assures that the best individuals encountered are retained in each new generation. This new version of NSGA reduced the complexity of the algorithm for k objectives and N individuals to $O(kN^2)$, created an elitist method, and removed the sharing parameters. This makes this algorithm one of the most efficient algorithms for finding the Pareto optimal set with an excellent variety of solutions.

Each population member i has two attributes: non-domination rank $rank(i)$ and crowding distance $CD(i)$. To lead the selection process with the uniform distribution of Pareto solutions, a comparison operator defined in terms of these two properties is used.

Given two persons i and j , we say that i is superior to j if and only if:

$(rank(i) < rank(j)) \vee (rank(i) = rank(j) \wedge (CD(i) > CD(j)))$ With this relation, we favor the solution belonging to the lowest order Pareto front when comparing two non-dominated solutions belonging to two Pareto fronts. Otherwise, when two solutions belong to the same Pareto front (the last front to finish the parent population's size), we choose the option with the greater crowding distance. The points in the top half of the sorted list are chosen after the crowding sort of the final front points to finish the size N . By computing the perimeter of the cuboid or hypercube created by the closest neighbors of the objective space, one can determine $CD(i)$ of a point i .

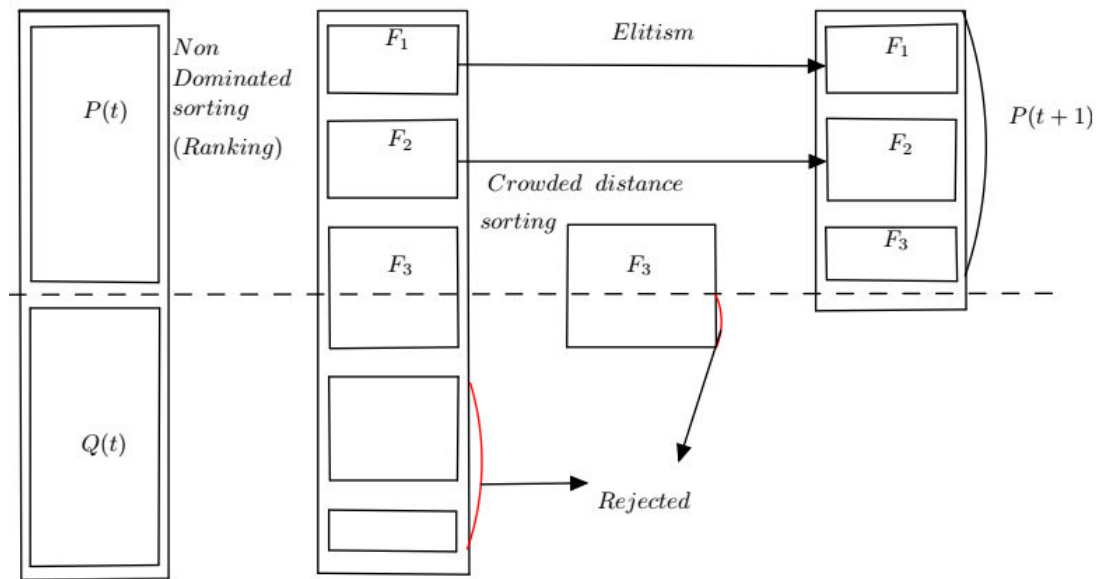


Figure 2.9 – Diagram that shows the way in which the NSGA-II works.

ϵ -Domination based Multi-Objective Evolutionary Algorithm (ϵ -MOEA):

To ensure elitism, the ϵ -MOEA algorithm [(Deb et al., 2005)] uses two populations co-evolving at the same time: population $P(t)$ and an archive $A(t)$ where t is the generation counter. The first step consists of building an initial population by a local or random search. The archive is initialized by the non ϵ -dominated individuals of this population. At each iteration of the algorithm, we find the classical steps: selection, crossover, and mutation, the only difference is that we generate a single child. The newborn child will be compared to the individuals of two populations for its possible inclusion. The pseudo-code of the algorithm is given below.

Algorithm 3 Pseudo code of the ϵ -MOEA algorithm**Input:** N : the size of population. T_{max} : the number of iterations.

- 1: Generate a uniform random population $P(0)$ of size N , and create the archive $A(0) = \emptyset$.
 - 2: Evaluate the fitness values of the population $P(0)$.
 - 3: Update of $A(0)$ from the non- ϵ -dominated individuals of $P(0)$.
 - 4: $t = 0$.
 - 5: **while** $t \leq T_{max}$ **do**
 - 6: Selection:
 - I. Randomly select two individuals from $P(t)$.
 - If one of them dominates the other, choose the dominant one. Let p be the selected individual.
 - II. Randomly select an individual a from $A(t)$.
 - 7: $e := \text{Variation_Operators}(p, a)$.
 - 8: Evaluate the fitness values of the new individual e .
 - 9: Update of the population:
 - I. If e dominates one or more individuals in the population then e replaces one of these randomly chosen individuals.
 - II. Else, If all the individuals of the population dominate e , then e is rejected.
 - III. If otherwise, If e is non-dominated to the individuals of the population and the individuals of the population is non-dominated to e , e replaces the random individual from the population.
 - 10: Update of the archive:
 - I. If e is ϵ -dominated by all members of the archive then e is rejected.
 - II. Else, If the child e ϵ -dominates one or more individuals then e will replace them in the archive.
 - III. If otherwise, If e is not ϵ -dominated by any member of the archive e is accepted.
 - 11: $t = t + 1$
 - 12: **end while**
- Output:**
 P : Population.
 A : Archive.

(b). Decomposition-based Algorithms (MOEA/D)

This class of MOEAs concerns approaches that decompose the multi-objective problems into a number of single-objective problems by generally relying on transformation algorithms to the single-objective. Although the premises of this approach are described by (Ishibuchi & Murata, 1998), it is the MOEA/D proposed by (Zhang & Li, 2007) that popularized this class of MOEAs due to its performance. In MOEA/D, the Tchebycheff method is, among others, used to

subdivide the original problem into subproblems. A classical evolutionary algorithm allows evolving a population in which each individual corresponds to the best solution of a specific sub-problem found since the beginning of the algorithm. A neighborhood relation, based on the distances between the weight vectors, is defined between the sub-problems. For the selection phase and for each sub-problem i , two individuals are selected from the neighborhood of i to participate in the crossovers. In the same way, during the replacement operation, the children generated by the individuals neighboring i can take the place of the individual i .

Algorithm 4 Pseudo code of the MOEA/D algorithm**Input:**

N : the number of the subproblems.

$\lambda^1, \lambda^2, \dots, \lambda^N$: weight vectors distributed uniformly.

T : neighborhood size

T_{max} : the number of iterations.

Step I: Initialization

- 1: Set $EP = \emptyset$.
- 2: **foreach** $(i, j \leq N \wedge i \neq j)$
- 3: Calculate Euclidean distance between λ^i and λ^j ;
- 4: **endforeach**
- 5: **for** $i = 1 \leftarrow N$ **do**
- 6: Set neighborhood of i as $B(i) = \{i_1, i_2, \dots, i_T\}$, where $\lambda^{i_1}, \lambda^{i_2}, \dots, \lambda^{i_T}$ are the T closest weight vectors to λ^i ;
- 7: **end for**
- 8: Generate initial population $P = (x^1, x^2, \dots, x^N)$ randomly.
- 9: Calculate objective function $FV_i = F(x^i)$, where $i = 1, 2, \dots, N$.
- 10: Initializ $z = (z_1, z_2, \dots, z_m)^T$;

Step II: Evolution

- 1: $t = 1$.
- 2: **while** $t \leq T_{max}$ **do**
- 3: **for** $i = 1 \leftarrow N$ **do**
- 4: Select two index k and l from $B(i)$.
- 5: Get a new solution y from x^k and x^l by applying reproduction operator.
- 6: Apply a problem-specific repair algorithm on y to produce y' .
- 7: **for** $j = 1 \leftarrow m$ **do**
- 8: **if** $f_j(y') < z_j$ **then**
- 9: $z_j := f_j(y')$
- 10: **end if**
- 11: **end for**
- 12: **for** $j \in B(i)$ **do**
- 13: **if** $g^{te}(y' | \lambda^j, z) \leq g^{te}(x^j | \lambda^j, z)$ **then**
- 14: Set $x^j = y'$ and $F(x^j) = F(y')$
- 15: **end if**
- 16: **end for**
- 17: Delete all vectors dominated by $F(y')$ from EP;
- 18: **if** no vectors in EP dominate $F(y')$ **then**
- 19: Add $F(y')$ to EP.
- 20: **end if**
- 21: **end for**
- 22: $t = t + 1$
- 23: **end while**

Output:

EP: external population.

(c). Indicator-based Algorithms

The last class of AEMOs uses a performance indicator (or metric). (Zitzler &

[Künzli, 2004](#)) were the first to propose such an approach and proposed the IBEA algorithm, which does not require a particular diversification mechanism. A binary performance indicator, which allows the comparison of two sets of solutions, is used. The fitness of an individual corresponds to the loss of quality (according to the indicator) generated by the deletion of this individual within the population. Thus, the higher this value is, the better the individual is. The selection and replacement phases are thus based on the fitness of the individuals.

Similarly, the SMS-EMOA algorithm ([Beume et al., 2007](#)) seeks to maximize the hypervolume metric at each generation. To do so, during the replacement phase, a sorting by front is first performed as in NSGA-II. In the last selected edge, the individuals that contribute the least to the hypervolume value are removed. Since the computation of the hypervolume metric can be excessively time consuming, ([Bader & Zitzler, 2011](#)) defined methods to speed up this computation, especially for solving problems with more than four objectives.

Finally, it is important to say that other metrics than hypervolume can be used as a performance indicator. For example, ([Brockhoff et al., 2015](#)) investigated the possibility of using a faster to compute alternative, the R2 metric ([Hansen & Jaszkiewicz, 1994](#)), in an AEMO that they named R2-EMOA. ([Menchaca-Mendez & Coello, 2015](#)) incorporated the GD convergence metric within their GDE-MOEA algorithm. Finally, an improved version of the IGD metric was defined by ([Tian et al., 2016](#)) and is used as a basis for comparing individuals.

2.4 Conclusion

In this chapter, we have presented the definitions and the main concepts used in the field of multi-objective optimization. As it has been defined, MOP is a problem that has several objectives to be optimized. The use of the word optimize comes from the fact that it is necessary to choose a solution that minimizes or maximizes the objectives of the problem, while in the case of a problem that contains a set of objectives it is not possible to find a solution that optimizes one objective function at the same time that it optimizes the other (in the case of two objectives). The appearance of this problem has led us to use a new concept of comparison between solutions, this concept is called Pareto dominance. Therefore, if we use the dominance relation to compare the solutions we find that the best solutions to choose are a set of solutions called non-dominated solutions or Pareto solutions, the application of the objective functions on the Pareto solutions produces what is called Pareto front.

The literature review of MOEAs was also presented. The use of evolutionary algorithms in multi-objective optimization has shown great success. EAs have the ability to find better approximations of Pareto solutions. EAs have three steps: selection, reproduction, and update. Several state-of-the-art algorithms in different multi-objective optimization frameworks have been discussed. In this chapter, we have also described the main algorithms that will be used in this thesis.

3

Supply Chain Management

3.1 Introduction

The function of supply chain management is in charge of supervising and coordinating the production, transportation, and delivery of goods and services from point of origin to final destination. We present the basic concepts of supply chain management, as well as the various problems that constitute the core of the thematic studied in this thesis are addressed in this chapter.

3.2 Basic concepts

3.2.1 Logistics

The use of the word logistics was in the French and American armies in order to secure the arrival of extensions such as moving military personnel equipment and goods, which helped armies win wars. As a result, the phrase was more frequently used following World War II to refer to the movement of commercial commodities along the supply chain. Logistics is the process of effectively organizing and carrying out the storage and transit of products from their point of origin to their point of consumption. The goal of logistics is to efficiently and promptly satisfy client needs. In order to meet consumer needs, the logistics component of the supply chain is in charge of planning, implementing, and managing the efficient, effective forward and reverse flow of goods, services, and related information between the place of origin and the site of consumption.



3.2.2 Supply chain (SC)

Supply chains are all activities responsible for managing and moving raw materials, parts and components necessary for the production process, as well as the final products, whether that movement is in the direction of the organization or from the organization to the direction of the markets.

Raw material: Raw materials are goods that will be used in the manufacturing process. Raw ingredients form part of a finished product and may be easily and affordably traced to that product.

Organization Company: is a group of people working to achieve a specific goal, which may be the goal of making profits by offering a product to its customers or providing a service to the community if it is an institution that is not based on profitability.

SC represents the connections and collaborations between suppliers, manufacturers, logistics businesses, wholesalers, retailers, and end customers. The supply chain process begins with the receipt of an order for a good or service by an organization and ends with the successful delivery of that good or service to the final customer.

The overall SC brings together multiple partners to source, manufacturer, transport, store, supply, and sell goods:

Suppliers: Produce raw materials or parts that can be manufactured into products.

Manufacturer: Create parts or products from raw materials and other inputs.

Logistics: Transports and stores goods as they move through the supply chain.

Wholesalers: Purchases goods for onward distribution to stores or other sales outlets.

Retailers: Sells finished products to end customers.

Customers: An individual or business that purchases the goods or services of another company. Customers are crucial because they provide revenue; businesses cannot function without them.

3.2.3 Supply chain management (SCM)

3.2.3.1 History

In (Tan, 2001), the authors present the historical evolution towards supply chain management. They identify the following main phases:

1950s and 1960s: In a market where supply was always satisfied because it was lower than demand, the strategy of companies was to minimize production costs. Thus, producers tended to mass-produce in order to reduce the unit production cost without worrying about product quality and innovation.

1970s: Manufacturing Resource Planning (also called Net Requirements Planning) was introduced. It represents a method of planning all the resources of a company. In addition, managers realized the importance of work in progress and its impact on cost, quality, innovation and delivery time. These criteria were considered as the four walls of a company.

1980s: At that time, competition increased, forcing companies to offer lower costs with better quality and greater flexibility. In addition, the concept of Just In Time (JIT), which consists of responding to demand as it arises, was born. Later, the importance of establishing partnerships between suppliers and customers began to be realized. And from this idea, the concept of SCM emerged.

Since the 1990s: Companies have started to apply the concept of SCM by creating partnerships with their suppliers and customers. Recent academic and professional research has shown the added value of this theory. However, the current market still presents companies that have not adapted this management method.

3.2.3.2 Example of Supply chain management

We can use the Coca-Cola firm as an example to demonstrate the concept of SCM (Figure 3.1). The Coca-Cola company is a multinational beverage corporation headquartered in Atlanta, Georgia. The business is interested in producing, retailing, and marketing alcoholic and non-alcoholic syrups and concentrates. One of the largest and most efficient SCM systems in the world is that of Coca-Cola.

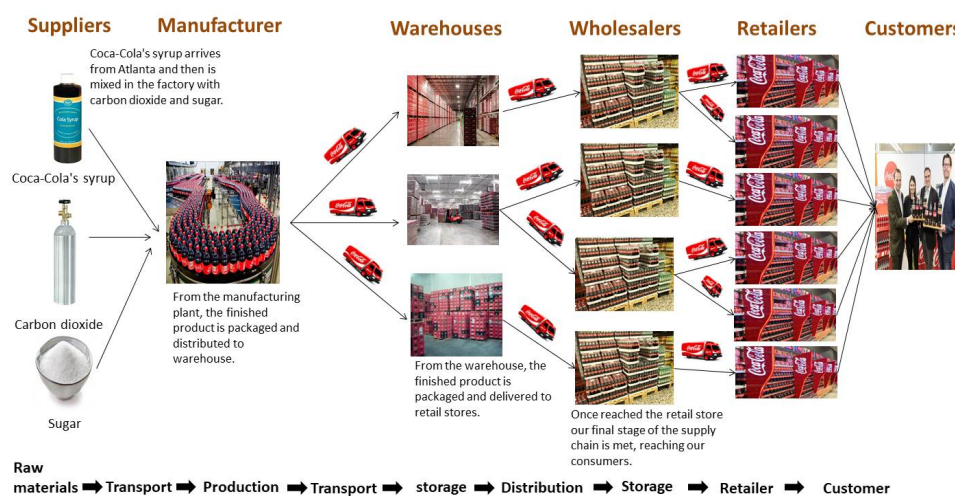


Figure 3.1 – An example of supply chain management-Coca-Cola.

Their SC involves many actors, starting with the suppliers (Coca-Cola's syrup, carbon dioxide, sugar), the manufacturer (product production), the warehouses (product storage, keep production, etc.), the distribution circuit (wholesalers, supermarkets, etc.) and of course the consumer. At each stage, transport companies are also involved.

3.2.3.3 Definition of SCM

The management of the flow of goods and services is known as SCM. It covers all processes that transform raw resources into finished goods. It entails the purposeful simplification of a business' supply-side operations in order to optimize customer value and obtain a competitive edge in the market. There are three types of flow in SCM:

- 1. Material flow:** The flow of material refers to the smooth movement of an item from the producer to the client. This is feasible thanks to a network of distribution, dealer, and retailer warehouses. The key problem we confront is ensuring that material flows as inventory promptly and without interruption via various points in the chain.
- 2. Information flow:** The flow of information is the basis for SCM's decision-making, the performance of a supply chain depends on it. Information serves as the foundation upon which managers make decisions and SC processes conduct transactions, making it critical to the SC's performance.

3. **Financial flow:** Financial transactions will take place between the customer and the supplier. Money may also travel from supplier to client in the form of debit notes. It is critical for an efficient and effective SC that all three flows are managed appropriately and with the least amount of work.

3.3 Decision phases in SCM

Decision phases are the several SCM processes that go into taking a decision or action connected to a certain good or service. Three decision processes must be completed in order for SCM to be successful in terms of information, product, and financial flow.

3.3.1 Supply Chain Strategy

In this stage, management makes the majority of the decisions. The choice to be made involves the cost of items that are highly expensive if it is incorrect and takes into account factors like long-term forecasting. At this point, it is crucial to analyze the market circumstances. These choices take into account the current and upcoming market situations. They make up the supply chain's structural design. The tasks and responsibilities of each are laid out after the layout has been prepared. The senior management or higher authority makes all of the strategic decisions. These decisions include selecting the production materials, choosing a factory location that makes it simple for transporters to load materials for delivery to specified places, choosing a warehouse location for the storage of finished goods, and many others.

3.3.2 Supply Chain Planning

Supply chain planning should take into account demand and supply. A market research should be conducted to identify customer demand. The second factor to take into account is being aware of current information about competitors and the tactics they employ to meet the needs and desires of their clients. Since different markets have distinct demand, they should be approached differently. This phase includes everything, starting with forecasting the market demand for the markets that will get the finished commodities for which a facility is designed in this stage. The company's participants and workers should all make an effort to make the process as adaptable as they can. A supply chain design phase is regarded as successful if it works well in short-term planning.

3.3.3 Supply Chain Operations

The third and final decision is made up of numerous practical choices that must be made immediately or in a matter of minutes, hours, or days. The decisional phase's goals include performance optimization and uncertainty minimization. Everything is covered in this step, starting with how the consumer is handled when using that product. Consider a scenario in which a customer demands a product that was made by our business. Orders are initially taken and forwarded to the manufacturing department and inventory department by the market department. Then, in response to a customer request, the production department sends the requested item to the warehouse via a suitable media, and the distributor promptly ships it to the consumer.

3.4 Key Issues in SCM

3.4.1 Globalization

Globalization offers firms and organizations with numerous significant supply chain management challenges:

First, businesses are shifting manufacturing operations to nations with lower labor costs, lower taxes, and/or lower transportation costs for raw materials in order to decrease expenses along the supply chain. Depending on the component of their product, some businesses may outsource production to more than one country. Second, as businesses increase their sales in international markets, they must localize their existing products in order to do so, which necessitates a substantial alteration to the supply chain as they adjust their offerings to various cultures and preferences. If business apps are not integrated, there is a risk of losing visibility, control, and proper management over inventory. Effective data management across different geographic contexts is necessary for this.

3.4.2 Fast-changing Markets

First, due to the rapid shifting market demands, product life cycles are shorter. Businesses are under pressure to stay on top of the most recent trends, innovate by launching new products, and maintain low total manufacturing costs because they are aware that trends will not continue for very long. This necessitates a supply chain that is adaptable and can be used to produce different goods as well as for future initiatives. Second, businesses need to update product features frequently in addition to developing new items. Businesses must adapt their supply chains to account for product changes while enhancing product features.

3.4.3 Quality and compliance

Compliance and product quality are frequently related. Businesses must make sure that they produce their goods in accordance with national and international legal requirements for packing, handling, and shipping. Enterprises must not only pass quality assurance and safety inspections, but they must also develop compliance documents, such as licenses, permits, and certifications, which can be overwhelming for them and their supply chain management systems. Blockchain, smart packaging, and Internet of Things (IoT) are revolutionizing how compliance is monitored and enforced. Managers should carefully consider where these investments make sense and check with IT to see if the company is employing platforms based on microservices and big data to meet these stringent data requirements.

3.5 Drivers of SCM

Supply chain capabilities are guided by the decisions you make about the two supply chain management drivers. Each of these drivers can be built and maintained to emphasize responsiveness or efficiency based on changing business requirements.

3.5.1 Logistical drivers of SCM

3.5.1.1 Location/ Facilities

In SCM, location is very important and that is why location is one of the drivers. Very large organisation worldwide have opted for smaller locations closer to their customers while to smaller manufacturers one plants is enough because it would be costly having branchers all over the place.

3.5.1.2 Inventory

As a driver of SC, inventory calls for efficiency in the way management handles their inventory. Efficient management inventory reduces cost in terms of holding, ordering and storage. Just in time (JIT) is one of the strategies that can be used in inventory management.

3.5.1.3 Transportation

This looks at the movement of raw materials from the raw material supplier to the manufacturer, then the movement of finished products to the final consumer as well as the reverse movement.

3.5.2 Cross-Functional drivers of SCM

3.5.2.1 Information

SC is complete only if there is both forward and backward flow. In these flows information is one of the most important, the way feedback is handled determines the responsiveness of the SC.

3.5.2.2 Sourcing

Sourcing is a collection of business operations used to acquire goods and services. Sourcing outlines the one who will carry out SC tasks such as production, warehouse, transportation, and information management.

3.5.2.3 Pricing

Pricing establishes the price to charge clients in SC. Pricing techniques can be used to balance supply and demand.

3.6 Classic SCM problems

3.6.1 The Facility Location Problem (FLP)

The FLP consists of deciding the location of facilities to satisfy customers while maximizing profits. This problem has been addressed by many authors, who present different types of models, but considering the same objective function: minimize location and transportation costs.

3.6.1.1 Problem Description

Being a directed network that represents any geographic region, where the nodes or cluster represent cities or groups of cities and the arcs represent the paths that join them. Suppose that the cities demand only one type of product and if we supply the demand of the city we will obtain income proportional to its number of inhabitants.

The model takes into account the following:

- 1) Each node represents a demand point.
- 2) This point of demand is satisfied only if a distribution center is installed.
- 3) Plants and distribution centers can only be located at nodes.
- 4) Once a distribution center is located, its demand must be supplied by one or more plants.
- 5) A plant can supply the demand of more than one distribution center at the same time.
- 6) Only one plant or one distribution center can be installed.

Graphically we can observe the different facilities p that are found on the left side, where for each point of demand the transport service must minimize the closest distance according to its location, at the same time considering both the quantity, weight and time in each of the shipments.

3.6.1.2 Mathematical formulation

Parameters:

I : Set of demand nodes.

J : Set of candidate locations.

f_j : Cost of locationg a facility at candidate site $j \in J$.

d_{ij} : Distance from demand node $i \in I$ to candidate facility site $i \in I$.

