Rebiha Kemcha

A Multi-Criteria Decision Support System for Multi-Objective Optimization Algorithm Selection

Final Master's Project

Directed by Dr. Aïda Valls Mateu

Master's Degree in Computer Security Engineering and Artificial Intelligence

UNIVERSITAT ROVIRA i VIRGILI

Tarragona
2024

Abstract

Multi-Objective Optimization (MOO) addresses the challenge of optimizing multiple conflicting objectives concurrently, typically leading to the identification of compromise or Pareto-optimal solutions. Selecting an appropriate optimization algorithm is crucial, as algorithmic performance can vary significantly depending on the problem's characteristics, data representation, and parameter settings. Identifying the most effective algorithm is a resource-intensive process and can lead to inefficiencies, particularly when an unsuitable algorithm is chosen, potentially resulting in suboptimal solutions.

Multi-Criteria Decision Making (MCDM) offers a systematic approach to addressing this challenge. MCDM provides methodologies for evaluating and selecting the best alternative from a set of options based on multiple, often conflicting criteria. Within MCDM, various methods exist, each offering distinct strategies to aid decision-making. One such method, the Logic Scoring of Preferences (LSP), is particularly effective in scenarios requiring a detailed evaluation of alternatives. LSP aggregates multiple criteria into a coherent evaluation model using logic operations, allowing decision-makers to systematically consider diverse aspects of each option.

In this Master's thesis, we propose the development of a decision support system that integrates MCDM principles to facilitate the selection of the most appropriate Multi-Objective Evolutionary Algorithm (MOEA), a class of MOO algorithms, for a given optimization problem. The proposed system employs the LSP method to systematically evaluate and rank algorithms based on a comprehensive set of criteria, including algorithmic performance and problem-specific features. This approach aims to enhance the efficiency of algorithm selection, ensuring alignment with the specific requirements of the problem at hand and the preferences of the decision-maker. The system is tested on established benchmarks and a case study in Evolutionary Circuit Design.

The findings demonstrate that the system consistently identifies algorithms that outperform others in both computational efficiency and solution quality. The consideration of landscape features empowers the system to identify algorithms that are most highly ranked in resolving complex problem instances at lower evaluation budget.

Resum

L'Optimització Multiobjectiu (MOO) aborda el repte d'optimitzar múltiples objectius conflictius de manera simultània, la qual cosa generalment porta a la identificació de solucions de compromís o Pareto-òptimes. La selecció d'un algoritme d'optimització adequat és crucial, ja que el rendiment algorísmic pot variar significativament en funció de les característiques del problema, la representació de les dades i la configuració dels paràmetres. Identificar l'algoritme més efectiu és un procés que consumeix molts recursos i pot conduir a ineficiències, especialment quan es tria un algoritme inadequat, cosa que pot resultar en solucions subòptimes.

La Presa de Decisions Multicriteri (MCDM) ofereix un enfocament sistemàtic per abordar aquest repte. MCDM proporciona metodologies per avaluar i seleccionar la millor alternativa d'un conjunt d'opcions basant-se en múltiples criteris, sovint conflictius. Dins de MCDM, existeixen diversos mètodes, cadascun dels quals ofereix estratègies distintes per ajudar en la presa de decisions. Un d'aquests mètodes, la Logic Scoring of Preferences (LSP), és particularment efectiu en escenaris que requereixen una avaluació detallada d'alternatives. LSP agrega múltiples criteris en un model d'avaluació coherent utilitzant operacions lògiques, permetent als responsables de la presa de decisions considerar sistemàticament diversos aspectes de cada opció.

En aquesta tesi de màster, proposem el desenvolupament d'un sistema de suport a la decisió que integra els principis de MCDM per facilitar la selecció de l'Algoritme Evolutiu Multiobjectiu (MOEA) més adequat, una classe d'algoritmes de MOO, per a un problema d'optimització determinat. El sistema proposat empra el mètode LSP per avaluar i classificar sistemàticament els algoritmes basant-se en un conjunt integral de criteris, incloent-hi el rendiment algorísmic i les característiques específiques del problema. Aquest enfocament té com a objectiu millorar l'eficiència en la selecció d'algoritmes, assegurant l'alineació amb els requisits específics del problema en qüestió i les preferències del decisor. El sistema es posa a prova amb benchmarks establerts i un estudi de cas en Disseny Evolutiu de Circuits.