Decision Variables:

$X_j = \begin{cases} 1 & \text{If candidate site } j \in J \text{ is selected.} \\ 0 & \text{Otherwise.} \end{cases}$

$Y_{ij} = \begin{cases} 1 & \text{If demands at node } i \in I \text{ are served by a facility at node } j \in J. \\ 0 & \text{Otherwise.} \end{cases}$

The complete formulations is

$$\text{Minimize } z = \sum_{i \in I} \sum_{j \in J} d_{ij} Y_{ij} + \sum_{j \in J} f_j X_j \quad (3.1)$$

$$\text{s.t. } \sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \quad (3.2)$$

$$Y_{ij} \leq X_j \quad \forall i \in I \forall j \in J \quad (3.3)$$

$$X_j, Y_{ij} \in \{0, 1\} \quad \forall i \in I \forall j \in J \quad (3.4)$$

3.6.1.3 Classification and extension of FLP

1. Capacitated Facility Location Problem (CFLP)

CFLP is a variant of FLP where the capacity constraint on deposits is imposed. So in this case the sum of customer requests assigned to each depot must not exceed its capacity.

2. Set Covering Problem (SCP)

The SCP recovery problem consists of determining a subset of deposits to cover all customers. In location problems in general, two decisions have to be made at the same time: the choice of repositories and the assignment of customers. However, in the SCP, each depot only covers a set of predefined customers and therefore, to solve the SCP, we will only need to choose a subset of depots to cover (serve) all customers.

3. P-Median Problem (PMP)

The p-median problem is a location problem where we have a set of candidate depots and the goal is to choose exactly p depots from the set of these candidates (a chosen depot is called the median). The other candidate deposits are then assigned to the chosen medians. The objective of PMP is to minimize the assignment cost.

4. P-Centre Problem (PCP)

The p-center problem is a variant similar to the PMP problem whose goal is to reduce the maximum of customer assignment costs.

5. Fixed Charge Facility Location Problems

Among the key issues in location science are Fixed-Charge Facility Location Problems. There is a finite number of customers who have a need for service, and there is also a finite number of viable sites for the facilities that will provide service to customers. Decisions must be made in two categories: While allocation decisions define how to meet user demand from the present facilities, placement considerations determine where to build the facilities. The number of facilities to be located was a model input in the majority of the models we have examined so far. A different picture appears when we include in the fixed cost of facility location.

3.6.2 The Vehicle Routing Problem (VRP)

The VRP is a problem belonging to the class of combinatorial optimisation problems. The VRP problem consists in assigning each customer to a route carried out by a single vehicle and in finding an order of customer visits for each vehicle so as to satisfy the capacity constraints of the vehicles, and the quantities of product requested by each customer, in the event of a delivery problem. The objective in this problem is to find the set of routes that minimize the total distance traveled for a minimal number of vehicles leaving a depot and returning to it.

3.6.2.1 Mathematical formulation

Parameters:

$G = (V, A)$ is a directed graph, with $V = \{0, 1, 2, \dots, n\}$ is the node set, and A is the arc set.

n = number of customers.

0 = represents the depot.

$N = V \setminus \{0\}$ set of customers.

K = set of vehicles.

C_{ij} = traveling cost from $i \in V$ to $j \in V$.

Decision Variables:

$$X_{ij}^k = \begin{cases} 1 & \text{If vehicle } k \in K \text{ travels from node } i \in V \text{ to node } j \in V. \\ 0 & \text{Otherwise.} \end{cases}$$

The complete formulations is

$$\text{Minimize } z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} C_{ij} X_{ij}^k \quad (3.5)$$

$$\text{s.t. } \sum_{k \in K} \sum_{i \in N} X_{ij}^k = 1 \quad \forall j \in N \quad (3.6)$$

$$\sum_{k \in K} \sum_{j \in N} X_{ij}^k = 1 \quad \forall i \in N \quad (3.7)$$

$$\sum_{i \in V, i \neq p} X_{ip}^k - \sum_{j \in V, j \neq p} X_{pj}^k = 0 \quad \forall p \in N \quad \forall k \in K \quad (3.8)$$

$$\sum_{i \in N} X_{i0}^k = 1 \quad \forall k \in K \quad (3.9)$$

$$\sum_{j \in N} X_{0j}^k = 1 \quad \forall k \in K \quad (3.10)$$

Constraints (3.6) and (3.7) stipulate that each customer must be served once and only once while constraint (3.8) checks that the flow is preserved. Constraints (3.9) and (3.10) ensure that each round begins and ends with the deposit.

3.6.2.2 Classification and extension of VRP

1. **Capacitated Vehicle Routing Problem (CVRP):** The CVRP is a VRP where the vehicle capacity constraint is applied, meaning that the total of the customer requests served by each vehicle cannot exceed its capacity. The fleet of vehicles in the CVRP is assumed to be homogeneous.
2. **Vehicle Routing Problem with Periodic (PVRP):** In the PVRP, we have a set of periods $P = \{1, \dots, M\}$ which represents the planning horizon, and for each customer we have a number of visits $k (1 \leq k \leq M)$ which must be carried out. Therefore, to solve the PVRP we must, for each period, determine the customers to visit and build the vehicle routes to satisfy these customers.
3. **Vehicle Routing Problem with Pickup and Delivery (VRPPD):** As with the VRP, the VRPPD problem consists of developing a set of routes to satisfy customer demands. However in the VRPPD, we distinguish two types of request: the delivery of a product and the collection of a product.
4. **Vehicle Routing Problem with Time Windows (VRPTW):** The VRPTW is a VRP where a time constraint is imposed. Indeed, in the VRP a vehicle can visit a customer at any time. However, in the VRPTW a vehicle can only visit a customer within a specific time interval, this is called a time window.
5. **Open Vehicle Routing Problem (OVRP):** Unlike the VRP where the route of each vehicle must end at the central depot, in the OVRP problem the vehicles are not obliged to return to the central depot. Indeed, this feature is important because it changes completely the structure of VRP.
6. **Vehicle Routing Problem with Backhaul (VRPB):** As for the VRPPD, in this variant we distinguish two types of customers: deliverers where the vehicle must make a pickup and receivers where the vehicle must make a delivery. However the difference between the two variants is that in the VRPB we visit all the receivers before visiting the deliverers.

7. **Multi-Depot Vehicle Routing Problem (MDVRP):** In the VRP, each vehicle's route starts and ends from a central depot. However, in the MDVRP we have several depots and in each depot we have a set of vehicles. As in the VRP, each vehicle's route begins and ends from its depot and each customer must be served exactly once.

3.6.3 The Inventory Problem (IP)

Inventory refers to the materials (or things) preserved for later use. Resources could include any kind of things, people, machinery, etc. IPs typically develop at stores (where some items are sold) or factories (where goods are produced and raw materials are needed). Even though inventory is a resource that is idle, it is crucial to keep some inventory on hand for the business to run smoothly. The issue of how much inventory should be kept on hand and how many orders should be placed is known as the Inventory Control Problem (ICP).

3.6.3.1 Problem Description

Inventory control is an integral part of production control which includes both physical as well as financial control on material physical control lays emphasis that proper quality of material should be ensured in proper quantity at the proper time and at proper place so as to avoid the situation of shortage or surplus of material. Financial control helps in minimizing the cost of materials and investment in inventory. Some definitions of inventory control may be placed as under.

Inventory costs: The various costs considered in IP are:

1. **Holding cost or maintenance cost:** Holding cost is the expense of keeping merchandise until it is sold or used. C_1 denotes the holding cost per unit good per unit time. Rent for space, interest on capital inventory, insurance premiums, spoilage, breakage, record upkeep, and so on are all included.
2. **Set-up cost or procurement cost:** The cost of assembling machinery before beginning production is referred to as the set-up cost. It is represented by the symbol C_3 . It is the cost of a single production run (cycle). The cost of setting up is believed to be independent of the quantity produced. It is a set fee. It is also known as the ordering cost, which is the cost of placing an order for products. Purchase cost, labor cost, transportation cost, quality control, and so on are all included.
3. **Shortage cost or penalty cost:** The cost associated with either a delay or failure to satisfy demand is referred to as shortage cost. C_2 is the shortage cost per unit good per unit time.
In the case of unfulfilled demand at a later stage (backlog case) the costs are directly proportional to the shortage quantity as well as the delay in time. In case of inability to meet the demand (no backlog), the costs are proportional only to the shortage quantity. It includes loss of goodwill, sales lost, profit lost etc.
4. **Capital cost (Production cost or purchase cost):** The cost associated with an item whether it is manufactured or purchased is called the production cost or purchase cost or capital cost. Capital cost is denoted by P .

Variables in an inventory problem: The variables associated with the inventory problem are classified into two categories:

1. **Controlled variables:** are those that can be altered, such as the quantity of things added to the inventory, how often they are replenished, and how far along they are in the production process.

2. **Uncontrolled variables:** The variables that may not be controlled in an inventory problem like the inventory costs, demand, lead time are known as uncontrolled variables.

Terms used in an inventory problem:

1. **Stock replenishment:** The quantity of goods acquired in one replenishment in order to maintain a certain level of inventory is known as stock replenishment.
2. **Demand:** It is the number of items required per period. It is denoted by R . It may be uniform or probabilistic. If the demand is known and fixed, it is uniform demand. If the demand is not known with certainty it is probabilistic demand.
3. **Lead time:** The lead time is the period of time between placing an order and the products actually arriving at the inventory. The decision of buffer stock is heavily influenced by lead time.
4. **Time horizon:** The time period over which the inventory control is planned is called the time horizon.
5. **Economic order quantity or economic lot size (EOQ/ ELS):** The quantity of products purchased in inventory is referred to as the order quantity. Thus, the economic order quantity is the order amount that minimizes the total expenses of carrying inventory and ordering. Similarly, the economic lot size is the lot size (made in one lot) that minimizes the total expenses of carrying inventory and ordering.

Some general notations used in inventory models:

Q : Lot size per production run (order quantity).

C_1 : Holding cost per unit good per unit time.

C_2 : Shortage cost per unit good per unit time.

C_3 : Set up cost per production run.

R : Demand rate.

t : Time interval between two consecutive replenishments of inventory.

n : Number of replenishments per unit time (year).

$C(Q)$ or $C(Q, S)$: Average total cost per unit time.

I : Carrying cost per rupee per unit time.

P : Purchase cost (Capital cost).

Inventory models: The two categories of inventory models are:

1. Deterministic models.
2. Probabilistic models.

Inventory models in which demand is assumed to be uniform are deterministic models and the inventory models in which demand is a random variable are known as probabilistic models. Inventory models meant for deterministic demand are called EOQ models.

3.6.4 Integration problems

3.6.4.1 The Location Routing Problem (LRP)

The LRP has a number of real-world applications in areas of transportation. Based on the solution method, we have suggested the following classification scheme: Heuristic algorithms based on trajectory, or constructive heuristic algorithms. The fortress and the absence of each published technique are highlighted particularly, highlighting research prospects in the context of the problem's practical application.

3.6.4.2 The Inventory Routing Problem (IRP)

The IRP is a logistical issue that arises when a product needs to be distributed to a number of consumers within a predetermined time frame. A maximum inventory level is specified by each client. As part of a vendor-managed inventory policy, the supplier keeps track of each customer's stock and decides how much to replenish it, ensuring that there is never a stockout at the customer. A particular capacity of vehicle is used to transport goods from the supplier to the clients.

3.7 Financial supply chain management(FSCM)

3.7.1 Definition

Financial supply chain management(FSCM) is the process of supplier finance and producers in the supply chain to obtain the goods and services the company needs to produce its products or provide its services. This includes financing the purchase of raw materials and finished goods and providing the capital needed to manage production operations. Supply chains are typically financed through bank loans, advance payments, bank guarantees or co-financing between suppliers and buyers. FSCM is intended to improve the liquidity of companies, ensure the continuity of their operations and reduce the risks that companies may face in procurement and supply operations.

3.8 Green supply chain management (GSCM)

3.8.1 Environment

Environment refers to the natural world and surrounds in which all plants, animals, people, and other living things exist and function. The environment affects all things, both living and non-living. All natural elements that support life on earth, such as air, water, soil, sunlight, forested areas, plants, and animals, are included in the environment. It is believed that Earth is the planet in the universe with the conditions necessary for life to exist. Real environmental beauty is found in all natural elements. In the same way that trees are referred to as the earth's surface's green blanket. They maintain clean air and deliver oxygen necessary for both animal and human living. Without the natural resources that make up our surroundings, we are unable to imagine life. In our quest for advancement, we have lost sight of its value and significance. Three important types of environmental contamination are those in the air, land, and water. This has a significant impact on both the environment and living conditions around the world. If we continue to harm the environment and squander natural resources, we risk losing our life. We cannot afford to let industrialization and urbanization lead to the destruction of our ecosystem. The most crucial resource for maintaining life is the environment. We cannot predict the existence of life in the world without a healthy environment, thus we must maintain a safe and clean environment now and in the future. We must all do our part to protect trees by planting trees, using less plastic, and conserving all natural resources. Each and every person that lives on the planet is accountable for it. Everyone should step forward and participate in the company's environmental safety program.

3.8.2 Definition

GSCM is a philosophy that incorporates environmentally friendly practices into all aspects of SCM, including product design, material sourcing and selection, manufacturing processes, consumer delivery, and end-of-life management of the product after it has served its purpose. The GSCM focuses on mitigating the harmful impact of SC operations on the environment. The GSCM not only focuses on reducing air, water, and waste pollution but also focuses on green practices to improve organizational performance through less waste manufacturing, reuse and recycle of products, less manufacturing cost, effective asset use, and greater customers satisfaction.

3.9 Emergency Medical Services (EMS)

3.9.1 Definition

An EMS system is defined as a group of companies that provide transportation to nearby medical facilities and emergency medical care to anyone in need within a given geographic area. An EMS system typically consists of a fleet of ambulances or emergency response vehicles, ambulance stations where crews and vehicles wait until they are needed, a dispatch center that responds to emergency calls, assesses the circumstances, and dispatches the ambulances, and the hospitals with emergency departments that take in the patients. In general, three levels of EMS planning may be described, which differ in the time horizon over which decisions are made: strategic, tactical, and operational.

1. Strategic level mainly concern the location and construction of fixed locations, the purchase of equipment and the hiring and training of specialized employees.
2. Tactical level includes the development of work schedules and the location and relocation of ambulances.
3. Operational level concerns the rules for dispatching and redeploying vehicles, and the intervention procedures to be followed by the personnel.

3.9.2 Planning problems

3.9.2.1 Ambulance Location Problem (ALP)

The ALP belongs to the class of Maximum Covering Location Problems (MCLP). It aims to find the deployment sites for the ambulance fleet in a certain area to ensure adequate coverage of potential emergencies.

3.9.2.2 Ambulance Dispatching Problem (ADP)

The ADP falls under the class of Vehicle Routing Problem (VRP). It aims to select the best ambulances to dispatch to the emergency location after an emergency call is received in the EMS system. In addition, it is also necessary to maintain adequate coverage of the area served in order to be able to respond to emergency calls as soon as possible.

3.9.2.3 Ambulance Relocation Problem (ARP)

The ARP has a huge impact in the optimization of the EMS system. Thus, ARP is a strategy that modifies the locations of ambulances, during a day to better adapt to the evolution of the system. It aims to minimize the response times and assure a balanced plan for the available ambulances considering the existing fleet.

3.10 Conclusion

In this chapter, we have given an overview of supply chain management and then described in detail the three basic problems, namely, the location problem, the vehicle routing problem, and the inventory management problem. Finally, we dedicate the last part of this chapter to provide an overview of green supply chain management and emergency medical services problem in supply chain management.

Part II

Research works (Contributions)

4

G-MOEA/D-SA algorithm for the ambulance dispatching and relocation problem during COVID-19.

4.1 Introduction

Throughout history, humanity has faced numerous significant pandemics and epidemics, such as the plague, cholera, flu, SARS-CoV, and Middle East respiratory syndrome coronavirus (MERS-CoV). Currently, the world is grappling with the 2019 coronavirus disease pandemic (COVID-19). To mitigate the impact of such outbreaks, an efficient emergency response and preparedness program is crucial in reducing fatalities and injuries. Emergency medical services (EMS), as one of the most vital healthcare services (HCS), plays a key role in providing prompt pre-hospital care to patients. The effectiveness of response measures can contribute to both reducing the mortality rate and saving lives.

In this chapter, we present G-MOEA/D-SA, an enhanced version of the MOEA/D algorithm that addresses the Ambulance Dispatching and Routing Problem (ADRP) by incorporating simulated annealing (SA). G-MOEA/D-SA employs a straightforward yet efficient approach to determine optimal ambulance routes. Additionally, we provide an experimental study to demonstrate the effectiveness of our method. In our previous presentation, we conducted a comparative analysis of several state-of-the-art multi-objective optimization methods using real data collected during the Covid-19 pandemic in Saudi Arabia.

4.2 Literature review

The primary focus of research in managing EMS vehicles revolves around addressing dispatching and relocation problems. In 2012, (Ibri et al., 2012) introduced the first multi-agent approach that aimed to enhance long-term system performance by considering these issues. Another study by (Majzoubi et al., 2012) developed an integrated model for dispatching and relocation in real-time scenarios, increasing system capacity by enabling vehicles to attend multiple patients in different locations. To improve dispatching and relo-

cation decisions, (Billhardt et al., 2014) proposed dynamic approaches that accounted for ambulance redeployment in Madrid.

The Ambulance Relocation and Dispatching Problem (ARDP) was formulated as a linear programming model by (Bélanger et al., 2015), who also devised a method of mathematical decomposition to handle real-world scenarios. In 2016, (Andersson & Värbrand, 2016) provided an overview of decision support systems for dynamic ambulance relocation and autonomous ambulance dispatching. They evaluated the impact of applying dispatching and relocation models sequentially on key performance indicators while considering the operational requirements of a Colombian EMS provider. A flexible optimization framework for real-time ambulance dispatching and relocation was developed by (Nasrollahzadeh et al., 2018), allowing decision-makers to dispatch idle ambulances to cover the location of a recently dispatched ambulance, queue calls, or select any available ambulance. (Pechina et al., 2019) presented and evaluated scalable dispatching and relocation algorithms inspired by extensive research on emergency medical services. In a study by (Carvalho et al., 2020), a mathematical model, a heuristic pilot approach, and a time-preparedness metric were utilized to maximize system coverage by enabling relocations to any base. Lastly, (Bendimerad et al., 2021) utilized real data from Saudi Arabia to demonstrate the effectiveness of artificial intelligence algorithms in coordinating emergency response while ensuring adequate coverage for subsequent calls in the examined region.

4.3 Mathematical model

In this section, we mathematically represent the distribution of Covid-19 incoming calls across period $t = 1, 2, \dots, T$ as the Ambulance Dispatch and Relocation Problem (ADRP). The ADRP is defined on a graph $G = (N, A)$, where: The set of vertices N is represented as $N = W \cup S \cup K$.

The set of possible arcs A is represented as $A = (i, j) : i, j \in N, i \neq j$.

In this context, W represents the list of emergency Covid-19 calls that arrive at each time period t and need to be attended to. Each emergency Covid-19 call w is associated with a priority $Prio_w$ and a time taken p_w . The set S represents the EMS stations, and the set K represents the hospitals. The set of ambulances is denoted as $V = (V^1 \cup V^2 \cup V^3)$, where: V^1 represents the set of ambulances moving to respond to an emergency Covid-19 call.

V^2 represents the set of ambulances moving to transport patients to hospitals.

V^3 represents the set of ambulances moving back to their original EMS stations.

The travel time t_{ij} associated with each arc $(i, j) \in A$ is determined by the following equation:

$$t_{ij} = \frac{UnitPerHour \times d_{ij}}{vs} \quad (4.1)$$

Here, vs represents the speed of the ambulance in kilometers per hour, $UnitPerHour$ denotes the number of time units in one hour, and d_{ij} is the spherical distance between two geographical sites, as given by equation 4.2.

$$d_{ij} = \arcsin\left(\sqrt{\sin\left(\frac{\alpha_i - \alpha_j}{2}\right)^2 + \cos(\alpha_i) \times \cos(\alpha_j) \times \sin\left(\frac{\beta_i - \beta_j}{2}\right)^2}\right) \times 2R. \quad (4.2)$$

$R = 6371 \text{ km}$ denotes the radius of the Earth. α and β are the radians of each location's latitude and longitude, respectively (Bendimerad et al., 2021).

To define ADRP, we must first identify the relevant variables, as well as the ADRP's varied constraints and objectives.

4.3.1 Decision variable

To model a real problem mathematically, it is essential to define the decision variables. In this work, we used binary variables :

$$x_{jw}^s(t) = \begin{cases} 1 & \text{If ambulance } j \text{ travels from an origin EMS station } s \text{ to an emergency Covid-19 call } w \\ & \text{at time } t. \\ 0 & \text{Otherwise} \end{cases}$$

$$y_{jk}^w(t) = \begin{cases} 1 & \text{If ambulance } j \text{ travels from an emergency Covid-19 call } w \text{ to a hospital } k \text{ at time } t. \\ 0 & \text{Otherwise} \end{cases}$$

$$z_{jw}^k(t) = \begin{cases} 1 & \text{If ambulance } j \text{ travels from a hospital } k \text{ to an emergency Covid-19 call } w \text{ at time } t. \\ 0 & \text{Otherwise} \end{cases}$$

$$unsat_w(t) = \begin{cases} 1 & \text{If emergency Covid-19 call } w \text{ is not covered at time } t. \\ 0 & \text{Otherwise} \end{cases}$$

$$dev_j(t) = \begin{cases} 1 & \text{If ambulance } j \text{ is rerouted at time } t. \\ 0 & \text{Otherwise} \end{cases}$$

4.3.2 Constraints

- (1) The requirement is that each emergency Covid-19 call must be assigned to an ambulance for service. This constraint can be expressed as follows:

$$\sum_{j \in V} \sum_{s \in S} x_{ji}^s(t) = 1 \quad \forall i \in W \quad (4.3)$$

- (2) Every dispatched ambulance is designated to respond to an emergency Covid-19 call.

$$\sum_{i \in W} \left(\sum_{s \in S} x_{ji}^s(t) + \sum_{k \in K} z_{jw}^k(t) \right) = 1 \quad \forall j \in V^1 \cup V^3 \quad (4.4)$$

- (3) Ambulances that are en route to attend emergency Covid-19 calls are not allowed to divert to hospitals or other emergency Covid-19 calls. This constraint can be formulated as follows:

$$\sum_{k \in K} \sum_{w \in W} (y_{jk}^w(t) + z_{jw}^k(t)) = 0 \quad \forall j \in V^1 \quad (4.5)$$

- (4) Assigns these ambulances to hospitals.

$$\sum_{k \in K} \sum_{w \in W} y_{jk}^w(t) = 1 \quad \forall j \in V^2 \quad (4.6)$$

4.3.3 Objective functions

In order to simplify the solution of our problem, we set some intermediate objectives. We formulate them as follows:

1. The first objective function is to minimize the response time and the number of priority emergency Covid-19 calls that are not covered :

$$\min F_1 = \sum_{j \in V^1} \sum_{s \in S} \sum_{w \in W} (t_{jw} + p_w) \times x_{jw}^s(t) + \sum_{w \in W} \text{Pri}o_w \times \text{unsat}_i(t) \quad (4.7)$$

2. The second objective function is to minimize the relocation time and the number of deviated ambulances:

$$\min F_2 = \sum_{j \in V^2 \cup V^3} \sum_{k \in K} \sum_{w \in W} (t_{jk} \times y_{jk}^w(t) + t_{jw} \times z_{jw}^k(t)) + \sum_{j \in V^3} \text{dev}_j(t) \quad (4.8)$$

4.4 The proposed algorithm

4.4.1 G-MOEA/D-SA for ADRP

In order to address ADRP, the task involves receiving emergency Covid-19 calls at specific time intervals and promptly dispatching available ambulances during those times. Consequently, our approach facilitates the continuous update of various lists such as occupied ambulances, diverted ambulances, idle ambulances, and incoming emergency Covid-19 calls at each EMS station's time interval. When emergency Covid-19 calls arise within these intervals and ambulances are unoccupied, the G-MOEA/D-SA algorithm aims to determine the most optimal routes for deploying ambulances to handle the emergency Covid-19 calls while adhering to all constraints.

4.4.2 Outlines of G-MOEA/D-SA algorithm

Within the context of addressing ADRP, this chapter introduces a novel variation of the MOEA/D algorithm called G-MOEA/D-SA. This approach combines the MOEA/D algorithm with SA and incorporates an external archive as a storage facility for nondominated solutions, utilizing adaptive epsilon dominance. The pseudo-code of the proposed algorithm (G-MOEA/D-SA) is illustrated in Algorithm 5 and described as follows:

Algorithm 5 Pseudocode of G-MOEA/D-SA**Input:**

$g^{pbi}(\cdot)$: the penalty-based boundary intersection (PBI) approach (Zhang & Li, 2007).

M : the number of objective functions.

H : the positive integer.

T : the neighborhood size.

α : the cooling factor.

It_{max} : the number iterations of the simulated annealing (SA).

Output:

P : the parent population.

$Arch$: the external archive population.

Step I: Initialization Procedure

- 1: Generate $N = \binom{H+m-1}{m-1}$ weight vectors $\omega = \{\omega^1, \dots, \omega^N\}$ using Das and Dennis' technique (Das & Dennis, 1998).
- 2: Initialize N neighborhoods $B(1), B(2), \dots, B(N)$, where $B(i) = \{i_1, \dots, i_T\}$ stores the indexes of T closest weight vectors to ω^i .
- 3: Generate an initial parent population $P = \{x^1, x^2, \dots, x^N\}$ randomly.
- 4: Evaluate the fitness values (f_1, f_2, \dots, f_M) of the parent population P .
- 5: Initialize the reference point z^* .
- 6: Calculate $g^{pbi}(x^i | \omega^i, z^*)$ of each solution x^i in the parent population P .
- 7: Initialize the external archive population $Arch$ by all non-dominated solutions of the parent population P .

Step II: Evolutionary Procedure

- 1: **while** termination criterion is not reached **do**
- 2: **for** $i \leftarrow 1$ to N **do**
- 3: Randomly select two indexes k, l from $B(i)$.
- 4: $y \leftarrow$ Crossover_Operators (x^k, x^l) .
- 5: $y^* \leftarrow$ SA (y, α, It_{max}) .
- 6: Evaluate the fitness values $(f_1(y^*), f_2(y^*), \dots, f_M(y^*))$ of the solution (y^*) .
- 7: **for** $k \leftarrow 1$ to M **do**
- 8: **if** $z_k^* > f_k(y^*)$ **then**
- 9: $z_k^* \leftarrow f_k(y^*)$
- 10: **end if**
- 11: **end for**
- 12: Calculate $g^{pbi}(y^* | \omega^i, z^*)$ of the solution y^* .
- 13: **foreach** $j \in B(i)$ **do**
- 14: **if** $g^{pbi}(y^* | \omega^i, z^*) < g^{pbi}(x^j | \omega^i, z^*)$ **then**
- 15: $P \leftarrow P \cup \{y^*\} \setminus \{x^j\}$.
- 16: **end if**
- 17: **end**
- 18: $Arch \leftarrow$ UPDATE_ARCHIVE $(Arch, y^*)$.
- 19: **end for**
- 20: **end while**
- 21: **return** $P, Arch$.

The main objective of this algorithm is to strike a balance between exploration and exploitation during the search process. The crossover operator and solution exploitation both play crucial roles in achieving this goal. Additionally, by employing a convergence indicator-based dominance relation (CDR), SA aims to identify optimal routes for ambu-

lances to efficiently cover all emergency COVID-19 calls. The integration of an external archive within the G-MOEA/D-SA algorithm ensures that valuable solutions are preserved and not lost once they are discovered.

4.4.3 Complexity of G-MOEA/D-SA

One loop and two fundamental operations are performed by G-MOEA/D-SA when calculating the computational complexity of one iteration. Let D is the dimension of the problem, N be the population size and the archive size, M is the number of objective functions, and $It_{e_{max}}$ is the number of iterations of the SA. The first operation (SA) need $It_{e_{max}}$ develop, so the complexity of this operation in this case is $O(N * It_{e_{max}} * (D + M))$. The second operation (updating archive) has a computational complexity of $O(M * N^2)$. As a result, the entire computational complexity of G-MOEA/D-SA algorithm is $max(O(N * It_{e_{max}} * (D + M)), O(M * N^2)) = O(M * N^2)$.

4.5 Experimental results

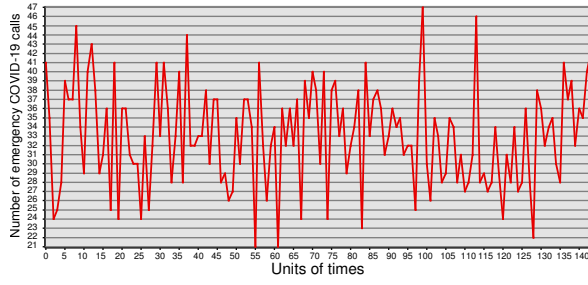
4.5.1 Dataset Description

In order to better understand how ambulances were distributed when Covid-19 instances increased, we first examine the daily statistics Saudi Arabia experienced (<https://raw.githubusercontent.com/RamiKrispin/nncoronavirus/master/csv/coronavirus.csv>). The greatest peak recorded by Saudi Arabia was 4919 cases, as determined by an analysis of the daily data up until August 3, 2021. We decided to generate these 4919 cases as emergency calls in order to create the dataset of Covid-19 emergency calls, while also taking into account their regional distribution as shown in figure 4.1.

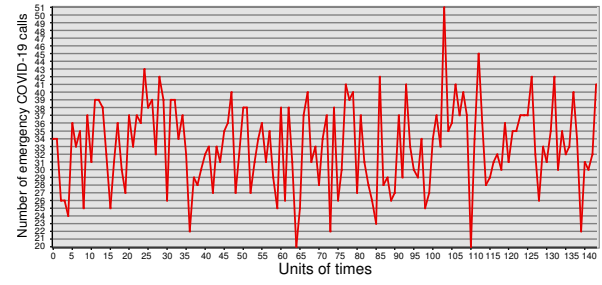


Figure 4.1 – A graphical summary of all emergency Covid-19 call locations (circles).

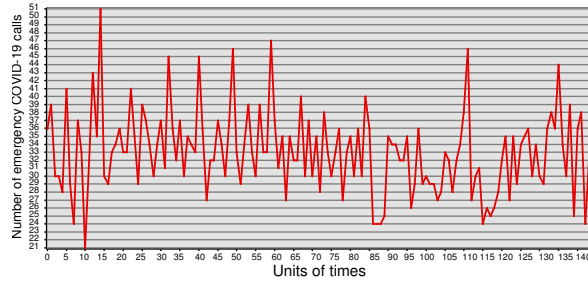
The time limit for each period, denoted as t , has been set to 10 minutes, resulting in a total of 144 time intervals per day. Within each time interval, EMS stations randomly select a specific number of emergency COVID-19 cases, generating ten distinct datasets based on these criteria. The random distribution of COVID-19 calls across all 144 intervals is illustrated in Figure 4.2. Each dataset instance represents an emergency COVID-19 call with a random priority level and location. To construct our dataset, we have included 207 hospital locations out of the 5873 hospitals in Saudi Arabia that are capable of admitting COVID-19 patients. Additionally, the dataset incorporates 374 EMS station sites, which are referenced from the source (<https://www.srca.org.sa/ar/Centers/All>). In this particular study, it is assumed that there is a single ambulance stationed at each ambulance station.



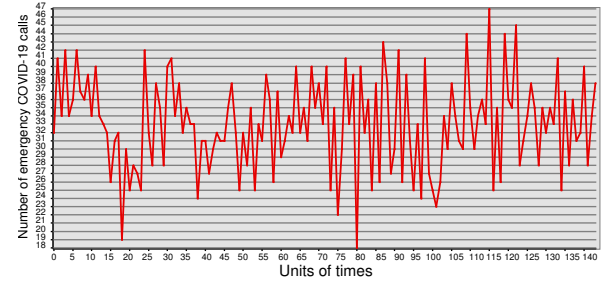
(a) Datasets 1



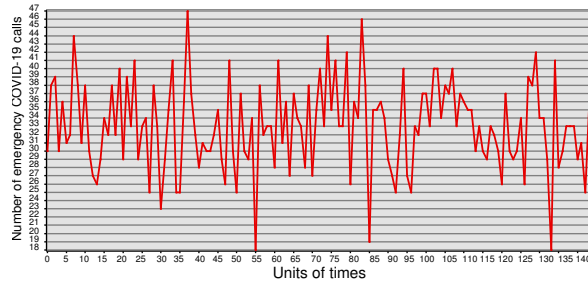
(b) Datasets 2



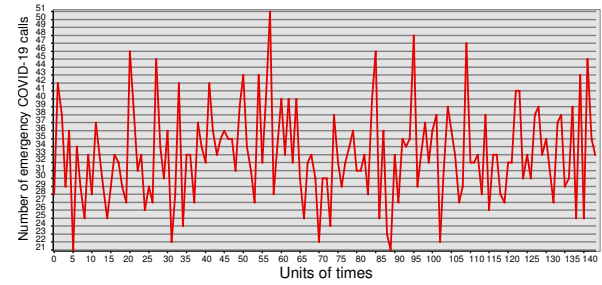
(c) Datasets 3



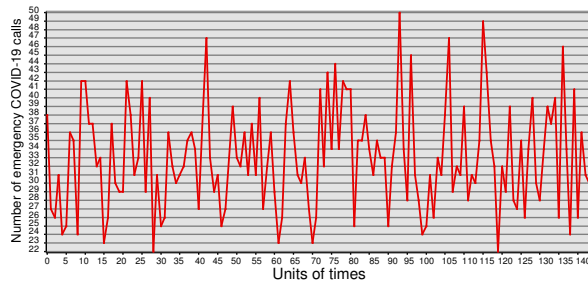
(d) Datasets 4



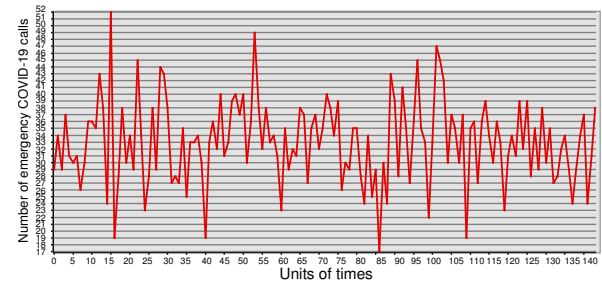
(e) Datasets 5



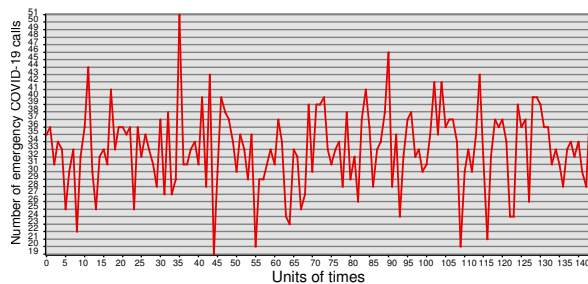
(f) Datasets 6



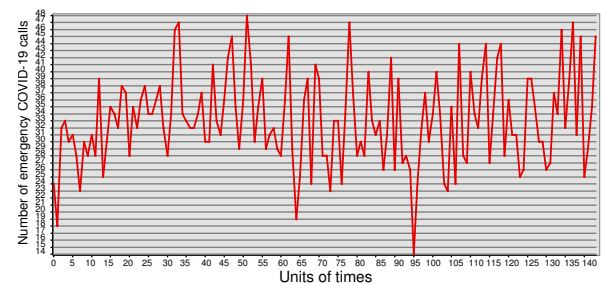
(g) Datasets 7



(h) Datasets 8



(i) Datasets 9



(j) Datasets 10

Figure 4.2 – All cases of randomization of incoming COVID-19 calls over 144 periods.

4.5.2 Experimental Results and Analysis

4.5.2.1 The effect of simulated annealing (SA):

To assess the impact of SA on the performance of the G-MOEA/D-SA algorithm, a comparison is made between the proposed algorithm with and without SA, using the IGD and HV metrics. In the experiment, the G-MOEA/D-SA algorithm is executed 10 times both with and without SA. The IGD and HV results across all datasets at each time unit are presented in Table 4.1. The differences in IGD and HV values between the G-MOEA/D-SA algorithm with and without SA are depicted in the corresponding figures in the table. Examining the first column of IGD figures, it is evident that the G-MOEA/D-SA algorithm with SA consistently achieves the lowest values compared to the G-MOEA/D-SA algorithm without SA across all datasets. Similarly, in the second column of HV figures, the G-MOEA/D-SA algorithm with SA consistently exhibits the highest values compared to the G-MOEA/D-SA algorithm without SA for all datasets at each time unit. These results demonstrate that the G-MOEA/D-SA algorithm with SA exhibits superior anytime behavior when compared to the G-MOEA/D-SA algorithm without SA.

Table 4.1 – The IGD and HV metrics were computed using the G-MOEA/D-SA algorithm applied to Datasets1 to Datasets10 with various time units. Both the versions of G-MOEA/D-SA with and without SA were used in the evaluation.

Datasets	IGD	HV
Datasets1		
Datasets2		
Datasets3		
Datasets4		

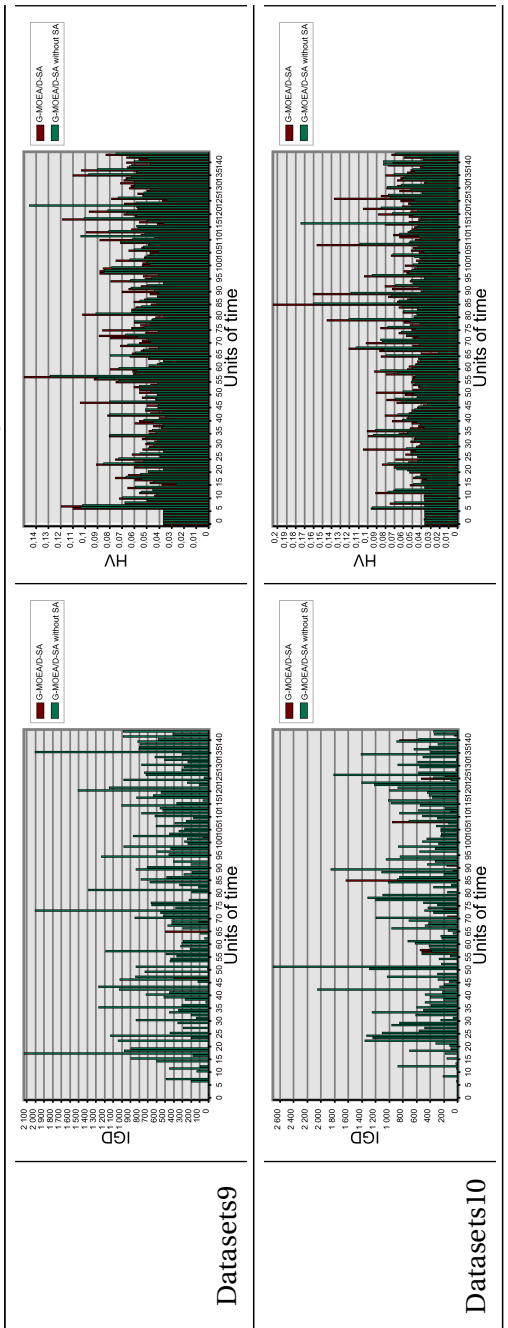
Continued on next page

Table 4.1 – continued from previous page

<p>Datasets5</p>		
<p>Datasets6</p>		
<p>Datasets7</p>		
<p>Datasets8</p>		

Continued on next page

Table 4.1 – continued from previous page



4.5.2.2 Comparisons with other multiobjective algorithms

The G-MOEA/D-SA, MOEA/D, MOEA/D-M2M, and NSGA-II algorithms are individually executed 10 times for each dataset. Table 4.2 and Table 4.3 present the median value (MV) and interquartile range value (IV) of IGD and HV, respectively, across all datasets at each time unit. These values are visually depicted as curves in the tables. Analysis of Table 4.2 reveals that all curves corresponding to G-MOEA/D-SA consistently achieve smaller IGD values compared to the curves of MOEA/D, MOEA/D-M2M, and NSGA-II across all datasets. The experimental results indicate that G-MOEA/D-SA outperforms the other algorithms in solving problems with varying time units. Similarly, Table 4.3 shows that all curves for G-MOEA/D-SA consistently exhibit larger HV values compared to the curves of MOEA/D-M2M and NSGA-II across all datasets. These statistical test results suggest that the incorporation of simulated annealing (SA) using CDR-dominance and an external archive based on epsilon dominance enables a balanced trade-off between convergence and diversity.

Additionally, Table 4.4 and Table 4.5 present the MV and IV values of IGD and HV, respectively, across all datasets. The MV and IV are calculated using equations 4.9 and 4.10, respectively.

$$MV = \left(\sum_{t=1}^{144} M(t) \right) / 144. \quad (4.9)$$

$$IV = \left(\sum_{t=1}^{144} I(t) \right) / 144. \quad (4.10)$$

Where $M(t)$ represents the median at time unit t and $I(t)$ represents the interquartile range at time unit t .

Analyzing Table 4.4, we observe that G-MOEA/D-SA exhibits the lowest MV and IV values of IGD across all datasets, while MOEA/D-M2M ranks second in terms of MV and NSGA-II ranks second in terms of IV. Turning to Table 4.5, it is apparent that G-MOEA/D-SA maintains its first rank with the highest MV values and lowest IV values of HV across all datasets. Similarly, MOEA/D ranks second in terms of MV, while NSGA-II ranks second in terms of IV. Based on these results, we can conclude that our algorithm, G-MOEA/D-SA, is highly competitive, outperforming MOEA/D-M2M, MOEA/D, and NSGA-II in terms of both convergence and diversity.

Table 4.2 – Median and interquartile range values of IGD produced by the four algorithms on Datasets 1–10 with various time units.

Datasets	Median of IGD	Interquartile range values of IGD
Datasets1		
Datasets2		
Datasets3		

Continued on next page

Table 4.2 – continued from previous page

<p>Datasets4</p>		
<p>Datasets5</p>		
<p>Datasets6</p>		
<p>Datasets7</p>		

Continued on next page

Table 4.2 – continued from previous page

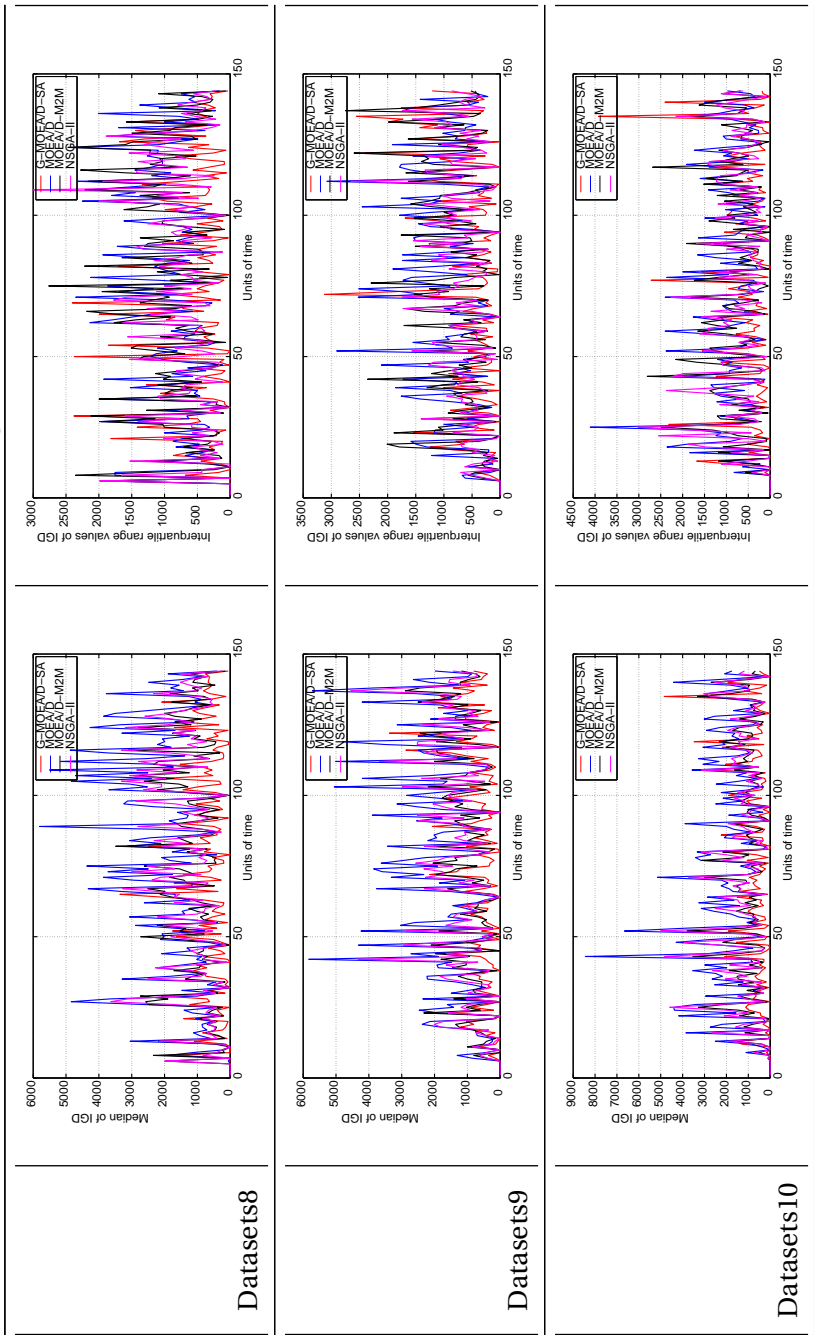


Table 4.3 – Median and interquartile range values of HV produced by the four algorithms on Datasets 1–10 with various time units.

Datasets	Median of HV	Interquartile range values of HV
Datasets1		
Datasets2		
Datasets3		

Continued on next page

Table 4.3 – continued from previous page

<p>Datasets4</p>		
<p>Datasets5</p>		
<p>Datasets6</p>		
<p>Datasets7</p>		

Continued on next page

Table 4.3 – continued from previous page

<p>Datasets8</p>		
	<p>Datasets9</p>	
<p>Datasets10</p>		

Table 4.4 – The MV and IV of IGD were computed for the four algorithms on Datasets1 to Datasets10.

	MV				IV			
	G-MOEA/D-SA	MOEA/D	MOEA/D-M2M	NSGA-II	G-MOEA/D-SA	MOEA/D	MOEA/D-M2M	NSGA-II
Datasets1	5.5147E+02	1.9350E+03	1.0168E+03	1.2819E+03	3.7098E+02	9.6189E+02	8.3864E+02	7.6046E+02
Datasets2	6.0095E+02	2.0571E+03	1.1367E+03	1.4022E+03	4.0497E+02	9.2070E+02	8.6458E+02	7.6932E+02
Datasets3	5.1645E+02	1.5165E+03	8.5984E+02	1.0205E+03	4.0126E+02	8.8509E+02	7.1883E+02	6.6695E+02
Datasets4	5.3381E+02	1.9328E+03	1.0330E+03	1.1593E+03	4.2887E+02	9.3662E+02	7.9342E+02	7.1339E+02
Datasets5	4.2225E+02	1.7267E+03	9.7107E+02	1.1835E+03	3.6777E+02	8.3802E+02	7.2649E+02	6.7613E+02
Datasets6	5.2435E+02	1.8441E+03	1.0343E+03	1.2164E+03	4.4350E+02	8.7453E+02	8.5064E+02	7.9590E+02
Datasets7	4.5706E+02	1.6203E+03	8.9941E+02	1.1293E+03	4.1823E+02	8.6446E+02	6.8467E+02	6.8102E+02
Datasets8	6.1072E+02	1.8228E+03	1.0961E+03	1.2336E+03	4.9190E+02	8.9901E+02	8.8272E+02	7.1245E+02
Datasets9	5.8966E+02	1.6750E+03	9.2095E+02	1.1667E+03	5.1984E+02	8.8420E+02	7.4079E+02	6.3417E+02
Datasets10	6.1752E+02	1.7454E+03	9.4285E+02	1.1777E+03	4.5319E+02	8.2947E+02	7.1101E+02	6.9147E+02

Bold values show the result is better

Table 4.5 – The MV and IV of HV were computed for the four algorithms on Datasets1 to Datasets10.