Els resultats demostren que el sistema identifica consistentment algoritmes que superen altres tant en eficiència computacional com en qualitat de la solució. La consideració de les característiques de l'entorn permet al sistema identificar els algoritmes més ben classificats per resoldre instàncies de problemes complexos amb un pressupost d'avaluació més baix.

Resumen

Optimización Multiobjetivo (MOO) aborda el desafío de optimizar múltiples objetivos conflictivos de manera simultánea, lo que generalmente lleva a la identificación de soluciones de compromiso o Pareto-óptimas. La selección de un algoritmo de optimización adecuado es crucial, ya que el rendimiento algorítmico puede variar significativamente dependiendo de las características del problema, la representación de los datos y la configuración de los parámetros. Identificar el algoritmo más efectivo es un proceso que consume muchos recursos y puede llevar a ineficiencias, especialmente cuando se elige un algoritmo inapropiado, lo que potencialmente resulta en soluciones subóptimas.

La Toma de Decisiones Multicriterio (MCDM) ofrece un enfoque sistemático para abordar este desafío. MCDM proporciona metodologías para evaluar y seleccionar la mejor alternativa de un conjunto de opciones basado en múltiples criterios, a menudo conflictivos. Dentro de MCDM, existen varios métodos, cada uno de los cuales ofrece estrategias distintas para ayudar en la toma de decisiones. Uno de estos métodos, la Puntuación Lógica de Preferencias (LSP), es particularmente efectivo en escenarios que requieren una evaluación detallada de alternativas. LSP agrega múltiples criterios en un modelo de evaluación coherente utilizando operaciones lógicas, lo que permite a los responsables de la toma de decisiones considerar sistemáticamente diversos aspectos de cada opción.

En esta tesis de maestría, proponemos el desarrollo de un sistema de apoyo a la decisión que integra los principios de MCDM para facilitar la selección del Algoritmo Evolutivo Multiobjetivo (MOEA) más adecuado, una clase de algoritmos de MOO, para un problema de optimización dado. El sistema propuesto emplea el método LSP para evaluar y clasificar sistemáticamente los algoritmos basándose en un conjunto integral de criterios, incluyendo el rendimiento algorítmico y las características específicas del problema. Este enfoque tiene como objetivo mejorar la eficiencia en la selección de algoritmos, asegurando la alineación con los requisitos específicos del problema en cuestión y las preferencias del tomador de decisiones. El sistema se prueba en puntos de referencia establecidos y un estudio de caso en Diseño Evolutivo de Circuitos.

Los resultados demuestran que el sistema identifica consistentemente algoritmos que superan a otros tanto en eficiencia computacional como en calidad de la solución. La consideración de las características del entorno permite al sistema identificar los algoritmos mejor clasificados para resolver instancias de problemas complejos con un presupuesto de evaluación más bajo.

Acknowledgments

I am profoundly grateful to my supervisor, Dr. Aïda Valls Mateu, for her invaluable guidance, insightful feedback, and continuous support throughout this research. Her expertise and encouragement were instrumental in shaping this thesis.

I would also like to thank the members of my thesis committee for their acceptance, time and effort to judge my work.

I am deeply indebted to my family for their unwavering support, patience, and understanding during the years of my study.

Contents

List of Acronyms

AHP	Analytic Hierarchy Process
ANP	Analytic Network Process
AS	Algorithm Selection
AS-LLM	Algorithm Selection for Large Language Models
AS-LSP-MOEA	Algorithm Selection using Logic Scoring of Preferences for Multi-Objective Evolutionary Algorithms
ASLib	Algorithm Selection Library
CFPR	Consistent Fuzzy Preference Relation
COPRAS	COmplex PRoportional ASsessment
CSP	Constraint Satisfaction Problem
DBEA	Decomposition-Based Evolutionary Algorithm
DEA	Data Envelopment Analysis
EA	Evolutionary Algorithm
ECD	Evolutionary Circuit Design
ELECTRE	ÉLimination Et Choix Traduisant la REalité
GDE3	Generalized Differential Evolution 3
GDF	Geoffrion-Dyer-Feinberg method
GRA	Grey Relational Analysis
HV	Hypervolume Indicator
IGD	Inverted Generational Distance
ISAC	Instance-Specific Algorithm Configuration