	MV				IV			
	G-MOEA/D-SA	MOEA/D	MOEA/D-M2M	NSGA-II	G-MOEA/D-SA	MOEA/D	MOEA/D-M2M	NSGA-II
Datasets1	7.0956E-02	4.6809E-02	6.1001E-02	5.3997E-02	4.0216E-03	8.9343E-03	7.7921E-03	8.1058E-03
Datasets2	7.5063E-02	4.8159E-02	6.3076E-02	5.6255E-02	3.6856E-03	8.0279E-03	7.3730E-03	7.1934E-03
Datasets3	5.7337E-02	4.1992E-02	5.1530E-02	4.7507E-02	3.0777E-03	6.6367E-03	5.2126E-03	5.6269E-03
Datasets4	7.1197E-02	4.9537E-02	6.5761E-02	6.1138E-02	7.8310E-03	1.1332E-02	1.1548E-02	1.2122E-02
Datasets5	6.2965E-02	4.4863E-02	5.5800E-02	5.1455E-02	4.1183E-03	8.3171E-03	6.5422E-03	8.1451E-03
Datasets6	5.9151E-02	4.3251E-02	5.3163E-02	4.9311E-02	3.0348E-03	7.9693E-03	7.1742E-03	8.2982E-03
Datasets7	5.7312E-02	4.3255E-02	5.2857E-02	4.9279E-02	3.4637E-03	9.7165E-03	8.1344E-03	8.6658E-03
Datasets8	6.5182E-02	4.5957E-02	5.8128E-02	5.2999E-02	4.8154E-03	8.4821E-03	7.2823E-03	8.2065E-03
Datasets9	6.0463E-02	4.3659E-02	5.4265E-02	4.9883E-02	2.9712E-03	7.0356E-03	6.1575E-03	6.8183E-03
Datasets10	6.2174E-02	4.6417E-02	5.8787E-02	5.0877E-02	3.0213E-03	1.3584E-02	1.1664E-02	1.2236E-02

Bold values show the result is better

4.5.2.3 Comparisons on performance computational time

To assess computational performance, all algorithms are executed once over 100 iterations. The computational times achieved by each algorithm on all datasets at each time unit are graphically represented in Figures 4.3 to 4.12. From the results, it can be observed that NSGA-II and MOEA/D-M2M are the fastest algorithms across all datasets. Following NSGA-II and MOEA/D-M2M, G-MOEA/D-SA, which delivers excellent results for one dataset (Datasets6), is the second fastest algorithm. These findings highlight that MOEA/D is the slowest algorithm, with a significant difference in execution time. It requires a substantial amount of time to complete. In comparison to the MOEA/D-M2M algorithm, the proposed G-MOEA/D-SA approach demonstrates exceptional competitiveness and generally outperforms MOEA/D in terms of computation time.

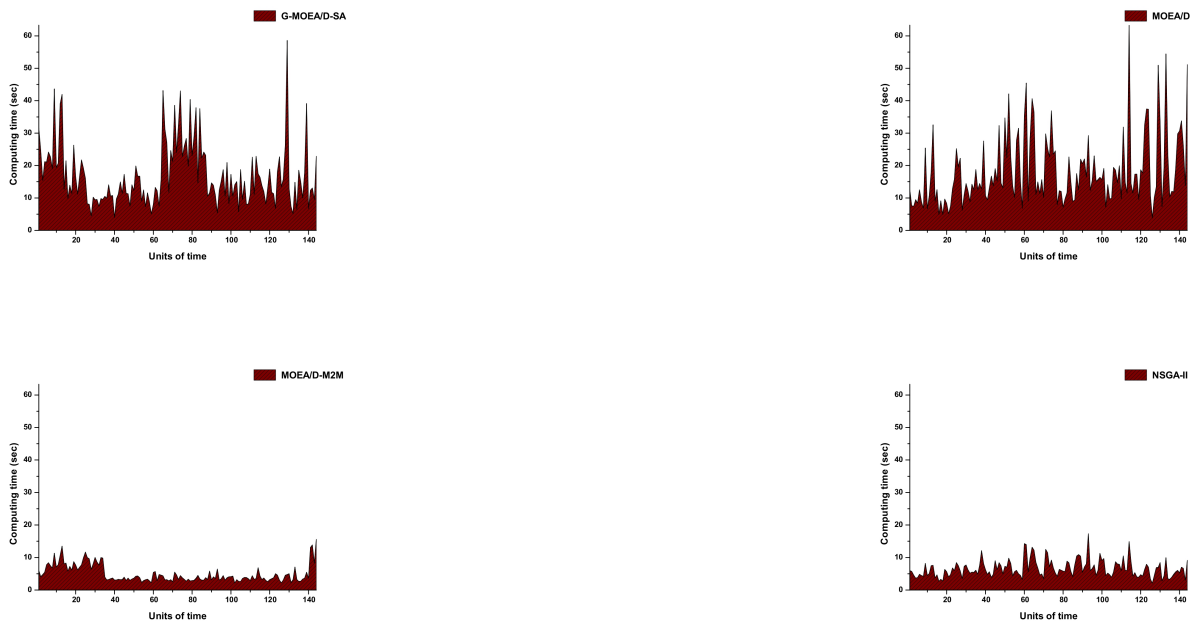


Figure 4.3 – Computational time for Datasets1.

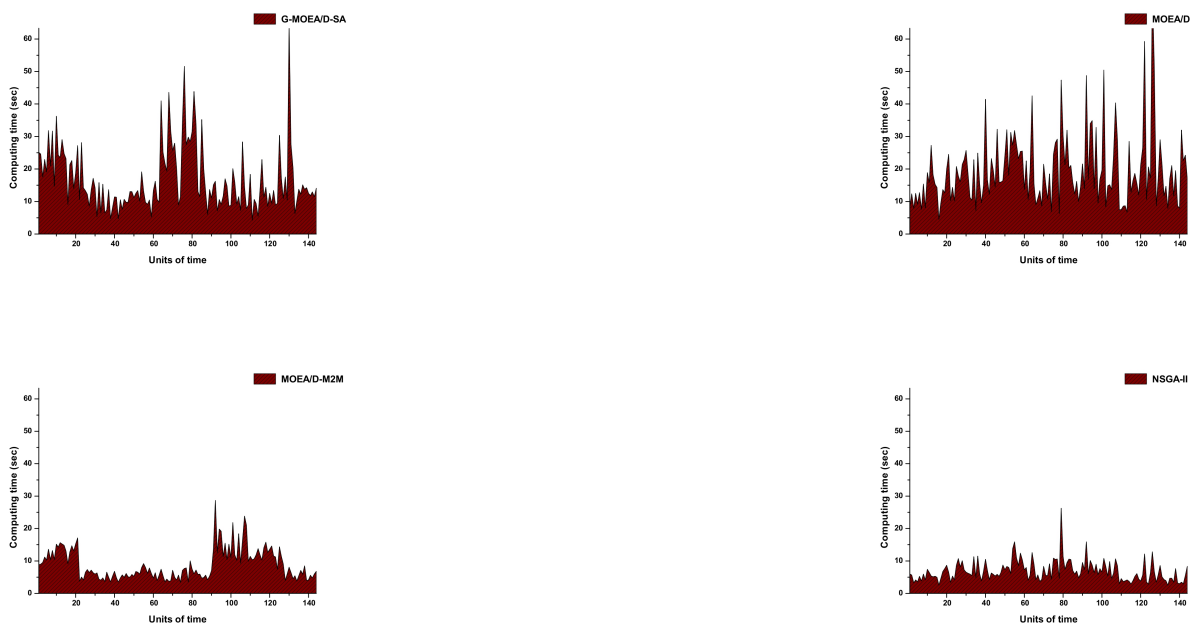


Figure 4.4 – Computational time for Datasets2.

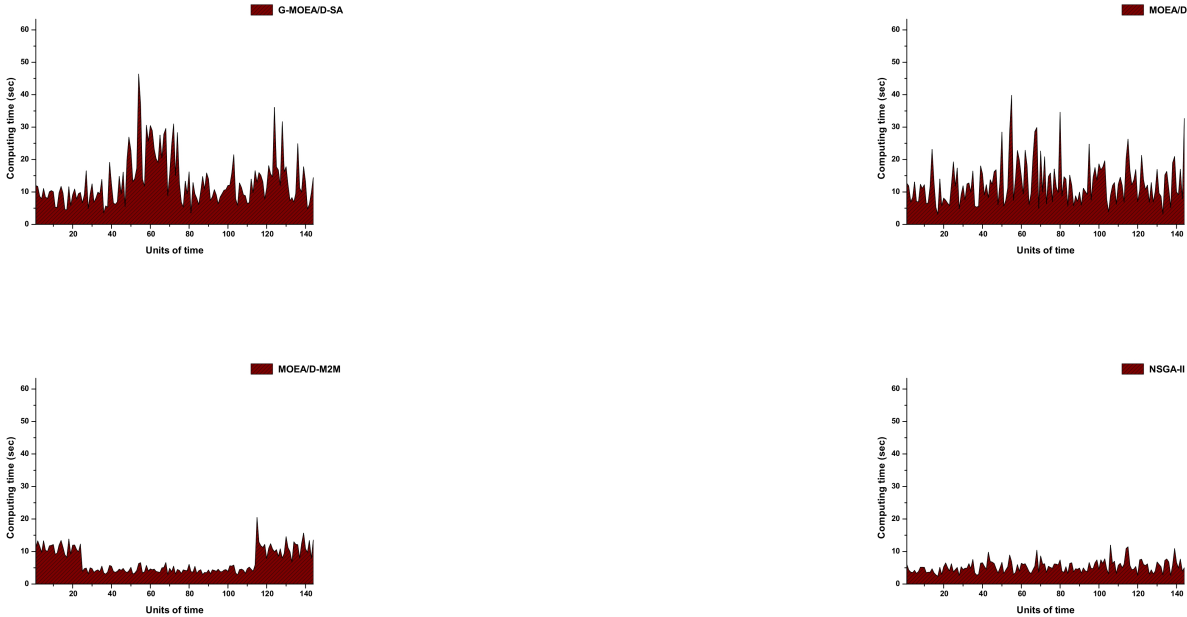


Figure 4.5 – Computational time for Datasets3.

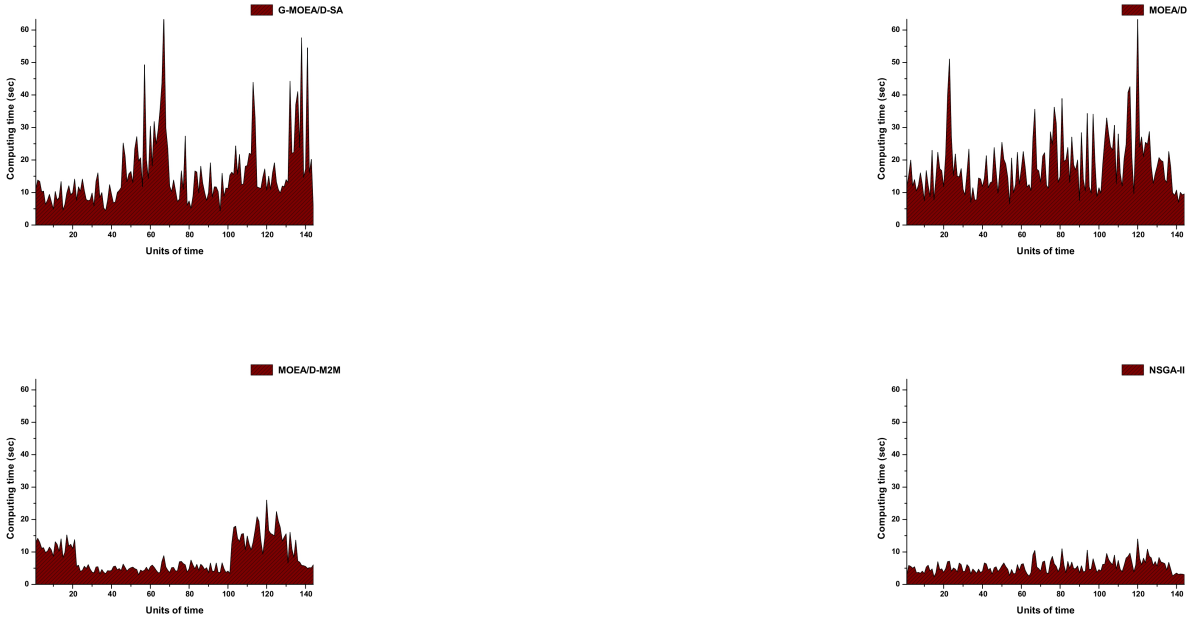


Figure 4.6 – Computational time for Datasets4.

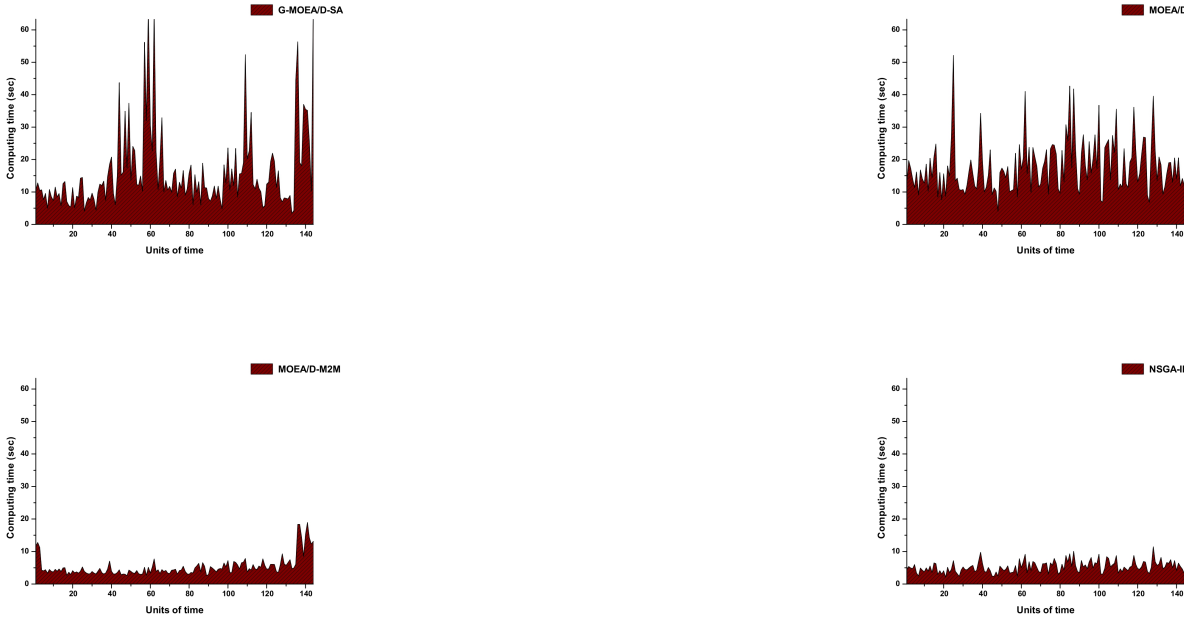


Figure 4.7 – Computational time for Datasets5.

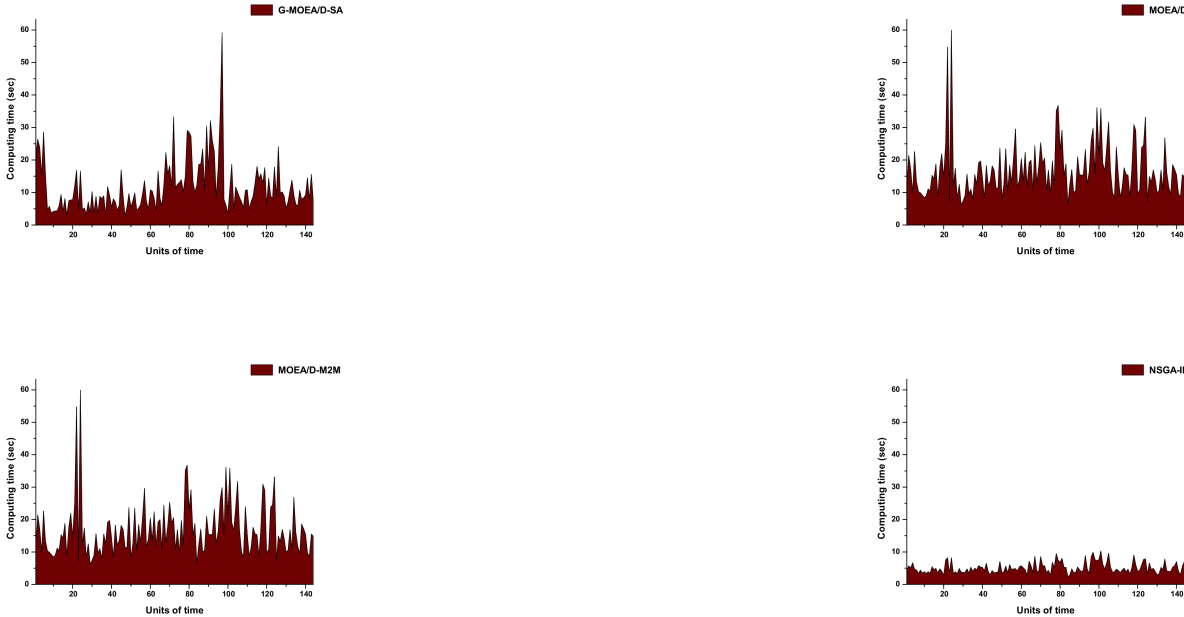


Figure 4.8 – Computational time for Datasets6.

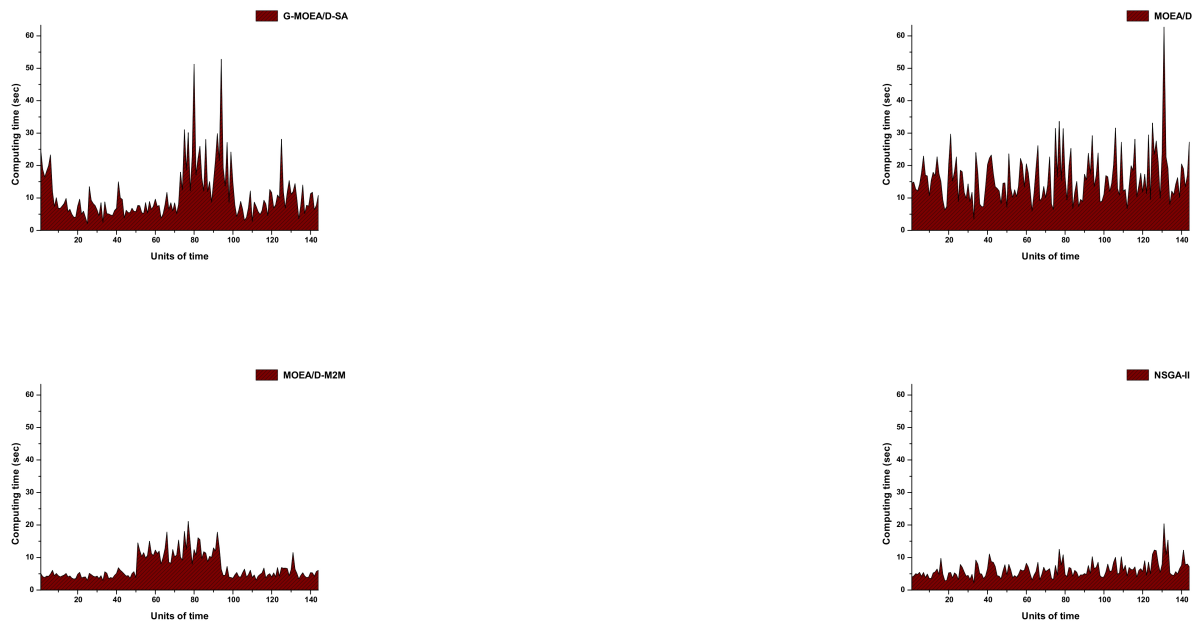


Figure 4.9 – Computational time for Datasets7.

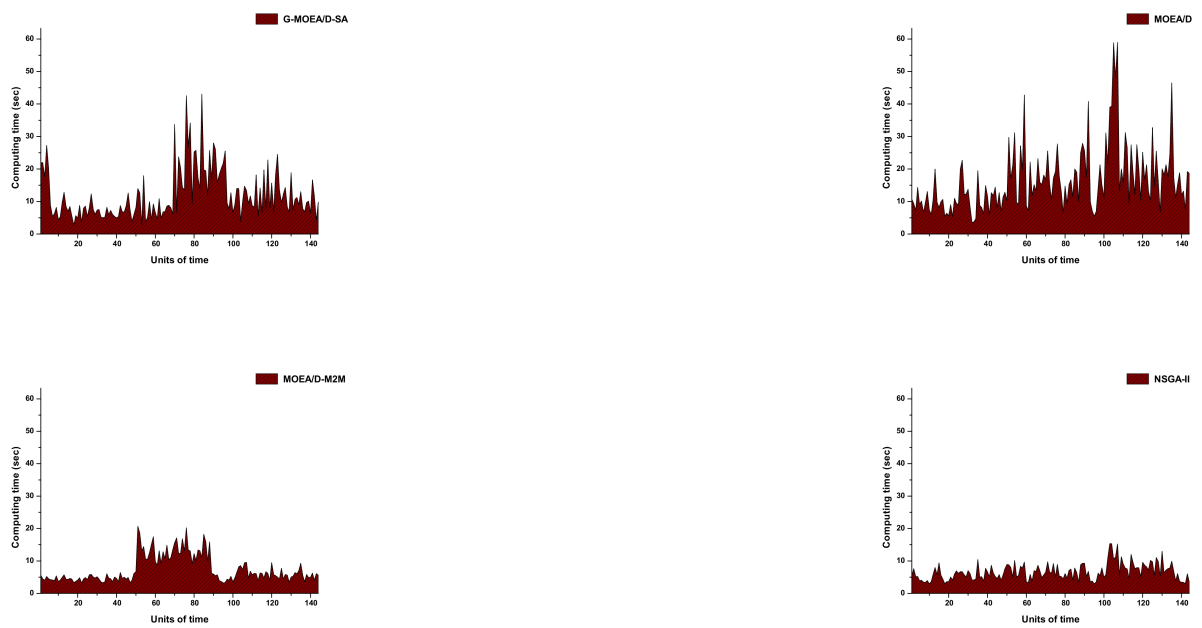


Figure 4.10 – Computational time for Datasets8.

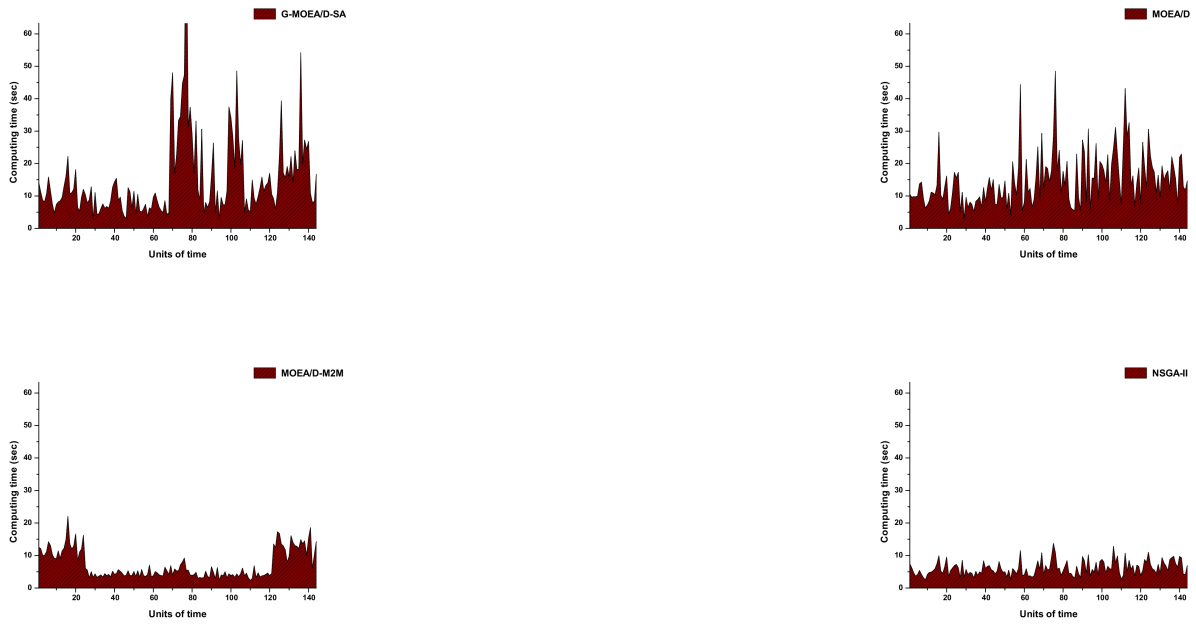


Figure 4.11 – Computational time for Datasets9.

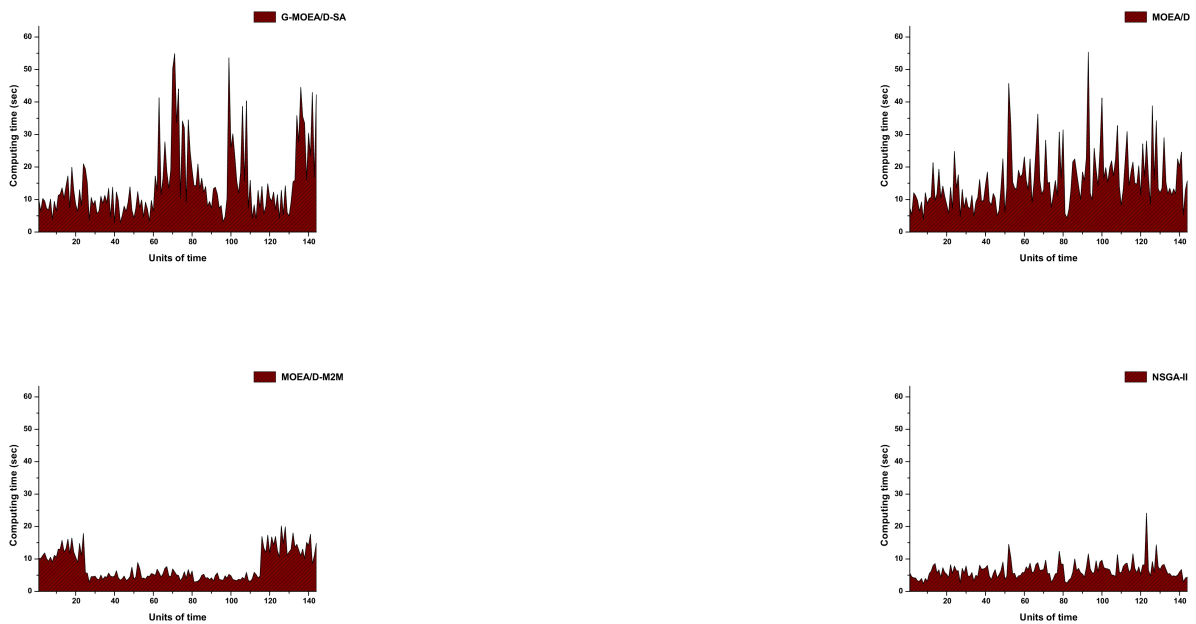


Figure 4.12 – Computational time for Datasets10.

4.6 Conclusion

In this chapter, we have introduced a novel multi-objective evolutionary algorithm called G-MOEA/D-SA, which we have applied to address the real-time challenges of ambulance

dispatching and relocation. Our proposed algorithm combines the standard MOEA/D framework with SA to enhance solution quality. Additionally, an external archive is utilized to maintain a favorable trade-off between convergence and diversity. To evaluate the effectiveness of the G-MOEA/D-SA algorithm, we conducted experiments using real data from Saudi Arabia during the Covid-19 pandemic. The algorithm has demonstrated promising results in terms of convergence and diversity, which are key objectives in heuristic multi-objective optimization. These findings support the applicability of our proposed algorithm in various domains and future applications. This work has been published in the international journal *Applied soft computing*¹, see (([Hemici et al., 2023](#))).

1. <https://www.sciencedirect.com/science/article/pii/S1568494623003009>

5

VRPTW-MOEA algorithm for the Vehicle Routing Problem with Time Windows

5.1 Introduction

In various organizations, effective management of collection and distribution has emerged as a crucial decision-making problem. Companies have come to recognize the significant impact of these activities on their overall costs, particularly transportation expenses that form a substantial portion of logistic costs. Among the critical problems in supply chain management, the vehicle routing problem (VRP) holds great significance, and several variants have been developed, including the vehicle routing problem with time window (VRPTW). The VRPTW involves routing a fleet of vehicles with capacity from a central depot to customers with specific demands and time windows for service.

In this chapter, we will introduce a novel multiobjective evolutionary algorithm named VRPTW-MOEA, designed to solve the multiobjective vehicle routing problems with time windows (MOVRPTW). To assess the performance of our algorithm, we include an experimental study at the conclusion of this chapter. Furthermore, we present a comparative analysis of our results with previously published studies using the widely recognized 56 Solomon's data sets.

5.2 Literature review

In the field of solving the multiobjective vehicle routing problem with time windows (MOVRPTW), several approaches have been proposed by different researchers. For instance, (Gambardella et al., 1999) utilized ant colony optimization and transformed the multiobjective VRPTW into a single objective problem by introducing a weight penalty term. (Tan et al., 2006) introduced a hybrid multiobjective evolutionary algorithm that simultaneously optimized all routing constraints and objectives, resulting in improved routing solutions. (Ombuki et al., 2006) developed a genetic algorithm approach using Pareto ranking to optimize both the number of vehicles and total distances. (Ghoseiri & Ghanadpour, 2010) proposed an adapted genetic algorithm that incorporated various heuristics for local exploitation during the evolutionary search. (Garcia-Najera & Bullinaria, 2011)

provided an improved multiobjective evolutionary algorithm by incorporating a similarity measure between solutions. (Hsu & Chiang, 2012) introduced a multiobjective evolutionary algorithm with enhanced route exchange crossover and mutation operators to solve the VRPTW. (Chiang & Hsu, 2014) proposed a knowledge-based evolutionary algorithm (KBEA) aiming to minimize the number of vehicles and total distance simultaneously. (Qi et al., 2015) developed a memetic multiobjective evolutionary algorithm with a novel selection operator (M-MOEA/D) for MOVRPTW. (Dong et al., 2018) proposed a modified discrete glowworm swarm optimization algorithm based on time window division (MDGSOTWD) for MVRPTW. (?) proposed a multiobjective evolutionary algorithm based on a fast elite sampling strategy and difference-based local search (MOEAFESS/DLS) to solve VRPTW. (Moradi, 2020) introduced a new multiobjective discrete learnable evolution model (MOD-LEM) for VRPTW. (Khoo & Mohammad, 2021) proposed a parallelization of a two-phase distributed hybrid ruin-and-recreate genetic algorithm (HRRGA) to solve VRPTW.

5.3 Mathematical model

The MOVRPTW problem extends the VRP problem by incorporating time windows for each customer, with the objective of minimizing both the total distance traveled by vehicles and the number of vehicles used. This problem considers constraints such as vehicle capacity and the specific time windows within which customers must be served.

The VRPTW can be represented as a directed graph denoted as $G = (V, A)$, where:

- V : Represents the set of cities where the clients are located plus the depot city, such that v_o represents the depot city and the set $N = \{v_1, v_2, \dots, v_n\}$ represents the client cities.
- $A = \{(v_i, v_j) \in A; v_i, v_j \in N\}$: The set of arcs between the cities.

To define MOVRPTW, we must first identify the relevant parameters and decision variables, as well as the constraints and objectives of the various.

5.3.1 Parameters

- N : The number of customer cities;
- K : The number of vehicles;
- (x_i, y_i) : The coordinates of the city v_i
- d_{ij} : The Euclidean distance between city v_i and city v_j , which is given by the following relation:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$
- q_i : The demand of goods at the customer v_i .
- s_i : The service time required to unload its demands at the customer v_i ;
- e_i : The earliest time of the customer's delivery v_i ;
- l_i : The latest time of the customer's delivery v_i ;
- t_{ij} : The travel time between city v_i and city v_j .

5.3.2 Decision variable

To mathematically model MOVRPTW, it is necessary to define the decision variables.

$$x_{ij}^k = \begin{cases} 1 & \text{If the vehicle } k \text{ travels from customer } i \text{ to customer } j \\ 0 & \text{Otherwise} \end{cases}$$

5.3.3 Constraints

The MOVRPTW constraints should be met as follows in order to serve all customers:

- (1) Each customer must be visited exactly once by exactly one vehicle. This constraint can be expressed as follows:

$$\sum_{k=1}^K \sum_{i=1}^N x_{ij}^k = 1 \quad \forall j \in \{1, \dots, N\}. \quad (5.1)$$

- (2) Each vehicle starts from a depot, and ends at a depot.

$$\sum_{i=1}^N x_{0i}^k = \sum_{i=1}^N x_{i0}^k = 1 \quad \forall k \in \{1, \dots, K\} \quad (5.2)$$

- (3) The same vehicle that visited a customer, leaves from that customer.

$$\sum_{j=1, j \neq i}^N x_{ji}^k - \sum_{j=1, j \neq i}^N x_{ij}^k = 0 \quad \forall i \in \{1, \dots, N\}, \forall k \in \{1, \dots, K\}. \quad (5.3)$$

- (4) The sum of the demands must not exceed the vehicle capacity Q . This constraint is expressed as follows:

$$\sum_{i=1}^N \sum_{j=1}^N q_i \times x_{ij}^k \leq Q \quad \forall k \in \{1, \dots, K\}. \quad (5.4)$$

- (5) Waiting is allowed when a vehicle arrives too soon at customer v_j after the service is finished at customer v_i , in other words, before e_j . The time at which service starts at customer v_j is defined as $b_j = \max\{e_j, a_j\}$, where $a_j = b_j + s_j + t_{ij}$ and the waiting time at customer v_j as $w_j = b_j - a_j$. This constraint is expressed as follows:

$$x_{ij}^k (b_i + w_i + s_i + t_{ij} - b_j) = 0 \quad \forall i, j \in \{1, \dots, N\}, \forall k \in \{1, \dots, K\}. \quad (5.5)$$

$$e_i \leq b_i + w_i \leq l_i \quad \forall i \in \{0, 1, \dots, N\}. \quad (5.6)$$

5.3.4 Objective functions

The objectives of MOVRPTW are based on the total travel distance of the tour and the number of vehicles. We formulate them as follows:

1. The first objective function is to minimize the total travel distance of all the vehicles :

$$\min F_1 = \sum_{k=1}^K \sum_{i=1}^N \sum_{j=1}^N d_{ij} \times x_{ij}^k \quad (5.7)$$

2. The second objective function is to minimize the number of vehicle (routes) given by:

$$\min F_2 = \sum_{k=1}^K \sum_{i=1}^N x_{0i}^k \quad (5.8)$$

5.4 The proposed algorithm

5.4.1 Motivation

Although ϵ -MOEA is widely used in multi-objective optimization problems, it has been observed that its convergence speed may be slower compared to other algorithms, particularly when the number of decision variables increases. This paper proposes a new variant of ϵ -MOEA, named VRPTW-MOEA, specifically designed for solving the MOVRPTW. The main objectives of this study are twofold: first, to introduce a novel mating strategy involving local search and multi-type crossover operators, which enables thorough exploration of the entire search space to discover high-quality solutions and significantly enhance the convergence speed of ϵ -MOEA. Second, to address the issue of losing good solutions once found in ϵ -MOEA, an external population called an archive is employed to store the non-dominated solutions based on ϵ -dominance. This archive plays a crucial role in multi-objective optimization.

5.4.2 Outlines of VRPTW-MOEA algorithm

This section provides an overview of the VRPTW-MOEA algorithm for solving the MOVRPTW. The pseudocode of VRPTW-MOEA is presented in Algorithm 6. In the initialization process (line 1), the parent population P consisting of N individuals is initialized using two methods: randomly and nearest neighbor heuristic. Subsequently, fitness values ($f_1(x^i), f_2(x^i)$) are evaluated for all solutions x^i in the parent population P (line 2). Non-dominated solutions from the parent population P are then stored in the external population A (line 3). In the evolutionary process, for each iteration i (ranging from 1 to It_{max}), two individuals (x^k and x^l) are selected using the selection operator to generate an offspring y through the multi-crossover operators (lines 4-5). The offspring y undergoes improvement using mutation operators with a probability P_m (lines 6-7). Subsequently, local search operators are applied to the newly created solution (line 8). The fitness values ($f_1(y), f_2(y)$) of the offspring y are evaluated (line 9). The parent population P is then updated using Algorithm 6 based on the offspring y (line 10). Additionally, the external archive population A is updated with y^* (line 11).

The number of non-dominated solutions in the external archive population A increases over iterations, potentially leading to increased algorithm complexity. To address this issue, the size of the external archive population A is controlled using ϵ -dominance, thereby limiting its size.

Algorithm 6 Pseudocode of VRPTW-MOEA**Input:**

N : the size of parent population.

p_c : the probability of crossover.

p_m : the probability of mutation.

$It e_{max}$: the number of iterations.

Output:

P : the parent population.

A : the external archive population .

Step I: Initialization Procedure

- 1: Generate an initial parent population $P = \{x^1, x^2, \dots, x^N\}$.
- 2: Evaluate the fitness values (f_1, f_2) of the parent population P .
- 3: Initialize the external archive population A by copy all non-dominated solutions of the parent population P .

Step II: Evolutionary Procedure

- 1: **for** $i \leftarrow 1$ to $It e_{max}$ **do**
- 2: Randomly select two individuals x^k and x^l using the selection operator.
- 3: Apply multi-crossover operators on x^k and x^l to generate y with probability p_c .
- 4: $r = random(0, 1)$.
- 5: **if** $r \leq p_m$ **then**
- 6: Apply mutation operators to improve y .
- 7: **end if**
- 8: Apply local search operators on y to generate y^* .
- 9: Evaluate the fitness values $(f_1(y^*), f_2(y^*))$ of the offspring y^* .
- 10: $P \leftarrow UPDATE_POPULATION(P, y^*)$.
- 11: $A \leftarrow UPDATE_ARCHIVE(A, y^*)$.
- 12: **end for**
- 13: **return** P, A .

5.4.2.1 Solution representation

Each solution in the VRPTW problem is represented as a chromosome, where the chromosome is encoded as a sequence S consisting of n customers. However, the sequence does not contain any route delimiters to indicate the start or end of routes. This sequence can be seen as a single vehicle's continuous tour with unlimited capacity, disregarding time windows. To obtain the best VRPTW solution from the sequence S , an optimal splitting technique called Split is applied.

The Split technique was initially developed by Prins (Prins, 2004) for the Capacitated Vehicle Routing Problem (CVRP) and later extended to address time windows. This approach involves computing the shortest route in an auxiliary graph H , which includes a dummy node 0 and n additional nodes representing the n customers. Each subsequence of customers $(S_i, S_{i+1}, \dots, S_j)$ that forms a feasible route is represented by a weighted arc $(i-1, j)$ in graph H . The weight assigned to this arc corresponds to the length of the route. To determine the shortest route from node 0 to node n in graph H , Bellman's algorithm for directed acyclic graphs is employed (Labadi et al., 2008).

5.4.2.2 Selection operator

The choice of the selection operator in VRPTW-MOEA plays a crucial role in determining the algorithm's performance. It is essential to select solutions in a manner that leads to Pareto optimal solutions close to the Pareto front. To achieve this objective, our approach utilizes two populations: the parent population P and the external archive population A . These populations are employed for selecting solutions that will generate high-quality offspring solutions. The selection operator is designed as follows:

1. We randomly select two solutions x_k and x_l from the population:
 - a) If the solution x_l dominates the solution x_k then x_l is selected as the first parent.
 - b) If the solution x_k dominates the solution x_l then x_k is selected as the first parent.
 - c) Else, we randomly select a solution x_l or x_k as the first parent.
2. We randomly select a solution from the external archive as the second parent.

5.4.2.3 Multi-type Crossover operators

The crossover operation plays a vital role in the VRPTW-MOEA algorithm as it is responsible for generating offspring from the selected parents, with a probability of p_c , while preserving the desirable characteristics of the parents. As mentioned earlier, we employ two types of crossover operations on the selected parents: Ordered Crossover and Best Cost Route Crossover. The procedure for performing these two crossover operations is presented in Algorithm 7. In each generation, the choice of crossover operation is determined based on the probability p_c . Specifically, if the randomly generated number is less than the probability p_c , the first type of crossover operation is utilized; otherwise, the second type of crossover operation is employed. By incorporating two types of crossover operations, we aim to extensively explore all regions of the search space in search of high-quality solutions.

Algorithm 7 Pseudocode of multi-crossover operators

Input:

parent 1: the first parent.

parent 2: the second parent.

p_c : the probability of crossover .

$U = \text{random}(0, 1)$.

if $U < p_c$ **then**

offspring \rightarrow BCRC (parent 1, parent 2).

else

offspring \rightarrow OX (parent 1, parent 2).

end if

Output: offspring

1. Best Cost Route Crossover (BCRC)

The Best Cost Route Crossover (BCRC) operator, originally introduced by (Ombuki et al., 2006), aims to simultaneously minimize the number of vehicles and the total travel distance while adhering to feasibility constraints. The steps involved in the BCRC operator are as follows:

- 1 Select two parents, denoted as P_1 and P_2 , using the selection operator described in subsection (5.4.2.2).
- 2 Randomly choose a route, denoted as r_i , from both parents' solutions.
- 3 Remove all customers associated with route r_1 from parent P_1 , and remove all customers associated with route r_2 from parent P_2 .
- 4 Reinsert the removed customers into the offspring solution as follows:
 - (a). For each customer associated with route r_1 (or r_2), calculate the cost of inserting that route into each position of parent P_2 (or P_1), and store the costs in an ordered list.
 - (b). Check the feasibility of insertion at each potential position.
 - (c). If there are no feasible insertion positions for the remaining route, create a new route.

By employing the BCRC operator, we aim to effectively manage the trade-off between reducing the number of vehicles and minimizing the total travel distance while ensuring the feasibility of the resulting offspring solutions.

5.4.2.4 Mutation operator

Within the VRPTW-MOEA algorithm, the inclusion of a mutation operator serves to enhance the diversity of the search space. This is achieved through the utilization of two mutation operators, namely reallocation and swap, which are applied to the offspring solutions with a probability denoted as p_m .

In the reallocation mutation operator, a random route is selected, and its customers are reallocated. The process is outlined in detail in Algorithm 8. Conversely, in the swap mutation operator, two routes are randomly chosen, and a customer is selected from each route. These selected customers then exchange positions, as illustrated in Figure 5.1.

Algorithm 8 Pseudocode of relocation mutation

Input: Offspring.

Output: New offspring.

- 1: Select a random route $R = (r_1, r_2, \dots, r_{|R|})$ of the offspring.
 - 2: Let $l_n = |R|$.
 - 3: Initialize an empty route $R_{new} = \emptyset$.
 - 4: Remove customer r_i with the shortest last time in R and put r_i in the first location of R_{new} ; $R_{new} = (r_i)$.
 - 5: Remove customer r_j with the longest last time in R and put r_j in the last location of R_{new} ; $R_{new} = (r_i, r_j)$.
 - 6: Put $s = i$.
 - 7: Find the nearest customer r_k to the customer r_s and goto step 8.
 - 8: Add customer r_k after the $s + 1$ th location in R_{new} .
 - 9: Let $s = s + 1$ and goto step 10.
 - 10: If R_{new} length equals l_n , terminate, else goto step 7.
 - 11: Replace R in the offspring with R_{new} .
 - 12: New offspring \leftarrow offspring.
-

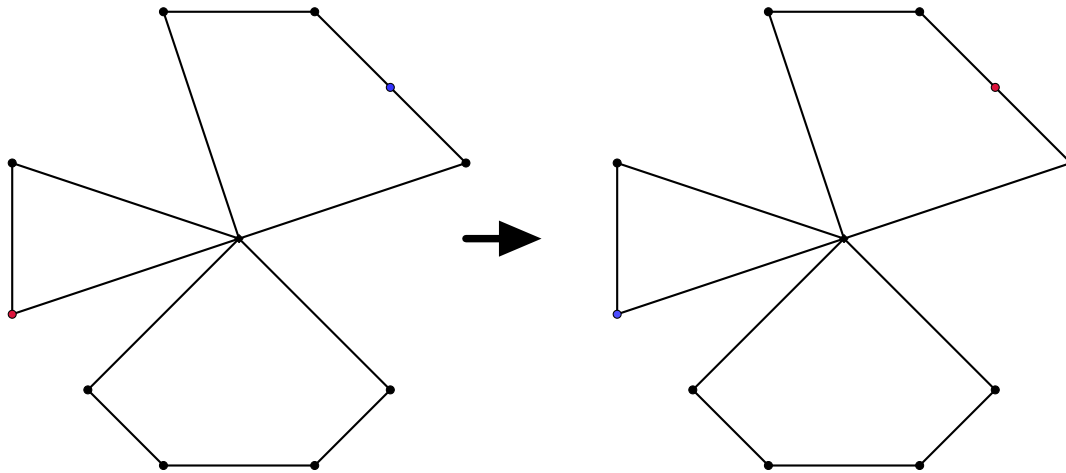


Figure 5.1 – An example of swap mutation operator.

5.4.2.5 Local search

The enhanced algorithm incorporates two types of neighborhood structures, namely Reinsert Route (RR) and Reinsert Customers (RC), into its local search phase. These neighborhood structures are designed to improve the efficiency of the algorithm in finding optimal route solutions for the MOVRPTW problem.

As the VRPTW-MOEA algorithm operates iteratively, only one of the local search methods is randomly selected and applied in each iteration. This random selection ensures diversity and provides an opportunity for both RR and RC methods to contribute to the search process. By leveraging these neighborhood structures, the algorithm becomes more effective and capable of exploring different solution spaces, ultimately enhancing its performance in solving the MOVRPTW problem.

Reinsert Route (RR): The RR operation involves selecting a route based on the ratio of travel distance to the number of customers. Specifically, the algorithm chooses a route with a larger travel distance and fewer customers. This selected route is then removed, aiming to minimize the number of vehicles involved in the solution. This process is visually depicted in Figure 5.2. The customers originally assigned to this route are referred to as "free customers."

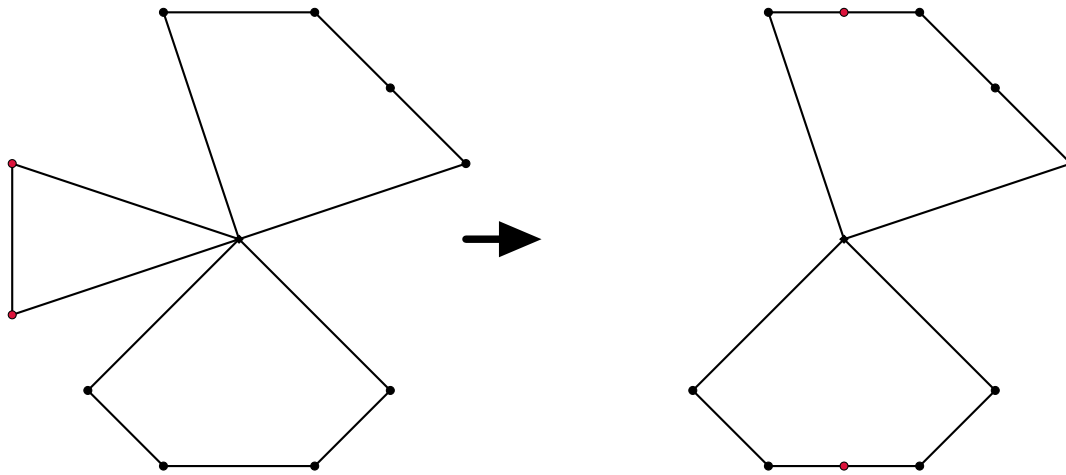


Figure 5.2 – An example of the reinsert route (RR).