LOPCOW	Logarithmic Percentage Change-driven Objective Weighting
LSMOP	Large-Scale Multi-Objective Problem
MCDA	Multi-Criteria Decision Analysis
MCDM	Multi-Criteria Decision Making
MAXSAT	Maximum Satisfiability
MOA	Multi-Objective optimization Algorithm
MOACO	Multi-Objective Ant Colony Optimization
MOEA	Multi-Objective Evolutionary Algorithm
MOEA/D	Multi-objective Evolutionary Algorithm based on Decomposition
MOIA	Multi-Objective Immune Algorithm
MOO	Multi-Objective Optimization
MOP	Multi-Objective Optimization Problem
MOPSO	Multi-Objective Particle Swarm Optimization
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OAS	Online Algorithm Selection
PF	Pareto Front
PIAS	Per-instance Algorithm Selection
PROMETHEE	Preference Ranking Organization METHod for Enrichment Evaluations
PSAS	Per-set Algorithm Selection
PESA2	Pareto Envelope-based Selection Algorithm 2
RVEA	Reference Vector Guided Evolutionary Algorithm
SAT	Boolean Satisfiability Problem
SNSGA-II	Synchronous Nondominated Sorting Algorithm II
SPEA2	Strength Pareto Evolutionary Algorithm 2
TOPSIS	Technique for Order Preference by Similarity to Ideal Solution
UNSGA-III	Unified Nondominated Sorting Genetic Algorithm III

VIKOR	VIseKriterijumska Optimizacija I Kompromisno Resenje
WASPAS	Weighted Aggregated Sum Product Assessment
WSM	Weighted Sum Model
ZDT	Zitzler-Deb-Thiele test problem family

Chapter 1

Introduction

1.1 Background and Motivation

Multi-objective optimization problems (MOPs) are ubiquitous in real-world applications, spanning diverse fields such as engineering, economics, logistics, and artificial intelligence. Unlike single-objective optimization problems, MOPs involve the simultaneous optimization of two or more, often conflicting, objectives, resulting in a set of trade-off solutions rather than a single optimal solution. This set of trade-off solutions, known as the Pareto front, represents the best possible compromises between objectives, making it essential for decision-makers to choose the most suitable solutions based on their preferences.

Multi-Objective algorithms (MOAs) are designed to tackle the complexities encountered in real-world MOPs. The growing importance of MOAs in addressing these complex problems has led to a proliferation of algorithms, each with unique strengths and weaknesses. While this diversity offers a range of tools, it also presents a significant challenge: systematically evaluating and selecting the most suitable algorithm for a given problem. The performance of an MOA is highly problem-dependent, meaning an algorithm that excels in one scenario may falter in another, making the selection process more complex. Algorithm Selection (AS) is, therefore, key, as it involves balancing multiple criteria such as accuracy, speed, and resource efficiency. The "No Free Lunch" theorem further emphasizes that no single algorithm is universally optimal, necessitating context-aware strategies. Despite advancements in the field, selecting the best algorithm remains challenging, particularly for complex problems with multifaceted performance metrics.

1.2 Problem Statement

Selecting the most suitable MOA for a specific problem is a challenging task due to the vast array of available algorithms, each designed to address different aspects of multi-objective problems under varying conditions. The diversity in algorithm design, especially within evolutionary algorithms (EAs), adds to this complexity. For instance,

platforms offering hundreds of MOAs highlighting the extensive range of options available, which, while providing numerous tools, also complicates the selection process. Key factors such as performance variability across different problem types, problem specificity, user expertise, and evaluation metrics further contribute to this challenge. As a result, selecting the most suitable MOA requires a strategic approach that carefully considers the problem's characteristics, the algorithm's capabilities, and the user's ability to implement and manage the chosen method effectively.

Current approaches to evaluating and selecting MOAs often rely on a limited set of performance indicators, which may not fully capture the intricacies of different optimization problems. Moreover, the stochastic nature of MOAs introduces variability in performance across different runs, making it difficult to draw definitive conclusions from a single set of results.

To address these challenges, there is a need for an evaluation framework that integrates multiple performance indicators, accounts for the inherent variability in MOA performance, and provides a systematic approach to selecting the most suitable algorithm for a given problem. This thesis seeks to fill this gap by developing a decision support system for Algorithm Selection in MOAs.