Reinsert Customers (RC): The RC operation involves randomly selecting a certain number of customers from the existing routes. These selected customers are then removed from their respective routes, as illustrated in Figure 5.3. The customers that have been removed in this process are referred to as "free customers."

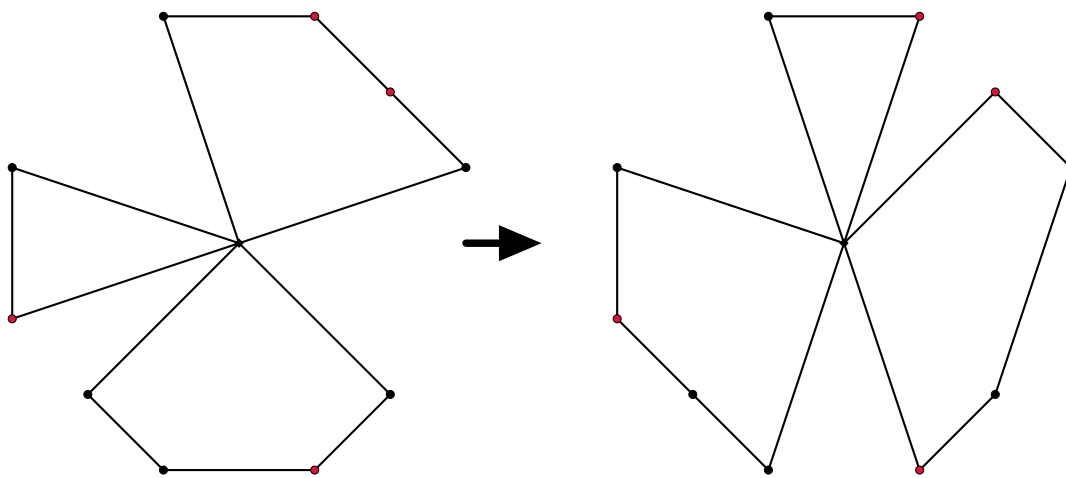


Figure 5.3 – An example of the reinsert customers (RC).

In both methods, free customers are added in different positions in the routes, where we insert these customers in the optimal position in an optimal route by depending on the minimum travel distance.

5.4.2.6 Updating the parent population

The parent population is updated with the offspring solutions that are generated. Specifically, for each newly created solution (offspring) y , it is considered for inclusion in the current population P . The dominance concept is applied to compare the new solution y with all existing solutions in P to determine its admission (Deb et al., 2005). If y^* is found

to dominate any solution in P , it will replace the dominated solution, thus updating the parent population.

5.5 Experimental study

5.5.1 Test problems

The proposed algorithm was tested on common problems found in Solomon (1987), including 56 data sets of 100 customers. The customer locations are distributed within a $[0, 100]^2$ square. The Solomon instances are one of the most known data instances for VRPTW¹. These instances are divided into six classes, as follows:

1. **R1 Class:** This class includes instances with customers uniformly distributed in a square region. The time windows for customers are randomly generated.
2. **R2 Class:** Similar to the R1 class, this class also has customers uniformly distributed in a square region, but the time windows for customers are defined as a function of their positions.
3. **C1 Class:** Instances in this class have customers clustered together in a square region. The time windows for customers are randomly generated.
4. **C2 Class:** Similar to the C1 class, this class also has customers clustered together in a square region, but the time windows for customers are defined as a function of their positions.
5. **RC1 Class:** This class combines the characteristics of the R1 and C1 classes. It includes instances with customers distributed both uniformly and in clusters, with randomly generated time windows.
6. **RC2 Class:** Similar to the RC1 class, this class combines the characteristics of the R2 and C2 classes. It includes instances with customers distributed both uniformly and in clusters, with time windows defined as a function of their positions.

Each class represents a different distribution and configuration of customers, as well as varying time window characteristics.

5.5.2 Parameter settings

The parameter setting significantly impacts the VRPTW-MOEA algorithm's capacity to generate excellent solutions. The settings are established in Table 5.1 based on a large number of tests.

5.5.3 Comparison with its variants

Each test problem is subjected to ten repetitions of VRPTW-MOEA and its variants. The best results are highlighted in bold. This section aims to compare VRPTW-MOEA with its variants and assess the effectiveness of the algorithm's components based on the average number of vehicles (NV) and the average total travel distance (TD) across different instances.

1. <http://www.bernabe.dorronsoro.es/vrp/>

Table 5.1 – Parameters of algorithms.

Parameter	Tested value	Best value
Population size	{50, 80, 100}	100
Generation	{2000, 2500, 3000}	3000
Mutation probability	{0.15, 0.2}	0.2
ϵ	{0.01, 0.015, 0.001}	0.001
Number of runs	/	10

Variant-1: VRPTW-MOEA without Best Cost Route Crossover (BCRC).

Variant-2: VRPTW-MOEA without Local search.

Table 5.2 indicates that out of 56 test instances, VRPTW-MOEA obtains 54 of the lowest HV values, while its variants, specifically variant 1, achieve 36 HV values compared to variant 2, which obtains the 20 best HV values in comparison to variant 1.

These findings demonstrate that VRPTW-MOEA exhibits better convergence and diversity than its variants. The results also highlight the significant impact of designing an algorithm with multi-type crossover operators and local search, leading to effective performance.

Table 5.2 – Comparison of VRPTW-MOEA and its variants on the obtained non-dominated solutions using hypervolume indicator

Instance	VRPTW-MOEA	Variant 1	Variant 2	Instance	VRPTW-MOEA	Variant 1	Variant 2
C101	0.2720	0.2720	0.2720	C201	0.2617	0.2617	0.2617
C102	0.2720	0.2720	0.2720	C202	0.2617	0.2617	0.2617
C103	0.2725	0.2725	0.2633	C203	0.2620	0.2620	0.2590
C104	0.2744	0.2744	0.2500	C204	0.2625	0.2625	0.2500
C105	0.2720	0.2720	0.2720	C205	0.2639	0.2639	0.2631
C106	0.2720	0.2720	0.2720	C206	0.2642	0.2642	0.2634
C107	0.2720	0.2720	0.2720	C207	0.2644	0.2644	0.2644
C108	0.2720	0.2720	0.2712	C208	0.2644	0.2644	0.2634
C109	0.2720	0.2720	0.2523				
Average	0.2723	0.2723	0.2663	Average	0.2631	0.2631	0.2608
R101	0.2912	0.2574	0.2752	R201	0.4800	0.3691	0.4673
R102	0.4135	0.3984	0.3573	R202	0.5747	0.4484	0.5528
R103	0.6194	0.6028	0.6087	R203	0.7951	0.7746	0.7824
R104	0.9198	0.8991	0.8790	R204	0.9219	0.9145	0.9053
R105	0.5519	0.4901	0.4951	R205	0.7312	0.6169	0.7193
R106	0.6508	0.6341	0.6169	R206	0.8007	0.7831	0.7874
R107	0.8197	0.7893	0.7976	R207	0.8577	0.8640	0.8382
R108	0.9285	0.9252	0.9050	R208	1.1043	1.0722	1.0804
R109	0.7199	0.6948	0.6889	R209	0.8104	0.7515	0.7935
R110	0.8127	0.7485	0.7523	R210	0.7759	0.8555	0.7449
R111	0.8259	0.8116	0.7229	R211	0.9052	0.8709	0.8776
R112	0.9162	0.9219	0.9099				
Average	0.7058	0.6811	0.6674	Average	0.7961	0.7564	0.7772
RC101	0.3051	0.2500	0.3005	RC201	0.6023	0.6032	0.5987
RC102	0.3953	0.3538	0.3835	RC202	0.7399	0.7330	0.7202
RC103	0.6074	0.5348	0.5131	RC203	0.9586	0.8549	0.9416
RC104	0.7366	0.6620	0.7035	RC204	1.0857	1.0775	1.0742
RC105	0.3667	0.3228	0.2815	RC205	0.6719	0.5525	0.5918
RC106	0.5113	0.3988	0.4538	RC206	0.7523	0.7348	0.7524
RC107	0.6354	0.5630	0.5513	RC207	0.8306	0.8014	0.8003
RC108	0.6837	0.6607	0.6544	RC208	1.0836	1.0596	1.0714
Average	0.5302	0.4682	0.4802	Average	0.8406	0.8021	0.8188

5.5.4 Comparison with previous multiobjective algorithms

This section presents a comparison between VRPTW-MOEA and other multiobjective algorithms. The results of each algorithm are evaluated in two aspects. The first aspect focuses on the minimal number of vehicles, where we select the best solution with the fewest number of vehicles from the Pareto optimal solutions set. The second aspect considers the minimal total distance, where we choose the best solution with the lowest total travel distance from the Pareto optimal solutions set.

From Table 5.3, it is evident that the results obtained by VRPTW-MOEA outperform all the results from (Ghoseiri & Ghannadpour, 2010). Additionally, in the R2 and RC2 categories, VRPTW-MOEA outperforms (Khoo & Mohammad, 2021), (Moradi, 2020), and (Zhang et al., 2018) in terms of the minimal number of vehicles. Regarding the minimal total distance, VRPTW-MOEA outperforms (Qi et al., 2015) in terms of the average number of vehicles in R1 and R2. For instances C1 and C2, VRPTW-MOEA achieves similar results to the previously published studies.

Table 5.3 – Average number of vehicles and average total travel distance of VRPTW-MOEA and five multiobjective approaches

Approach	C1	C2	R1	R2	RC1	RC2
VRPTW-MOEA (minimal number of vehicles)	10.00	3.00	12.67	3.09	12.25	3.625
	828.38	589.86	1220.51	944.75	1384.73	1126.13
VRPTW-MOEA (minimal total distance)	10.00	3.00	13.08	4.18	12.875	5.75
	828.38	589.86	1206.18	900.51	1368.10	1037.56
HRRGA (Khoo & Mohammad, 2021) (minimal number of vehicles)	10.00	3.00	12.25	3.09	11.88	4
	828.38	589.86	1211.8	966.24	1360.19	1055.53
HRRGA (Khoo & Mohammad, 2021) (minimal total distance)	10.00	3.00	13.25	5.27	12.75	6.25
	828.38	589.86	1179.35	877.66	1338.56	1004.7
MODLEM (Moradi, 2020)	10.00	3.00	11.9	4.55	11.50	4
	828.38	589.86	1210.4	916.95	1384.17	1074.67
ESS-PSO (Zhang et al., 2018) (minimal total distance)	10.00	3.00	13.3	5	12.9	6.13
	828.38	589.86	1180.22	879.86	1343.28	1004.54
M-MOEA/D (Qi et al., 2015) (minimal number of vehicles)	10.00	3.00	12.42	3.09	11.88	3.38
	828.38	589.86	1216.40	924.18	1390.35	1119.95
M-MOEA/D (Qi et al., 2015) (minimal total distance)	10.00	3.00	13.17	4.55	12.88	5.13
	828.38	589.86	1192.35	889.66	1356.76	1026.81
MOGP (Ghoseiri & Ghannadpour, 2010) (minimal number of vehicles)	10.00	3.00	12.92	3.45	12.75	3.75
	828.38	591.49	1228.60	1033.53	1392.09	1162.40
MOGP (Ghoseiri & Ghannadpour, 2010) (minimal total distance)	10.00	3.00	13.50	3.82	13.25	4.00
	828.38	591.49	1217.03	1049.62	1384.3	1157.41

5.6 Conclusion

In this chapter, we introduced VRPTW-MOEA, a novel variant of a multiobjective evolutionary algorithm designed for solving the MOVRPTW. Dealing with multiobjective optimization problems presents challenges in maintaining a balance between convergence and diversity. To address this, we enhanced the ϵ -MOEA by incorporating a multi-type crossover operator (Ordered Crossover and Best Cost Route Crossover), integrating local search techniques, and utilizing an external archive based on adaptive ϵ -dominance. These improvements significantly accelerated convergence, preventing the loss of promising solutions once discovered. The effectiveness of VRPTW-MOEA was evaluated using the well-known Solomon's 56 data sets. The experimental results demonstrated that VRPTW-MOEA outperformed its variants, five other multiobjective algorithms, and even the best-known

solutions in solving the MOVRPTW. This highlights the efficiency and efficacy of VRPTW-MOEA as a solution approach for the MOVRPTW.

6

EAG-NSGA-II algorithm for Multi-Depot Green Vehicle Routing Problem

6.1 Introduction

Managing a green supply chain involves various challenges, and one of the significant problems in this domain is the green vehicle routing problem (GVRP). GVRP entails creating routes to deliver specific quantities of products to consumers whose consumption corresponds to predetermined amounts. To meet customer demands, a fleet of vehicles with identical capacities is available at the depot. Each vehicle starts from the depot, serves customers individually, and returns to the same depot. However, the multi-depot green vehicle routing problem (MDGVRP) introduces the concept of multiple depots, making it a more intricate variation compared to other vehicle routing problems.

In this chapter, we introduce a novel variant of the NSGA-II algorithm called EAG-NSGA-II, specifically designed to address the MDGVRP. Our approach incorporates two key contributions. Firstly, we integrate a local search strategy to enhance convergence speed, thereby improving the efficiency of the algorithm. Secondly, we utilize an external archive based on adaptive epsilon dominance to strike a balance between convergence and diversity, ensuring that the algorithm maintains a diverse set of high-quality solutions throughout the optimization process. These two strategies collectively enhance the performance of EAG-NSGA-II in solving the MDGVRP.

6.2 Literature review

In the existing literature, numerous studies have focused on addressing the multi-depot vehicle routing problem (MDVRP). The initial description of MDVRP was provided by (Cassidy & Bennett, 1972), presenting it as an extension of the conventional vehicle routing problem (VRP) by considering the existence of multiple depots (Yu et al., 2011). The primary objective of MDVRP research is to propose and develop new techniques and algorithms to tackle this problem. According to (Montoya-Torres et al., 2015), most researchers prefer to employ heuristics or meta-heuristics to solve MDVRP. For example, (Vidal et al., 2012) utilized a hybrid genetic algorithm, while (Yu et al., 2011) transformed MDVRP into a

single-depot VRP (SVRP) by introducing a virtual depot and using an enhanced Ant Colony Optimization (ACO) for solving SVRP. (Escobar et al., 2014) employed a hybrid granular tabu search technique to address MDVRP. Similarly, (Kaabachi et al., 2017), (Jabir et al., 2017), (Li et al., 2018) and (Wang et al., 2019) focused on the multi-depot green vehicle routing problem (MDGVRP). However, their MDGVRP models were derived from the Pollution Routing Problem (PRP) and did not consider the tank capacity of vehicles. In this paper, our contribution lies in the development of a new MDGVRP model that incorporates a constraint on the tank capacity of alternative fuel vehicles (AFV). We formulate a mathematical model for the proposed MDGVRP and design an algorithm for solving it. (Wang et al., 2019) addressed MDVRP with pickups and deliveries by classifying clients as borderline and non-borderline. Building on their concept, we propose a Partition-Based Algorithm (PBA) and further suggest a Two-stage Ant Colony System (TSACS) to find superior solutions for the problem.

6.3 Mathematical model

The MDGVRP mathematical model is presented in this section. MDGVRP is represented by a complete graph G consisting of a set of nodes representing customers and arcs on the one hand. It is technically represented as a linear integer program.

The graph G is defined by the formula $G = (V, E)$, where:

- $V = N \cup D$ Set of nodes, where:
 - $N = \{1, 2, \dots, n\}$ Set of customers.
 - $D = \{n + 1, n + 2, \dots, n + m\}$ Set of depot.
- $E = \{(i, j) : i, j \in V, i \neq j\}$ Set of arcs.

To define MDGVRP, we must first identify the relevant variables, as well as the MDGVRP's varied constraints and objectives.

6.3.1 Decision variable

To mathematically model MDGVRP, it is necessary to define the decision variables.

$$x_{ij}^k = \begin{cases} 1 & \text{If the vehicle } k \text{ travels from node } i \text{ to node } j \\ 0 & \text{Otherwise} \end{cases}$$

Where K is the number of vehicles. The travel distance between customers i and j is d_{ij} . The conversion factor for carbon emissions is CCF .

6.3.2 Constraints

The MDGVRP constraints should be met as follows in order to serve all customers:

- (1) Each customer must be visited exactly once by exactly one vehicle. This constraint can be expressed as follows:

$$\sum_{k=1}^K \sum_{i=1}^{n+m} x_{ij}^k = 1 \quad \forall i \in \{1, \dots, n + m\} \quad (6.1)$$

(2) Each vehicle starts from a depot, and ends at a depot.

$$\sum_{i=n+1}^{n+m} \sum_{j=1}^n x_{ij}^k \leq 1 \quad \forall k \in \{1, \dots, K\} \quad (6.2)$$

(3) The same vehicle that visited a customer, leaves from that customer.

$$\sum_{i=1}^{n+m} x_{ih}^k - \sum_{j=1}^{n+m} x_{hj}^k = 0 \quad \forall h \in \{1, \dots, n+m\}, \forall k \in \{1, \dots, K\} \quad (6.3)$$

(4) The sum of the demands must not exceed the vehicle capacity Q . This constraint is expressed as follows:

$$\sum_{i=1}^{n+m} \sum_{j=1}^{n+m} q_i \times x_{ij}^k \leq Q \quad \forall k \in \{1, \dots, K\} \quad (6.4)$$

Where q_i is the demand of customer i .

6.3.3 Objective functions

The objectives of MDGVRP are based on the total carbon emissions of the tour and the number of vehicles. We formulate them as follows:

1. The first objective function is to minimize the total sum of the carbon emissions produced by all the vehicles :

$$\min F_1 = \sum_{k=1}^K \sum_{i=1}^{n+m} \sum_{j=1}^{n+m} d_{ij} x_{ij}^k \times CCF \quad (6.5)$$

2. The second objective function is to minimize the number of "used vehicle" routes given by:

$$\min F_2 = \sum_{k=1}^K \sum_{i=1+n}^{n+m} \sum_{i=1}^n x_{ij}^k \quad (6.6)$$

6.4 The proposed algorithm

6.4.1 Outlines of EAG-NSGA-II algorithm

In this section, we propose a new variant of NSGA-II algorithm to solve the MDGVRP problem that we have named EAG-NSGA-II, referring to the adopted combination of the following: NSGA-II algorithm, local search (LS) and external archive. The procedure of the proposed algorithm is illustrated in Algorithm 9.

Algorithm 9 Pseudocode of EAG-NSGA-II algorithm for MDGVRP**Input:**

N : the size of parent population.

Step I: Initialization Procedure

- 1: Generate an initial parent population P .
- 2: Evaluate the fitness values (f_1, f_2) of the parent population P .
- 3: Initialize the external archive population A by copy all non-dominated solutions of the parent population P .

Step II: Evolutionary Procedure

- 1: **while** termination criterion is not reached **do**
- 2: $Q \leftarrow$ Tournament_Selection (P).
- 3: $Q' \leftarrow$ Crossover_Operators (Q).
- 4: $Q^* \leftarrow$ Local_Search (Q').
- 5: $RL =$ Combination ($P \cup Q^*$).
- 6: Sort the non-dominated solution of $RL = \{Fr_1, Fr_2, Fr_3, \dots\}$.
- 7: Create a current population $P_{current} = \emptyset$, and $i = 1$.
- 8: **while** $|P_{current}| + |Fr_i| < N$ **do**
- 9: $P_{current} = P_{current} \cup Fr_i$.
- 10: $i = i + 1$.
- 11: **end while**
- 12: Sort the front Fr_i according to the crowding distances.
- 13: $P = P_{current} \cup Fr_i(1 : N - |P_{current}|)$.
- 14: $A \leftarrow$ UPDATE_ARCHIVE (A, Q^*).
- 15: **end while**
- 16: **return** P, A .

Like all population-based algorithms, the first step of EAG-NSGA-II is to generate an initial population P_0 of $|P_0|$ feasible solutions. At each generation, the solutions of the new population Q are generated by a crossover operator, and then improved by a local search to explore new regions in the search space. Then, the solutions of the new population Q are used to update the external archive.

Finally, they are gathered into a set $RL = P \cup Q$, which is sorted according to the dominance principle: The set RL is divided into several separate classes Fr_j as follows: All non-dominated individuals of RL belong to the set Fr_1 ; next, all the non-dominated members of $RL \cup Fr_1$ are placed in the set Fr_2 , and so on until the entire population is sorted.

When the entire population is sorted, the next population $P(t+1)$ is filled by the solutions of the non-dominated subsets of $RL(t)$, one by one, beginning with the first edge. To select the solutions that will survive from the front, only a subset of which can be inserted in the next population, a measure of the density of solutions in the criteria space termed crowding distance is utilized.

6.4.2 Solution representation

The solution representation involves three essential phases. In the initial phase, each customer is assigned to a specific depot. The second phase, known as customer grouping, involves dividing customers assigned to the same depot into different groups. Finally, in the third phase, customer service is organized within each group.

6.4.3 Crossover operator

To preserve the desirable characteristics of the selected parents, the crossover operators are responsible for generating offspring from these parents with a probability of p_c . The Best Cost Route Crossover (BCRC) [Ombuki et al. \(2006\)](#) is utilized as a crossover operator to reduce both the total carbon emissions generated by all vehicles and the number of vehicles simultaneously while respecting the feasibility constraints.

6.4.4 Local search (LS)

In order to improve the offspring solutions, two types of neighborhood structures, interdepot and intradepot neighborhoods, are employed in the local search (LS) phase. The choice of neighborhood strategy to enhance the selected solution is determined using a roulette wheel selection. Interdepot neighborhood approaches are constructed using several fundamental operators, including swap, shift, 2-opt*, Or-opt, and 2-opt. Intradepot neighborhood methods consist of a cross operator, an inversion operator, and a 2-opt* operator.

6.4.5 Updating the archive

To determine if a new solution should be added to the archive (A), we compare it to all existing solutions in A using the ϵ -dominance concept. If the new solution dominates any solution in the archive, the new solution replaces the dominated one and the latter is removed from A . If the new solution is non-dominated by any solution in the archive and the archive solution is non-dominated by the new solution, the new solution is accepted and added to the archive.

6.5 Experimental results and discussion

In order to assess the performance of the proposed algorithm, 11 benchmark datasets chosen of [Cordeau et al., 2001](#) are used in the experiments. Table 6.1 lists the characteristics of the datasets, including the number of customers (C), depots (D), maximum vehicle load (Q), and longest duration of each route (L).

Table 6.1 – Characteristics of MDGVRP instances.

Instances	C	D	Q	L
P01	50	4	80	16
P02	50	4	160	8
P03	75	5	140	15
P04	100	2	100	16
P05	100	2	200	10
P06	100	3	100	18
P07	100	4	100	16
P012	80	2	60	10
P015	160	4	60	20
P018	240	6	60	30
P021	360	9	60	

The proposed algorithm is implemented using the programming language on MATLAB R2014a under 64-bit Windows 10. The tests are carried out on a laptop equipped with Intel 2 GHz processor and 8GB RAM. For each instance, we performed 10 independent runs. The parameters of the EAG-NSGA-II algorithm were set as follows: the size of the population $N = 50$. The number of generations is set to 500. A vehicle changes its route with a probability (i.e. the probability of rotation) $p_c = 0.9$, and the parameter value for converting carbon emissions is 0.2.

In Figure 6.1, we plot the non-dominated solutions for all datasets with different objectives obtained by EAG-NSGA-II and the three algorithms NSGA-II, SPEA-II, and MOEA/D to qualitatively illustrate the obtained results used for the comparison. These numbers indicate that our method EAG-NSGA-II is among the most competitive ones, having outperformed the other three in terms of convergence and diversity across all datasets with various objectives.

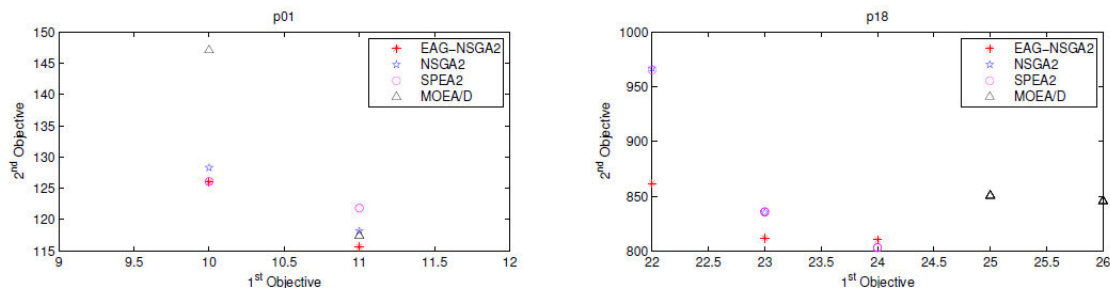
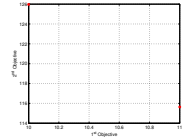
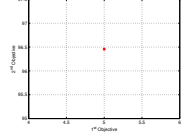
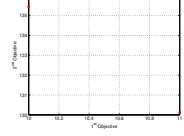
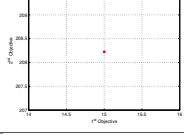
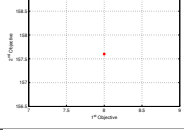
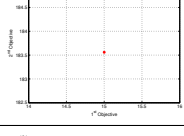
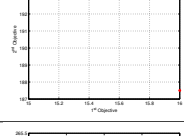
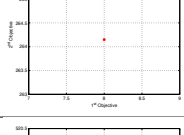
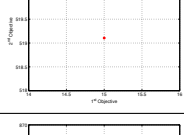
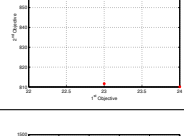
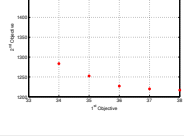


Figure 6.1 – Pareto set obtained by the multi-objective algorithms on some instances.

The computational outcomes for the instances given by [Cordeau et al. \(1997\)](#) are shown in Table 6.2. In presenting the Pareto-optimal set-based results, we offered the most effective solution for each issue (taking into account both the minimization of CO2 emission and the number of vehicles). In some instances, where neither one dominates the other (such as issues 1 and 3, 18, and 21), we have two or more solutions for each instance. The

situations issue where our approach performs better than the best-known results in the literature by having fewer vehicles or having the same number of vehicles is indicated by the boldface values in the designated columns.

Table 6.2 – Results of computations for the instances used by [Cordeau et al. \(1997\)](#)

Instance	N	M	Objective 1, Total number of vehicles	Objective 2, CO2 emission [kg]	Pareto's Front
p01	50	4	11 10	115.64 125.97	
p02	50	4	5	96.46	
p03	75	5	11 10	130.12 135.45	
p04	100	2	15	208.22	
p05	100	2	8	157.60	
p06	100	3	15	183.56	
p07	100	4	16 15	187.51 193.72	
p12	80	2	8	264.15	
p15	160	4	15	519.11	
p18	240	6	24 23 22	810.11 811.66 861.52	
p21	360	9	38 37 36 35 34 33	1216.76 1219.86 1226.96 1252.55 1283.54 1486.46	

6.6 Conclusions

In this chapter, we propose a new variant of NSGA-II called EAG-NSGA-II, designed specifically to tackle the MDGVRP. Our approach incorporates a local search to greatly enhance convergence speed and integrates an external archive based on adaptive ϵ -dominance to achieve a favorable trade-off between convergence and diversity. We evaluate the performance of EAG-NSGA-II using the well-known Cordeau data sets, consisting of eleven instances. Finally, we present experimental results that demonstrate the effectiveness of EAG-NSGA-II and discuss the impact of its parameters on its performance.

7

General conclusion

In the domain of supply chain management (SCM), the primary focus is on optimizing the supply chain operations while considering multiple conflicting objectives. These objectives encompass various aspects such as cost reduction, lead time minimization, enhanced customer service, inventory optimization, sustainability improvement, and overall supply chain performance enhancement. However, addressing one objective often leads to compromises in other areas. For instance, reducing costs may result in longer delivery times or compromised product quality, while prioritizing customer service may increase costs. Consequently, decision-makers face the challenge of finding a balance and identifying trade-offs among these objectives to achieve an optimal solution.

In this thesis, we have introduced novel multi-objective optimization methods to address three variants of the SCM problem. More specifically, we proposed three algorithms: the G-MOEA/D-SA algorithm for ambulance dispatching and relocation problems, the VRPTW-MOEA algorithm for the vehicle routing problem with time windows, and the EAG-NSGA-II algorithm for the multi-depot green vehicle routing problem.

Chapter 1 provided an overview of the thesis, while Chapter 2 delved into the combinatorial optimization problems, their theoretical complexities, and the class of NP-hard problems, which were fundamental to this project. Moreover, we introduced basic concepts related to multi-objective optimization problems and explored relevant approaches employing evolutionary algorithms. Various algorithms, particularly Multi-Objective Evolutionary Algorithms (MOEAs), widely recognized in the literature and utilized in Chapters 4 and 5, were presented. This chapter concluded with quality metrics for evaluating the performance of multi-objective evolutionary algorithms, emphasizing their advantages. Chapter 3 was dedicated to the supply chain management problem and its variants, which formed the basis of the studies conducted in this thesis.

After that, we recalled the definitions and basic notions about the multi-objective optimization problems, the supply chain management problem and its variants, and evolutionary algorithms; which are critical to understand the scope of the present thesis. We have presented two research works developed in this thesis project:

1. In Chapter 4, we presented an improved MOEA/D algorithm for real-time ADRP, called G-MOEA/D-SA. The proposed algorithm combines the standard version of the MOEA/D with SA and the external archive. In addition, a brief explanation was given as we looked for possible improvements, which opens up several directions to explore as

prospects for future work.

2. In Chapter 5, we presented a new variant of MOEA, namely VRPTW-MOEA for the vehicle routing problem with time windows. The proposed algorithm essentially consists of an adapted local search of ϵ -MOEA, aiming to maintain a mechanism of convergence and diversity when dealing with multi-objective optimization problems. In addition, we applied a multi-type crossover operators to accelerate the convergence significantly, and also employed the external archive based on adaptive ϵ -dominance to prevent the loss of satisfactory solutions once they are found. Experimental results demonstrated that VRPTW-MOEA was effective in solving VRPTW when compared with its variants, the five most well-known MOEAs, and the best-known solutions.
3. In Chapter 6, we presented a new variant of NSGA-II, namely EAG-NSGA-II for the multi-depot green vehicle routing problem (MDGVPR). We have proposed an initialization algorithm that generates only valid conformations for the initial population of EAG-NSGA-II. This algorithm eliminates reverse moves during solution construction. EAG-NSGA-II consists of using the local search algorithm and the external archive to explore the search space more efficiently. According to our experimental results, EAG-NSGA-II can find the best known solutions and is more efficient than other existing algorithms in terms of stability. In terms of future applications, EAG-NSGA-II can be used to solve other optimization problems in the context of combinatorial optimization.

Bibliography

- Aboueljinane, L., Sahin, E., & Jemai, Z. (2013). A review on simulation models applied to emergency medical service operations. *Computers & Industrial Engineering*, 66, 734–750.
- Alaya, I., Solnon, C., & Ghedira, K. (2007). Ant colony optimization for multi-objective optimization problems. In *19th IEEE international conference on tools with artificial intelligence (ICTAI 2007)* (pp. 450–457). IEEE volume 1.
- Alvarenga, G. B., Mateus, G. R., & De Tomi, G. (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34, 1561–1584.
- Anagnostopoulos, K. P., & Mamanis, G. (2011). The mean–variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms. *Expert Systems with Applications*, 38, 14208–14217.
- Andersson, T., & Värbrand, P. (2016). Decision support tools for ambulance dispatch and relocation. In *Operational Research for Emergency Planning in Healthcare: Volume 1* (pp. 36–51). Springer.
- Armananzas, R., & Lozano, J. A. (2005). A multiobjective approach to the portfolio optimization problem. In *2005 IEEE Congress on Evolutionary Computation* (pp. 1388–1395). IEEE volume 2.
- Badeau, P., Guertin, F., Gendreau, M., Potvin, J.-Y., & Taillard, E. (1997). A parallel tabu search heuristic for the vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 5, 109–122.
- Bader, J., & Zitzler, E. (2011). Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19, 45–76.
- Baldacci, R., Mingozzi, A., & Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218, 1–6.
- Bard, J. F., Kontoravdis, G., & Yu, G. (2002). A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36, 250–269.
- Bélangier, V., Lanzarone, E., Ruiz, A., & Soriano, P. (2015). *The ambulance relocation and dispatching problem*. CIRRELT Montréal, QC, Canada.

- Belenguer, J.-M., Benavent, E., Prins, C., Prodhon, C., & Calvo, R. W. (2011). A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Research*, 38, 931–941.
- Bellman, R. (1966). Dynamic programming. *Science*, 153, 34–37.
- Bendimerad, L. S., Houacine, N. A., & Drias, H. (2021). Swarm intelligent approaches for ambulance dispatching and emergency calls covering: Application to covid-19 spread in saudi arabia. In *International Conference on Soft Computing and Pattern Recognition* (pp. 617–626). Springer.
- Bent, R., & Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38, 515–530.
- Berger, J., Barkaoui, M., & Bräysy, O. (2003). A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR: Information Systems and Operational Research*, 41, 179–194.
- Beume, N., Naujoks, B., & Emmerich, M. (2007). Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181, 1653–1669.
- Bi, X., Han, Z., & Tang, W. K. (2017). Evolutionary multi-objective optimization for multi-depot vehicle routing in logistics. *International journal of computational intelligence systems*, 10, 1337–1344.
- Billhardt, H., Lujak, M., Sánchez-Brunete, V., Fernández, A., & Ossowski, S. (2014). Dynamic coordination of ambulances for emergency medical assistance services. *Knowledge-Based Systems*, 70, 268–280.
- Bin Shalan, S. A., & Ykhlef, M. (2015). Solving multi-objective portfolio optimization problem for saudi arabia stock market using hybrid clonal selection and particle swarm optimization. *Arabian Journal for Science and Engineering*, 40, 2407–2421.
- Booker, L. B., Goldberg, D. E., & Holland, J. H. (1989). Classifier systems and genetic algorithms. *Artificial intelligence*, 40, 235–282.
- Boyd, S., & Mattingley, J. (2007). Branch and bound methods. *Notes for EE364b, Stanford University, 2006*, 07.
- Brandão, J. (2018). Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows. *Computers & Industrial Engineering*, 120, 146–159.
- Branke, J., Scheckenbach, B., Stein, M., Deb, K., & Schmeck, H. (2009). Portfolio optimization with an envelope-based multi-objective evolutionary algorithm. *European Journal of Operational Research*, 199, 684–693.
- Bräysy, O., & Gendreau, M. (2005). Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation science*, 39, 119–139.
- Brockhoff, D., Wagner, T., & Trautmann, H. (2015). 2 indicator-based multiobjective search. *Evolutionary Computation*, 23, 369–395.

- Carvalho, A., Captivo, M., & Marques, I. (2020). Integrating the ambulance dispatching and relocation problems to maximize system's preparedness. *European Journal of Operational Research*, 283, 1064–1080.
- Cassidy, P., & Bennett, H. (1972). Tramp—a multi-depot vehicle scheduling system. *Journal of the Operational Research Society*, 23, 151–163.
- Castro, O. R., Santana, R., Lozano, J. A., & Pozo, A. (2017). Combining cma-es and moea/dd for many-objective optimization. In *2017 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1451–1458). IEEE.
- Chang, P. C., Chen, S. H., Zhang, Q., & Lin, J. L. (2008). Moea/d for flowshop scheduling problems. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)* (pp. 1433–1438). IEEE.
- Chen, B., Lin, Y., Zeng, W., Xu, H., & Zhang, D. (2017). The mean-variance cardinality constrained portfolio optimization problem using a local search-based multi-objective evolutionary algorithm. *Applied Intelligence*, 47, 505–525.
- Chen, C.-H., & Ting, C.-J. (2005). A hybrid ant colony system for vehicle routing problem with time windows. *Journal of the Eastern Asia Society for Transportation Studies*, 6, 2822–2836.
- Chen, M.-R., Weng, J., & Li, X. (2009). Multiobjective extremal optimization for portfolio optimization problem. In *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems* (pp. 552–556). IEEE volume 1.
- Chen, Y., & Zhou, A. (2022). Multiobjective portfolio optimization via pareto front evolution. *Complex & Intelligent Systems*, (pp. 1–17).
- Chiam, S. C., Tan, K. C., & Al Mamum, A. (2008). Evolutionary multi-objective portfolio optimization in practical context. *International Journal of Automation and Computing*, 5, 67–80.
- Chiang, T.-C., & Hsu, W.-H. (2014). A knowledge-based evolutionary algorithm for the multiobjective vehicle routing problem with time windows. *Computers & Operations Research*, 45, 25–37.
- Chiang, W.-C., & Russell, R. A. (1996). Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, 63, 3–27.
- Chiang, W.-C., & Russell, R. A. (1997). A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on computing*, 9, 417–430.
- Cordeau, J.-F., Gendreau, M., & Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks: An International Journal*, 30, 105–119.
- Cordeau, J.-F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, 52, 928–936.
- Crama, Y., Kolen, A. W., & Pesch, E. (2005). Local search in combinatorial optimization. *Artificial Neural Networks: An Introduction to ANN Theory and Practice*, (pp. 157–174).

- Cvijović, D., & Klinowski, J. (2002). Taboo search: An approach to the multiple-minima problem for continuous functions. In *Handbook of global optimization* (pp. 387–406). Springer.
- Czech, Z. J., & Czarnas, P. (2002). Parallel simulated annealing for the vehicle routing problem with time windows. In *Proceedings 10th Euromicro workshop on parallel, distributed and network-based processing* (pp. 376–383). IEEE.
- Darwin, C., & Mayr, E. (1859). On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. John Murray, London. *On the Origin of Species by Means of Natural Selection*, (pp. 204–8).
- Das, I., & Dennis, J. E. (1998). Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM journal on optimization*, 8, 631–657.
- Davis, L. (1985). Applying adaptive algorithms to epistatic domains. In *IJCAI* (pp. 162–164). volume 85.
- De Backer, B., Furnon, V., Shaw, P., Kilby, P., & Prosser, P. (2000). Solving vehicle routing problems using constraint programming and metaheuristics. *Journal of Heuristics*, 6, 501–523.
- De Jong, K. A., & Spears, W. M. (1990). An analysis of the interacting roles of population size and crossover in genetic algorithms. In *International Conference on Parallel Problem Solving from Nature* (pp. 38–47). Springer.
- Deb, K., & Jain, H. (2012). Handling many-objective problems using an improved nsga-ii procedure. In *2012 IEEE Congress on Evolutionary Computation* (pp. 1–8). IEEE.
- Deb, K., & Jain, H. (2013). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. *IEEE transactions on evolutionary computation*, 18, 577–601.
- Deb, K., Mohan, M., & Mishra, S. (2005). Evaluating the ϵ -domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evolutionary computation*, 13, 501–525.
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6, 182–197.
- Deep, K., & Mebrahtu, H. (2011). Combined mutation operators of genetic algorithm for the travelling salesman problem. *International Journal of Combinatorial Optimization Problems and Informatics*, 2, 1–23.
- Deep, K., & Thakur, M. (2007). A new mutation operator for real coded genetic algorithms. *Applied mathematics and Computation*, 193, 211–230.
- Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1, 3–18.

- Di Pierro, F., Khu, S.-T., & Savic, D. A. (2007). An investigation on preference order ranking scheme for multiobjective evolutionary optimization. *IEEE transactions on evolutionary computation*, *11*, 17–45.
- Díaz-Ramírez, J., & De la Peña, M. G. B. (2017). Effects of ambulance dispatching and re-location decisions on ems quality. In *Proceedings of the International Conference on Industrial Engineering and Operations Management* (pp. 557–563). IEOM Society.
- Ding, Q., Hu, X., Sun, L., & Wang, Y. (2012). An improved ant colony optimization and its application to vehicle routing problem with time windows. *Neurocomputing*, *98*, 101–107.
- Ding, R., Dong, H., He, J., & Li, T. (2019). A novel two-archive strategy for evolutionary many-objective optimization algorithm based on reference points. *Applied Soft Computing*, *78*, 447–464.
- Diosan, L. (2005). A multi-objective evolutionary approach to the portfolio optimization problem. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)* (pp. 183–187). IEEE volume 2.
- Dong, H., Sun, J., Sun, X., & Ding, R. (2020). A many-objective feature selection for multi-label classification. *Knowledge-Based Systems*, *208*, 106456.
- Dong, N., & Dai, C. (2019). An improvement decomposition-based multi-objective evolutionary algorithm using multi-search strategy. *Knowledge-Based Systems*, *163*, 572–580.
- Dong, W., Zhou, K., Zhang, G., & Chao, H.-C. (2018). Modified discrete glowworm swarm optimization algorithm based on time window division for multi-objective vrptw. *Journal of Internet Technology*, *19*, 001–013.
- Drechsler, N., Drechsler, R., & Becker, B. (2001). Multi-objective optimisation based on relation favour. In *International conference on evolutionary multi-criterion optimization* (pp. 154–166). Springer.
- Eckhardt, R., Ulam, S., & Von Neumann, J. (1987). the monte carlo method. *Los Alamos Science*, *15*, 131.
- Elarbi, M., Bechikh, S., & Said, L. B. (2018). On the importance of isolated infeasible solutions in the many-objective constrained nsga-iii. *Knowledge-Based Systems*, (p. 104335).
- Ellabib, I., Basir, O. A., & Calamai, P. (2002). An experimental study of a simple ant colony system for the vehicle routing problem with time windows. In *International Workshop on Ant Algorithms* (pp. 53–64). Springer.
- Erdoğan, S., & Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation research part E: logistics and transportation review*, *48*, 100–114.
- Escobar, J. W., Linfati, R., Toth, P., & Baldoquin, M. G. (2014). A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem. *Journal of heuristics*, *20*, 483–509.
- Falkenauer, E. (1999). The worth of the uniform [uniform crossover]. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)* (pp. 776–782). IEEE volume 1.

- Farina, M., & Amato, P. (2004). A fuzzy definition of "optimality" for many-criteria optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 34, 315–326.
- Forrest, S. (1996). Genetic algorithms. *ACM computing surveys (CSUR)*, 28, 77–80.
- Galindres-Guancha, L., Toro-Ocampo, E., & Rendón, R. (2018). Multi-objective mdvrp solution considering route balance and cost using the ils metaheuristic. *International Journal of Industrial Engineering Computations*, 9, 33–46.
- Gambardella, L. M., Taillard, É., & Agazzi, G. (1999). Macs-vrptw: A multiple colony system for vehicle routing problems with time windows. In *New ideas in optimization*. Citeseer.
- Garcia-Najera, A., & Bullinaria, J. A. (2009). Bi-objective optimization for the vehicle routing problem with time windows: Using route similarity to enhance performance. In *International Conference on Evolutionary Multi-Criterion Optimization* (pp. 275–289). Springer.
- Garcia-Najera, A., & Bullinaria, J. A. (2011). An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, 38, 287–300.
- Ghoseiri, K., & Ghannadpour, S. F. (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing*, 10, 1096–1107.
- Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, 13, 533–549.
- Gong, Y.-J., Zhang, J., Liu, O., Huang, R.-Z., Chung, H. S.-H., & Shi, Y.-H. (2011). Optimizing the vehicle routing problem with time windows: a discrete particle swarm optimization approach. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42, 254–267.
- Got, A., Moussaoui, A., & Zouache, D. (2020). A guided population archive whale optimization algorithm for solving multiobjective optimization problems. *Expert Systems with Applications*, 141, 112972.
- Grosso, R., Muñozuri, J., Escudero-Santana, A., & Barbadilla-Martín, E. (2018). Mathematical formulation and comparison of solution approaches for the vehicle routing problem with access time windows. *Complexity*, 2018.
- Grotschel, M., & Lovász, L. (1995). Combinatorial optimization. *Handbook of combinatorics*, 2, 4.
- Hansen, M. P., & Jaszkiwicz, A. (1994). *Evaluating the quality of approximations to the non-dominated set*. Citeseer.
- Hasle, G., Lie, K.-A., & Quak, E. (2007). *Geometric modelling, numerical simulation, and optimization*. Springer.
- He, Y., & Aranha, C. (2020). Solving portfolio optimization problems using moea/d and levy flight. *Advances in Data Science and Adaptive Analysis*, 12, 2050005.

- Hemici, M., Zouache, D., Boualem, B., & Hemici, K. (2021). An external archive guided nsga-ii algorithm for multi-depot green vehicle routing problem. In *International Conference on Artificial Intelligence and its Applications* (pp. 504–513). Springer.
- Hemici, M., Zouache, D., Brahmi, B., Got, A., & Drias, H. (2023). A decomposition-based multiobjective evolutionary algorithm using simulated annealing for the ambulance dispatching and relocation problem during covid-19. *Applied Soft Computing*, (p. 110282).
- Homberger, J. (2000). Verteilt-parallele metaheuristiken zur tourenplanung: Lösungsverfahren für das standardproblem mit zeitenfensterrestriktionen: Fernuniv. hagen, diss., 2000 (1. aufl.). gabler edition wissenschaft. wiesbaden: Dt. univ. Verl.[ua], .
- Homberger, J., & Gehring, H. (1999). Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR: Information Systems and Operational Research*, 37, 297–318.
- Homberger, J., & Gehring, H. (2005). A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European journal of operational research*, 162, 220–238.
- Hsu, W.-H., & Chiang, T.-C. (2012). A multiobjective evolutionary algorithm with enhanced reproduction operators for the vehicle routing problem with time windows. In *2012 IEEE Congress on Evolutionary Computation* (pp. 1–8). IEEE.
- Hu, W., Liang, H., Peng, C., Du, B., & Hu, Q. (2013). A hybrid chaos-particle swarm optimization algorithm for the vehicle routing problem with time window. *Entropy*, 15, 1247–1270.
- Hu, X.-B., & Di Paolo, E. (2009). An efficient genetic algorithm with uniform crossover for air traffic control. *Computers & Operations Research*, 36, 245–259.
- Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T., & Yagiura, M. (2005). Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation science*, 39, 206–232.
- Ibri, S., Nourelfath, M., & Drias, H. (2012). A multi-agent approach for integrated emergency vehicle dispatching and covering problem. *Engineering Applications of Artificial Intelligence*, 25, 554–565.
- Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. In *International Conference on Evolutionary Multi-Criterion Optimization* (pp. 359–372). Springer.
- Ishibuchi, H., & Murata, T. (1998). A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, 28, 392–403.
- Jabir, E., Panicker, V. V., & Sridharan, R. (2017). Design and development of a hybrid ant colony-variable neighbourhood search algorithm for a multi-depot green vehicle routing problem. *Transportation Research Part D: Transport and Environment*, 57, 422–457.
- Jalilian, J., Ehtesham Rasi, R., & Fallah Shams, M. (2021). Multi objective portfolio optimization for a private equity investment company under data insufficiency condition. *International Journal of Finance & Managerial Accounting*, 6, 23–37.