To achieve this, we turned to Multi-Criteria Decision Making (MCDM), a research field focused on developing systematic tools that aid in selecting the best solution from a fixed and known set of alternatives. MCDM includes various families of methods, each designed to systematically evaluate multiple conflicting criteria. One prominent family within MCDM comprises methods that aggregate utility functions using logical operators, offering a structured approach to decision-making. In this context, we will employ the Logic Scoring of Preferences (LSP) method, which facilitates a detailed evaluation of alternatives based on a combination of criteria, making it particularly well-suited for complex decision-making scenarios. This approach is central to our methodology, especially in the selection of optimization algorithms for Multi-Objective Evolutionary Algorithms (MOEAs).

1.3 Research Objectives

The primary objective of this research is to develop a decision support system for "Algorithm Selection using Logic Scoring of Preferences for Multi-Objective Evolutionary Algorithms (AS-LSP-MOEA)", for the evaluation, ranking, and selection of MOO, particularly Multi-Objective Evolutionary Algorithms (MOEAs), based on their performance across a wide range of multi-objective optimization problems. The specific objectives of this research are:

- **To conduct a state-of-the-art study on Algorithm Selection for MOO and MOEAs:** For establishing a foundational understanding of the existing methods, challenges, and advancements in this field. It ensures that the proposed solution addresses unresolved issues in algorithm selection for MOEAs.

- **To develop the AS-LSP-MOEA system:** A MCDM-based system that integrates multiple performance indicators and landscape features to provide a comprehensive evaluation of MOEAs for Online Algorithm Selection .
- **To systematically test and validate AS-LSP-MOEA:** Using a portfolio of state-of-the-art MOEAs on benchmark problems such as the ZDT and LSMOP families, as well as a complex real-world case study in Evolutionary Circuit Design (ECD).
- **To analyze the adaptability of AS-LSP-MOEA:** Across different problem complexities and landscape features, identifying the strengths and weaknesses of various MOEAs in different configurations.
- **To provide insights and guidelines:** For selecting the most appropriate MOEA for a given optimization problem, based on the comprehensive evaluation provided by AS-LSP-MOEA.

1.4 Methodology Overview

The research methodology involves several key steps:

1. **Development of the AS-LSP-MOEA System:** The AS-LSP-MOEA system is designed to evaluate MOEAs using a multi-criteria approach. It integrates various performance indicators such as convergence, diversity, and computational efficiency, as well as landscape features such as multi-modality and ruggedness. The system allows for the adjustment of criteria weights, making it adaptable to different problem scenarios.
2. **Benchmarking with Standard Problems:** The AS-LSP-MOEA system will be tested on a portfolio of MOEAs using standard benchmark problems from the ZDT and LSMOP families. These problems were chosen for their varying levels of complexity and their ability to represent a wide range of real-world optimization scenarios.
3. **Case Study in Evolutionary Circuit Design:** To validate the practical applicability of AS-LSP-MOEA, the system will be applied to a complex, large-scale real-world problem: Evolutionary Circuit Design (ECD). This case study highlights the challenges of large-scale optimization and the effectiveness of MOEAs in finding innovative solutions in high-dimensional spaces.
4. **Analysis and Evaluation:** The results are evaluated for consistency, robustness, and adaptability of AS-LSP-MOEA to various problem landscapes. The findings provide insights into the selection of MOEAs based on problem-specific characteristics.

1.5 Thesis Structure

After the **Introduction chapter**, this thesis is organized as follows:

- **Chapters 2, 3, and 4: Literature Review:** The literature review in this thesis is divided into three chapters. **Chapter 2** explores the theoretical foundations and methodologies of Multi-Criteria Decision Systems, detailing the development and application of methods like AHP, TOPSIS, and LSP, and their evolution through the integration of fuzzy logic and stochastic models for enhanced decision-making under uncertainty. **Chapter 3** delves into Multi-Objective Optimization, examining various algorithms such as Evolutionary Algorithms and Particle Swarm Optimization, with a focus on their effectiveness in addressing the challenges of convergence, diversity, and scalability in optimizing multiple objectives. **Chapter 4** addresses the critical issue of Algorithm Selection (AS), reviewing techniques like Per-instance Algorithm Selection and Online Algorithm Selection, and advocating for a more adaptive and robust framework to guide algorithm selection better in complex, real-world problems.