- Jia-li, L., & Zu-jun, M. (2013). Multi-depot open vehicle routing problem with time windows based on vehicle leasing and sharing [j]. *Systems Engineering-Theory & Practice*, 3.
- Jung, S., & Moon, B. R. (2002). A hybrid genetic algorithm for the vehicle routing problem with time windows. In *GECCO* (pp. 1309–1316).
- Jungnickel, D., & Jungnickel, D. (1999). The greedy algorithm. *Graphs, Networks and Algorithms*, (pp. 129–153).
- Jungnickel, D., & Jungnickel, D. (2013). The greedy algorithm. *Graphs, Networks and Algorithms*, (pp. 135–161).
- Kaabachi, I., Jriji, D., & Krichen, S. (2017). An improved ant colony optimization for green multi-depot vehicle routing problem with time windows. In *2017 18th IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD)* (pp. 339–344). IEEE.
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (pp. 1942–1948). IEEE volume 4.
- Khoo, T. S., & Mohammad, B. B. (2021). The parallelization of a two-phase distributed hybrid ruin-and-recreate genetic algorithm for solving multi-objective vehicle routing problem with time windows. *Expert Systems with Applications*, 168, 114408.
- Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, 220, 671–680.
- Köppen, M., Vicente-Garcia, R., & Nickolay, B. (2005). Fuzzy-pareto-dominance and its application in evolutionary multi-objective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization* (pp. 399–412). Springer.
- Kramer, O., & Kramer, O. (2013). K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, (pp. 13–23).
- Kremmel, T., Kubalik, J., & Biffel, S. (2011). Software project portfolio optimization with advanced multiobjective evolutionary algorithms. *Applied Soft Computing*, 11, 1416–1426.
- Kwon, Y.-J., Choi, Y.-J., & Lee, D.-H. (2013). Heterogeneous fixed fleet vehicle routing considering carbon emission. *Transportation Research Part D: Transport and Environment*, 23, 81–89.
- Labadi, N., Prins, C., & Reghioui, M. (2008). A memetic algorithm for the vehicle routing problem with time windows. *RAIRO-Operations research*, 42, 415–431.
- Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary computation*, 10, 263–282.
- Lawler, E. L., & Wood, D. E. (1966). Branch-and-bound methods: A survey. *Operations research*, 14, 699–719.
- Le Bouthillier, A., & Crainic, T. G. (2005). A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. *Computers & Operations Research*, 32, 1685–1708.

- Li, H., Deng, J., Zhang, Q., & Sun, J. (2019). Adaptive epsilon dominance in decomposition-based multiobjective evolutionary algorithm. *Swarm and Evolutionary Computation*, 45, 52–67.
- Li, H., & Lim, A. (2003). Local search with annealing-like restarts to solve the vrptw. *European journal of operational research*, 150, 115–127.
- Li, J., Wang, R., Li, T., Lu, Z., & Pardalos, P. M. (2018). Benefit analysis of shared depot resources for multi-depot vehicle routing problem with fuel consumption. *Transportation Research Part D: Transport and Environment*, 59, 417–432.
- Li, K., Deb, K., Zhang, Q., & Kwong, S. (2014a). Efficient non-domination level update approach for steady-state evolutionary multiobjective optimization. *Department of Electrical and Computer Engineering, Michigan State University, East Lansing, USA, Tech. Rep. COIN Report, 2014014*.
- Li, K., Deb, K., Zhang, Q., & Kwong, S. (2014b). An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Transactions on Evolutionary Computation*, 19, 694–716.
- Liagkouras, K., & Metaxiotis, K. (2014). A new probe guided mutation operator and its application for solving the cardinality constrained portfolio optimization problem. *Expert Systems with Applications*, 41, 6274–6290.
- Liagkouras, K., & Metaxiotis, K. (2018). A new efficiently encoded multiobjective algorithm for the solution of the cardinality constrained portfolio optimization problem. *Annals of Operations Research*, 267, 281–319.
- Liagkouras, K., Metaxiotis, K. et al. (2013). The constrained mean-semivariance portfolio optimization problem with the support of a novel multiobjective evolutionary algorithm. *Journal of Software Engineering and Applications*, 6, 22–29.
- Liu, C., & Du, Y. (2019). A membrane algorithm based on chemical reaction optimization for many-objective optimization problems. *Knowledge-Based Systems*, 165, 306–320.
- Liu, C., Kou, G., Zhou, X., Peng, Y., Sheng, H., & Alsaadi, F. E. (2020a). Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach. *Knowledge-Based Systems*, 188, 104813.
- Liu, C.-A., Lei, Q., & Jia, H. (2022). Maximum entropy bi-objective model and its evolutionary algorithm for portfolio optimization. *Asia-Pacific Journal of Operational Research*, (p. 2250014).
- Liu, R., Liu, J., Zhou, R., Lian, C., & Bian, R. (2020b). A region division based decomposition approach for evolutionary many-objective optimization. *Knowledge-Based Systems*, 194, 105518.
- Mahmudy, W. F. (2014). Improved simulated annealing for optimization of vehicle routing problem with time windows (vrptw). *Jurnal Ilmiah Cursor*, 7.
- Majzoubi, F., Bai, L., & Heragu, S. S. (2012). An optimization approach for dispatching and relocating ems vehicles. *IIE Transactions on Healthcare Systems Engineering*, 2, 211–223.

- Mamanis, G. (2021). A comparative study on multi-objective evolutionary algorithms for tri-objective mean-risk-cardinality portfolio optimization problems. In *Computational Management* (pp. 277–303). Springer.
- Marinakis, Y., Marinaki, M., & Migdalas, A. (2019). A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows. *Information Sciences*, *481*, 311–329.
- Menchaca-Mendez, A., & Coello, C. A. C. (2015). Gde-moea: a new moea based on the generational distance indicator and ϵ -dominance. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (pp. 947–955). IEEE.
- Mester, D. (2002). An evolutionary strategies algorithm for large scale vehicle routing problem with capacitate and time windows restrictions. In *proceedings of the conference on mathematical and population genetics, University of Haifa, Israel*.
- Miettinen, K. (2012). *Nonlinear multiobjective optimization* volume 12. Springer Science & Business Media.
- Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, *69*, 46–61.
- Mishra, S. K., Ganapati, P., Meher, S., & Majhi, R. (2002). A fast multiobjective evolutionary algorithm for finding wellspread pareto-optimal solutions. In *In KanGAL Report No. 2003002, Indian Institute Of Technology Kanpur*. Citeseer.
- Mishra, S. K., Panda, G., Majhi, B., & Majhi, R. (2012). Improved portfolio optimization combining multiobjective evolutionary computing algorithm and prediction strategy. In *World Congress on Engineering*. volume 1.
- Mitchell, J. E. (2002). Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, *1*, 65–77.
- Mkaouer, W., Kessentini, M., Shaout, A., Koligheu, P., Bechikh, S., Deb, K., & Ouni, A. (2015). Many-objective software remodularization using nsga-iii. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *24*, 1–45.
- Molina, J. C., Salmeron, J. L., & Eguia, I. (2020). An acs-based memetic algorithm for the heterogeneous vehicle routing problem with time windows. *Expert Systems with Applications*, *157*, 113379.
- Montoya-Torres, J. R., Franco, J. L., Isaza, S. N., Jiménez, H. F., & Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, *79*, 115–129.
- Moradi, B. (2020). The new optimization algorithm for the vehicle routing problem with time windows using multi-objective discrete learnable evolution model. *Soft Computing*, *24*, 6741–6769.
- Nasrollahzadeh, A. A., Khademi, A., & Mayorga, M. E. (2018). Real-time ambulance dispatching and relocation. *Manufacturing & Service Operations Management*, *20*, 467–480.

- Niu, Y., Yang, Z., Chen, P., & Xiao, J. (2018). Optimizing the green open vehicle routing problem with time windows by minimizing comprehensive routing cost. *Journal of cleaner production*, 171, 962–971.
- de Oliveira, H. C. B., & Vasconcelos, G. C. (2010). A hybrid search method for the vehicle routing problem with time windows. *Annals of Operations Research*, 180, 125–144.
- Ombuki, B., Ross, B. J., & Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24, 17–30.
- Pechina, A., Usanov, D., van de Ven, P., & van der Mei, R. (2019). Real-time dispatching and relocation of emergency service engineers. *arXiv preprint arXiv:1910.01427*, .
- Peterson, L. E. (2009). K-nearest neighbor. *Scholarpedia*, 4, 1883.
- Pirlot, M. (1996). General local search methods. *European journal of operational research*, 92, 493–511.
- Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers & operations research*, 34, 2403–2435.
- Poli, R., & Langdon, W. B. (1997). A new schema theorem for genetic programming with one-point crossover and point mutation. *Cognitive Science Research Papers-University of Birmingham CSRP*, .
- Poli, R., & Langdon, W. B. (1998). *Genetic programming with one-point crossover*. Springer.
- Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & operations research*, 31, 1985–2002.
- Qi, Y., Hou, Z., Li, H., Huang, J., & Li, X. (2015). A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows. *Computers & Operations Research*, 62, 61–77.
- Repoussis, P. P., Tarantilis, C. D., & Ioannou, G. (2007). The open vehicle routing problem with time windows. *Journal of the Operational Research Society*, 58, 355–367.
- Repoussis, P. P., Tarantilis, C. D., & Ioannou, G. (2009). Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE Transactions on Evolutionary Computation*, 13, 624–647.
- Rochat, Y., & Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1, 147–167.
- Rousseau, L.-M., Gendreau, M., & Pesant, G. (2002). Using constraint-based operators to solve the vehicle routing problem with time windows. *Journal of heuristics*, 8, 43–58.
- Schaffer, J. D. (1985). *Some experiments in machine learning using vector evaluated genetic algorithms*. Technical Report Vanderbilt Univ., Nashville, TN (USA).
- Schott, J. R. (1995). *Fault tolerant design using single and multicriteria genetic algorithm optimization*. Ph.D. thesis Massachusetts Institute of Technology.

- Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159, 139–171.
- Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, 46.
- Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming* (pp. 417–431). Springer.
- Shen, L., Tao, F., & Wang, S. (2018). Multi-depot open vehicle routing problem with time windows based on carbon trading. *International journal of environmental research and public health*, 15, 2025.
- Skolpadungket, P., Dahal, K., & Harnpornchai, N. (2007). Portfolio optimization using multi-objective genetic algorithms. In *2007 IEEE Congress on Evolutionary Computation* (pp. 516–523). IEEE.
- Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35, 254–265.
- Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2, 221–248.
- Streichert, F., & Tanaka-Yamawaki, M. (2006). The effect of local search on the constrained portfolio selection problem. In *2006 IEEE International Conference on Evolutionary Computation* (pp. 2368–2374). IEEE.
- Suganya, N., & Vijayalakshmi Pai, G. (2010). Pareto-archived evolutionary wavelet network for financial constrained portfolio optimization. *Intelligent Systems in Accounting, Finance & Management*, 17, 59–90.
- Sun, Y., Yen, G. G., & Yi, Z. (2018). Igd indicator-based evolutionary algorithm for many-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, 23, 173–187.
- Taillard, É., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31, 170–186.
- Tan, K. C. (2001). A framework of supply chain management literature. *European Journal of Purchasing & Supply Management*, 7, 39–48.
- Tan, K. C., Chew, Y. H., & Lee, L. (2006). A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Computational Optimization and Applications*, 34, 115.
- Tan, K. C., Lee, L. H., Zhu, Q., & Ou, K. (2001). Heuristic methods for vehicle routing problem with time windows. *Artificial intelligence in Engineering*, 15, 281–295.

- Tarantilis, C., & Kiranoudis, C. (2002). Distribution of fresh meat. *Journal of Food Engineering*, 51, 85–91.
- Thangiah, S. R., Osman, I. H., & Sun, T. (1994). Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows. *Computer Science Department, Slippery Rock University, Technical Report SRU CpSc-TR-94-27*, 69.
- Tian, Y., Zhang, X., Cheng, R., & Jin, Y. (2016). A multi-objective evolutionary algorithm based on an enhanced inverted generational distance metric. In *2016 IEEE congress on evolutionary computation (CEC)* (pp. 5222–5229). IEEE.
- Ting, C.-K., Su, C.-H., & Lee, C.-N. (2010). Multi-parent extension of partially mapped crossover for combinatorial optimization problems. *Expert systems with applications*, 37, 1879–1886.
- Toffolo, A., & Benini, E. (2003). Genetic diversity as an objective in multi-objective evolutionary algorithms. *Evolutionary computation*, 11, 151–167.
- Umbarkar, A. J., & Sheth, P. D. (2015). Crossover operators in genetic algorithms: a review. *ICTACT journal on soft computing*, 6.
- Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60, 611–624.
- Wang, F., Zhang, H., Li, Y., Zhao, Y., & Rao, Q. (2018a). External archive matching strategy for moea/d. *Soft Computing*, 22, 7833–7846.
- Wang, J., Weng, T., & Zhang, Q. (2018b). A two-stage multiobjective evolutionary algorithm for multiobjective multidepot vehicle routing problem with time windows. *IEEE transactions on cybernetics*, 49, 2467–2478.
- Wang, S., Tao, F., & Shi, Y. (2018c). Optimization of location–routing problem for cold chain logistics considering carbon footprint. *International journal of environmental research and public health*, 15, 86.
- Wang, Y., Assogba, K., Fan, J., Xu, M., Liu, Y., & Wang, H. (2019). Multi-depot green vehicle routing problem with shared transportation resource: Integration of time-dependent speed and piecewise penalty cost. *Journal of Cleaner Production*, 232, 12–29.
- Wang, Y., Sun, Y., Guan, X., Fan, J., Xu, M., & Wang, H. (2021). Two-echelon multi-period location routing problem with shared transportation resource. *Knowledge-Based Systems*, (p. 107168).
- Wang, Z., Zhang, Q., Zhou, A., Gong, M., & Jiao, L. (2015). Adaptive replacement strategies for moea/d. *IEEE transactions on cybernetics*, 46, 474–486.
- Wei, C., Qiu, C., Xin, Q., & Fan, Z. (2018). An improved genetic algorithm for vehicle routing problem with time windows. *Entropy*, 17, 2.
- Wei, Z., Yang, J., Hu, Z., & Sun, H. (2021). An adaptive decomposition evolutionary algorithm based on environmental information for many-objective optimization. *ISA transactions*, 111, 108–120.

- Williams, E. A., & Crossley, W. A. (1998). Empirically-derived population size and mutation rate guidelines for a genetic algorithm with uniform crossover. In *Soft computing in engineering design and manufacturing* (pp. 163–172). Springer.
- Xiong, Z., Yang, J., Hu, Z., Zhao, Z., & Wang, X. (2021). Evolutionary many-objective optimization algorithm based on angle and clustering. *Applied Intelligence*, *51*, 2045–2062.
- Yalcinoz, T., Altun, H., & Uzam, M. (2001). Economic dispatch solution using a genetic algorithm based on arithmetic crossover. In *2001 IEEE Porto Power Tech Proceedings (Cat. No. 01EX502)* (pp. 4–pp). IEEE volume 2.
- Yang, F., Xu, L., Chu, X., & Wang, S. (2021). A new dominance relation based on convergence indicators and niching for many-objective optimization. *Applied Intelligence*, *51*, 5525–5542.
- Yang, S., Li, M., Liu, X., & Zheng, J. (2013). A grid-based evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, *17*, 721–736.
- Yu, B., Yang, Z., & Xie, J. (2011). A parallel improved ant colony optimization for multi-depot vehicle routing problem. *Journal of the Operational Research Society*, *62*, 183–188.
- Yuan, J., Liu, H.-L., Gu, F., Zhang, Q., & He, Z. (2020). Investigating the properties of indicators and an evolutionary many-objective algorithm using promising regions. *IEEE Transactions on Evolutionary Computation*, *25*, 75–86.
- Yuan, Y., Xu, H., Wang, B., & Yao, X. (2015). A new dominance relation-based evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, *20*, 16–37.
- Zhang, J., Yang, F., & Weng, X. (2018). An evolutionary scatter search particle swarm optimization algorithm for the vehicle routing problem with time windows. *IEEE Access*, *6*, 63468–63485.
- Zhang, Q., & Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, *11*, 712–731.
- Zhang, W., Gajpal, Y., Appadoo, S. S., & Wei, Q. (2020). Multi-depot green vehicle routing problem to minimize carbon emissions. *Sustainability*, *12*, 3500.
- Zhao, P., Gao, S., & Yang, N. (2020a). Solving multi-objective portfolio optimization problem based on moea/d. In *2020 12th International Conference on Advanced Computational Intelligence (ICACI)* (pp. 30–37). IEEE.
- Zhao, W., Zhang, Z., & Wang, L. (2020b). Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. *Engineering Applications of Artificial Intelligence*, *87*, 103300.
- Zhao, Y. W., Wu, B., Wang, W., Ma, Y. L., Wang, W., & Sun, H. (2004). Particle swarm optimization for vehicle routing problem with time windows. In *Materials Science Forum* (pp. 801–805). Trans Tech Publ volume 471.
- Zhou, Z., Liu, X., Xiao, H., Wu, S., & Liu, Y. (2019). A dea-based moea/d algorithm for portfolio optimization. *Cluster Computing*, *22*, 14477–14486.

- Zhu, Q., Qian, L., Li, Y., & Zhu, S. (2006). An improved particle swarm optimization algorithm for vehicle routing problem with time windows. In *2006 IEEE International Conference on Evolutionary Computation* (pp. 1386–1390). IEEE.
- Zitzler, E., & Künzli, S. (2004). Indicator-based selection in multiobjective search. In *International conference on parallel problem solving from nature* (pp. 832–842). Springer.
- Zitzler, E., Laumanns, M., & Bleuler, S. (2004). A tutorial on evolutionary multiobjective optimization. In *Metaheuristics for multiobjective optimisation* (pp. 3–37). Springer.
- Zitzler, E., Laumanns, M., & Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report, 103*.
- Zitzler, E., & Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms—a comparative case study. In *International conference on parallel problem solving from nature* (pp. 292–301). Springer.
- Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation, 3*, 257–271.
- Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation, 7*, 117–132.
- Zitzler, E., Laumanns, M., & Thiele, L. (2002). Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. *Evolutionary Methods for Design, Optimization, and Control*, (pp. 95–100).

ملخص

يندرج النشاط البحثي لهذه الأطروحة تحت موضوعين رئيسيين هما التحسين متعدد الأهداف وإدارة سلسلة التوريد، حيث يعد التحسين متعدد الأهداف أحد فروع التحسين التجميعي الذي تتمثل خصوصيته في السعي إلى تحسين العديد من الأهداف في وقت واحد لنفس المشكلة. حيث يتيح استخدام أساليب التحسين متعددة الأهداف للشركات استخدام الحلول الإستراتيجية المثلى لتحسين جودة القرارات لإدارة سلسلة التوريد. وخلال دراستنا اقترحنا ثلاثة خوارزميات جديدة متعددة الأغراض لحل المشكلات المتعلقة بإدارة سلسلة التوريد (SCM) حيث تركز إهتمام على الجانب المتعلق بمشاكل النقل و التوزيع واهمها مشكلة توجيه السيارة مع نوافذ الوقت، مشكلة توجيه المركبات الخضراء متعددة المستودعات، و مشكلة إرسال ونقل سيارات الإسعاف، وتستند الخوارزمية الأولى المقترحة الى دمج هيمنة إبسيلون الخوارزمية التطورية متعددة الأغراض (ϵ -MOEA) مع البحث المحلي وعوامل التقاطع متعددة الأنواع لحل مشكلة توجيه السيارة مع نوافذ الوقت. أما في المقترح الثاني فقد تم تحسين خوارزمية NSGA-II عن طريق دمج طريقة البحث المحلي و مجموعة خارجية تسمى أرشيف" لتخزين أفضل الحلول "المنخبة" لحل مشكلة توجيه المركبات الخضراء متعددة المستودعات. وتم اقتراح خوارزمية MOEA/D المحسنة في المقترح الثالث باستخدام التلدين المحاكي (SA) لحل مشكلة إرسال ونقل سيارات الإسعاف. و بعد تجربها، اعطت الخوارزميات المقترحة نتائج مشجعة سواء بالنسبة لحل مشكلة توجيه السيارة مع نوافذ الوقت، مشكلة توجيه المركبات الخضراء متعددة المستودعات أو مشكلة إرسال ونقل سيارات الإسعاف، وأثبتت أنها أكثر كفاءة بالمقارنة مع العديد من خوارزميات التطورية متعددة الأغراض.

الكلمات المفتاحية: إدارة سلسلة التوريد، التحسين متعدد الأهداف، خوارزميات التطورية متعددة الأغراض، مشكلة توجيه السيارات، مشكلة إرسال ونقل سيارات الإسعاف.

Résumé

L'activité s'inscrit dans deux grands thèmes : la recherche sur l'optimisation multi-objectifs et la gestion de la chaîne logistique (GCL). L'optimisation multi-objectifs est une branche combinatoire dont la spécificité est de chercher à optimiser plusieurs objectifs simultanément pour un même problème. Par conséquent, l'utilisation de méthodes d'optimisation multi-objectifs permet aux entreprises d'utiliser des solutions stratégiques efficace afin d'optimiser et d'améliorer la qualité des décisions pour la GCL. Dans cette thèse, nous proposons trois nouveaux algorithmes évolutionnaires multi-objectifs pour résoudre trois difficultés importantes dans GCL où notre attention s'est portée sur la partie liée aux soucis de transport, de distribution, à savoir le problème des tournées de véhicules avec fenêtres, véhicules multi-dépôts, respect de l'environnement, problème de relocalisation ainsi que la répartition des ambulances. Le premier algorithme appelé External Archive Guided Nondominated Sorting Genetic Algorithm II (EAG-NSGA-II) est basé sur l'amélioration de l'algorithme NSGA-II en intégrant la méthode de recherche locale et une population externe appelée "archive" pour stocker les solutions non dominées afin de résoudre le problème de tournées de véhicules multi-dépôts respectueuses de l'environnement. Dans le deuxième algorithme, un algorithme MOEA/D amélioré en utilisant le recuit simulé (RS) qui a été proposé pour résoudre le problème de relocalisation et la répartition des ambulances. Dans le dernier algorithme, un nouvel algorithme évolutionnaire a été développé en combinant l'algorithme ϵ -MOEA, la méthode de recherche locale et les opérateurs de croisement multi-types pour résoudre le problème de tournées de véhicules avec fenêtres de temps.

Les algorithmes que nous avons proposés ont été soumis à des tests et ont donné des résultats encourageants pour la résolution des problèmes de tournées de véhicules avec fenêtres de temps, de tournées de véhicules multi-dépôts respectueuses de l'environnement et de la relocalisation et de la répartition des ambulances. Ils se sont révélés plus efficaces que de nombreux autres algorithmes évolutionnaires multi-objectif.

Mots clés : Gestion de la chaîne logistique ; optimisation multi-objectifs ; algorithmes évolutionnaires multi-objectifs ; problème de tournées des véhicules ; problème de relocalisation et de la répartition des ambulances.

Abstract

The research activity of this research falls under two major themes: multi-objective optimization and supply chain management (SCM). Multiobjective optimization is a branch of combinatorial optimization whose specificity is to seek to optimize several objectives simultaneously for the same problem. As a result, the use of multi-objective optimization methods allows companies to use optimal strategic solutions to optimize and improve the quality of decisions for SCM.

In this thesis, we proposed three new multi-objective evolutionary algorithms to solve three important problems in SCM where our attention was focused on the part related to transportation and distribution problems, namely the vehicle routing problem with time windows, the multi-depot green vehicle routing problem, and the ambulance relocation and dispatching problem.

The first algorithm called External Archive Guided Nondominated Sorting Genetic Algorithm II (EAG-NSGA-II) is based on improving the NSGA-II algorithm by integrating the local search method and an external population called “archive” to store the non-dominated solutions to solve the multi-depot green vehicle routing problem. While in the second algorithm, an improved MOEA/D algorithm using simulated annealing (SA) was proposed to solve the ambulance relocation and dispatching problem. In the last algorithm, a new evolutionary algorithm was developed by combining the ϵ -MOEA algorithm, local search method, and multi-type crossover operators to solve the vehicle routing problem with time windows.

After being tested, the proposed algorithms have shown encouraging results in both solving the vehicle routing problem with time windows, the multi-depot green vehicle routing problem, and the ambulance relocation and dispatching problem, and have been found to be more efficient compared to many multi-objective evolutionary algorithms.

Keywords : Supply chain management; Multi-objective optimization; Multi-objective evolutionary algorithms; Vehicle routing problem; Ambulance relocation and dispatching problem.