- **Chapter 5: Methodology and Development of AS-LSP-MOEA:** This chapter details the development of the AS-LSP-MOEA system, including its design principles, the selection of performance indicators, and the integration of landscape features. The chapter also discusses the configuration options and how AS-LSP-MOEA can adapt to different problem scenarios.

- **Chapter 6: Tests and Results:** This chapter presents the results of applying the AS-LSP-MOEA system to a portfolio of MOEAs across standard benchmark problems and the ECD case study. The chapter provides a detailed analysis of the results, highlighting the strengths and weaknesses of each algorithm under different configurations.

- **Chapter 7: Conclusions and Future Work:** The final chapter summarizes the key findings of the research, discusses the implications of the results, and suggests directions for future work. The chapter also reflects on the potential of AS-LSP-MOEA as a decision-support system for multi-objective optimization.

Chapter 2

Multi-Criteria Decision Systems

2.1 Introduction

Multi-Criteria Decision Analysis (MCDA) or Multi-Criteria Decision Making (MCDM) is a methodological approach used in complex decision-making contexts where multiple, often conflicting criteria must be considered. The primary objective of MCDM is to assist decision-makers in selecting the optimal alternative from a range of possible options, considering diverse qualitative and quantitative criteria [121].

MCDM is particularly beneficial in sectors where decision outcomes must balance varied, competing factors, such as engineering, economics, healthcare, and environmental management. The discipline encompasses various methods and tools that facilitate the evaluation and prioritization of different alternatives based on criteria pertinent to the decision-maker's specific situation.

One of the foundational aspects of MCDM involves the structuring of decision problems, which includes the explicit definition of criteria and alternatives and understanding the decision space where these elements interact. For instance, the decision space might be conceptualized as a set of feasible solutions, with the value of each solution being determined based on the defined criteria. Since each criterion contributes differently to the overall evaluation, MCDM requires an integrated approach to aggregate these disparate impacts into a unified, comprehensive evaluation measure [121].

2.2 MCDM concepts

The paradigms of decision aiding involve three basic concepts that underpin the various approaches and methods. These are Alternative, Criterion, and Problematic [52].

2.2.1 Alternative

In MCDA, an alternative, also referred to as a potential action, is considered the object of the decision process. It is defined as a potential choice that does not inherently include any feasibility or implementation assumptions. Alternatives are often mutually

exclusive due to the modeling approach, which simplifies the decision-making process by ensuring that only one alternative can be selected at a time. However, this exclusivity is not a necessary condition; in many decision contexts, multiple alternatives might be simultaneously viable .

2.2.2 Criterion

A criterion in MCDA is a means for evaluating and comparing potential actions from a specific point of view, which should be as clear and well-defined as possible. Each criterion is associated with effects or attributes relevant to that perspective, aiding in the structured comparison of alternatives. The performance of an alternative according to a criterion helps establish its desirability or suitability relative to others. Each method provides a different way to combine the criteria values into a single score that represents the overall efficacy or value of an option [121].

2.2.3 Problematic

This concept refers to how the decision-aiding process is envisioned in terms of the questions it must answer. Problematics in MCDA include deciding on the type of outcomes desired (e.g., choosing, sorting, ranking alternatives), how these outcomes are achieved, and the roles and methodologies involved in reaching these conclusions. The choice of a specific problematic influences the entire approach to the decision-making process, determining the methods and tools to be employed .

2.3 Methods

Multi-Criteria Decision Making (MCDM) methods are classified in different types, with three main approaches standing out:

- Relational models, based on pairwise comparisons of alternatives to rank or filter them, such as ELECTRE[113], PROMETHEE [19], Analytic Hierarchy Process (AHP)[114], and Consistent Fuzzy Preference Relations (CFPR)
- Reference-based models, which compare the alternatives with some ideal or anti-ideal solutions, with examples including TOPSIS[68], Grey Relational Analysis (GRA) [37], and VIKOR (VIseKriterijumska Optimizacija I Kompromisno Resenje)[101].
- Functional models based on aggregation operators that calculate an overall utility score for each alternative, such as the LSP (Logic Scoring of Preferences)[40] and WASPAS (Weighted Aggregated Sum Product ASsessment)[21].

In addition to these approaches, hybrid MCDM methods combine the strengths of different techniques, such as integrating AHP's detailed criteria weighting with TOPSIS's proximity assessment to the ideal solution. Furthermore, recent advancements in MCDM

have incorporated fuzzy logic and stochastic models to better address the ambiguity and uncertainty in decision-making processes.

The following sections will provide a brief overview of some of the most widely used methods within each of these categories. A more detailed introduction to LSP will be provided, as it is the method utilized in this thesis.

2.3.1 ELECTRE

ELECTRE stands for: ÉLimination Et Choix Traduisant la REalité (Elimination and Choice Translating Reality is a family of MCDM methods that use an outranking approach to evaluate and compare alternatives based on conflicting criteria. Developed in the 1960s by Bernard Roy and his colleagues [113], ELECTRE is particularly useful for situations when decisions involve multiple, often conflicting criteria that cannot be easily quantified or compared directly.

The ELECTRE methods are founded on several key principles:

1. ELECTRE establishes an **outranking relation** for each pair of alternatives, determining if one alternative is at least as good as another based on the criteria. This relation helps identify if one alternative is preferable or if equivalency or hesitation exists between the two.

2. The **Concordance and Discordance Indexes** measure the degree to which one alternative is preferable to another. The concordance index assesses positive alignments, while the discordance index evaluates negative divergences against criteria.

3. To manage judgment ambiguities, ELECTRE uses thresholds. The **Indifference Threshold** marks where differences between alternatives' criteria are negligible. Above the **Preference Threshold**, one alternative becomes evidently preferable. The **Veto Threshold** disqualifies an alternative with poor performance on a critical criterion.

4. The matrix formed from outranking relations is analyzed to derive decision recommendations, which involves filtering to resolve conflicts, identifying dominating sets, and possibly clustering alternatives.

The ability of ELECTRE to explicitly incorporate decision-maker preferences and accommodate veto considerations makes these methodologies highly adaptable and well-suited for complex decision-making scenarios.

2.3.2 TOPSIS

The Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) [68], developed by Hwang and Yoon in 1981, determines the optimal alternative by measuring its proximity to a positive ideal solution and distance from a negative ideal solution. The positive ideal solution aggregates the best potential values for each criterion, while the

negative represents the worst. This method ranks alternatives based on their relative closeness to these ideal solutions, using a comprehensive analysis of criteria data. This approach allows for broad adaptability without the necessity for independent criteria.

The process of applying TOPSIS involves several steps:

1. Constructing and normalizing the decision matrix to standardize the range of different criteria.
2. Creating a weighted normalized decision matrix to reflect the importance of each criterion.
3. Identifying the positive and negative ideal solutions based on the maximum and minimum values in the weighted decision matrix.
4. Calculating the distance of each alternative from these ideal solutions to determine how close or far they are.
5. Computing the relative closeness coefficient for each alternative, which helps in ranking the alternatives according to their suitability.

TOPSIS is appreciated for its straightforward implementation and its ability to deliver a clear hierarchy of options based on their relative performance against the most desirable and least desirable outcomes.

2.3.3 LSP

Logic Scoring of Preference (LSP) [40] is a quantitative method used for the evaluation, comparison, and ranking of systems where multiple criteria or user preferences must be considered simultaneously.

The method interprets system evaluation as a logic decision problem. It uses a hierarchical process of logic aggregation to generate a global preference score, reflecting the degree to which a system satisfies user requirements. The method allows for a model that can accommodate various logical relationships through different aggregation operators [42] as shown in Table 2.1

Core Concepts of LSP

- **Preferences:** In the LSP method, a preference represents the degree of satisfaction of a specific criterion from the user's perspective. These preferences are typically quantified on a scale from 0 (completely unsatisfactory) to 1 (completely satisfactory). This quantification allows for a standardized comparison of different system criteria.
- **Attribute and Criterion/Utility function** Once the preference values are defined for a raw system attribute, this becomes the "criterion" by which that attribute is evaluated. Utility functions translate these raw system attributes (like response time, distance, etc.) into a preference scores, usually within a standard

range (e.g., 0 to 1 or 0 to 100) using the defined criterion. This preference score indicates the degree of satisfaction of the user's requirements for that criterion.

- **Global Preference Score:** is the final output of the LSP method, summarizing the overall performance/suitability of the system in a single value. This score is derived by aggregating the individual preference scores of all relevant criteria, reflecting how well the

Structure of the LSP Method

- **Criteria Tree Construction:** is a hierarchical model that structures the system criteria into several levels by breaking down high-level system requirements into more detailed and specific criteria, which ultimately lead to measurable preference scores (*cf.* figure 2.1). Each leaf in the tree represents a raw system criterion, and necessitates the definition of the utility or user's preference values of this criterion.

  The parent node serve as both the point of aggregation and the definition of a more general criterion derived from its child nodes. It groups together several lower-level nodes (child nodes) and uses an aggregator or preference aggregation operator—a mathematical function or logic, such as weighted averages or Generalized Conjunction/Disjunction (GCD) functions—to combine the preferences of these child nodes into a single, unified score.

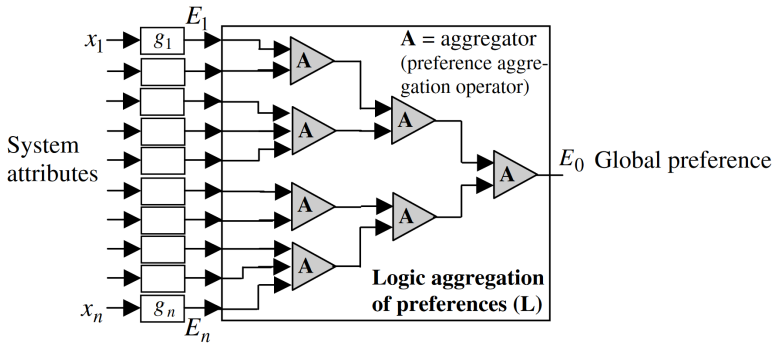


Figure 2.1: LSP Model (source [42])

- **Preference Scores and Aggregation:**

  Preference scores derived from elementary criteria indicate how well a particular system criterion meets the user's expectations. These scores are then aggregated to produce a global preference score, reflecting the system's overall performance.

  Once individual preference scores are calculated, they are aggregated to produce a global preference score. This process is guided by the Generalized Conjunction/Disjunction (GCD) function, which combines the individual preferences according to the logical relationships between the criteria.

Generalized Conjunction/Disjunction

The Generalized Conjunction/Disjunction (GCD) function [43] is a mathematical tool to aggregate multiple preference scores or criteria into a global preference score. The GCD function allows for a smooth transition between two extremes of logical aggregation: conjunction (AND) and disjunction (OR). This transition is controlled by a parameter, often denoted as r, which adjusts the degree of andness or orness in the aggregation process. The GCD function is often based on the Weighted Power Mean (WPM), which is a general formula that encompasses various specific means (harmonic, geometric, arithmetic, and quadratic) depending on the value of the exponent r:

$$E = \left(\sum_{i=1}^{n} W_i \cdot x_i^r \right)^{1/r}$$

where W_i are the weights assigned to each criterion, x_i are the values of the criteria, and r the degree of andness/orness in the aggregation.

The Table 2.1 displays the 17-levels GCD aggregators, their degree of conjunction/disjunction and the values of the corresponding exponent r in the WPM. The values of r also change according to the number of preference inputs to be aggregated. Here is what the aggregators represent in the context of GCD aggregators:

- **Simultaneity**
  - **Pure Conjunction ($r = -\infty$):** This is the strongest form of conjunction (AND), meaning that all criteria must be fully satisfied for the aggregation to result in full satisfaction.
  - **Partial Conjunction:**
    - **Hard Partial Conjunction (Very Strong to Medium Weak Conjunction) (r values between -9.06 to -0.148):** These r values, which are **highlighted** in the table, represent a stricter form of partial conjunction. In this context, the aggregation heavily emphasizes the need for all criteria to be satisfied to a high degree, although it does allow for some degree of replaceability between criteria. This characteristic is referred to as the property of **annihilation**, meaning that if one input in the aggregation is 0, the result of the aggregation will also be 0, regardless of the values of the other inputs. The property of annihilation is particularly important in cases where a single unsatisfactory criterion can override the overall evaluation, leading to a more conservative aggregation approach.
    - **Soft Partial Conjunction (Weak and Very Weak Conjunction) (r values between 0.261 to 0.619):** These r values represent a less strict form of conjunction, allowing more substitution between criteria. A higher r value means that the aggregation is more lenient, meaning that not all criteria need to be fully satisfied.

- **Neutrality (Arithmetic Mean)**
  - $r = 1$**:** This value represents the arithmetic mean of the inputs, where each criterion is considered equally without any bias toward conjunction or disjunction.
- **Replaceability**
  - **Partial Disjunction (Very Weak to Very Strong) (**r **values between** 0.619 **and** 9.521**):** These values represent different levels of disjunction (OR), where the satisfaction of one criterion can partially or completely compensate for the lack of satisfaction in others. This is a "soft" disjunction. A higher r value indicates a stronger tendency toward pure disjunction.
  - **Pure Disjunction (**$r = +\infty$**):** This is the hard form of disjunction, where if any criterion is fully satisfied, the whole aggregation results in full satisfaction.

Table 2.1: Values of parameter for the GCD aggregators (sources [42, 41])

				Inputs			
				2	**3**	**4**	**5**
GCD	**Simultaneity**	**Strongest**	C	$-\infty$	$-\infty$	$-\infty$	$-\infty$
		Very Strong	C++	-9.06	-7.639	-6.689	-6.013
		Strong	C+	-3.51	-3.114	-2.823	-2.606
		Medium Strong	C+-	-1.655	-1.55	-1.455	-1.38
		Medium	CA	-0.72	-0.732	-0.721	-0.707
		Medium Weak	C-+	-0.148	-0.208	-0.235	-0.251
		Weak	C-	0.261	0.192	0.153	0.129
		Very Weak	C–	0.619	0.573	0.546	0.526
	Neutrality		A	1	1	1	1
	Replaceability	**Very Weak**	D–	1.449	1.519	1.565	1.596
		Weak	D-	2.018	2.187	2.302	2.384
		Medium Weak	D-+	2.792	3.101	3.318	3.479
		Medium	DA	3.929	4.45	4.825	5.111
		Medium Strong	D+-	5.802	6.675	7.316	7.819
		Strong	D+	9.521	11.095	12.27	13.235
		Very Strong	D++	20.63	24.3	27.11	30.09
		Strongest	D	$+\infty$	$+\infty$	$+\infty$	$+\infty$

The Uniform GCD aggregators (UGCD) [94] are a simplified and symmetric variant of the GCD function. UGCD are designed to treat all inputs uniformly, applying the same thresholds for andness and orness across all criteria. UGCD are often used in situations where a uniform approach to aggregation is sufficient and where the complexity

of GCD is unnecessary. It is particularly suited to scenarios where the evaluation criteria are considered equally important, or where the goal is to maintain a consistent approach to aggregation across all inputs.

Mandatory/Optional Criterion

In the LSP method with Weighted Power Mean (WPM), distinguishing between mandatory and optional criteria is crucial for accurate system evaluation and selection. Mandatory criteria are core requirements that must be met for a solution to be viable, such as system reliability and security, where failure to meet these standards disqualifies the solution. Optional criteria enhance the system's quality or usability but are not essential to its primary function, such as additional features or user experience improvements. The inclusion of optional criteria allows for a more nuanced evaluation, particularly when multiple solutions meet the mandatory requirements.

The integration of Conjunctive Partial Absorption (CPA) and Disjunctive Partial Absorption (DPA) operators within the LSP framework further refines this evaluation. CPA ensures that mandatory criteria are prioritized, penalizing the overall score if these are not fully satisfied, while also rewarding systems that meet both mandatory and optional criteria. DPA, on the other hand, emphasizes sufficient criteria that can independently secure a high score, with optional criteria providing additional benefits but not critical enhancements. When these operators are used within the WPM aggregation process, which considers various criteria based on assigned weights and an exponent parameter, they enable a balanced and flexible evaluation.

2.4 Tools

The International Society on Multiple Criteria Decision Making maintains a comprehensive list of software tools related to MCDM on their official website[1]. This list includes a variety of free, semi-commercial, and commercial software that support decision-makers in evaluating, comparing, and ranking alternatives across multiple criteria. These tools cover a wide range of MCDM methods, providing both specialized and flexible approaches to decision analysis.

For those utilizing the ELECTRE family of methods, MCDA-ULaval [92] is a free software specifically designed to support these techniques. In contrast, the AHP is supported by the free tool PriEsT [110], which handles AHP calculations and offers clear visualizations of priority scales and consistency ratios..

In addition to these specialized tools, platforms like the MCDA Package for R [15], MCDA Calculator [91], and PyMCDM [111] integrate various MCDM methodologies, such as TOPSIS, VIKOR, and Multi-Attribute Value Theory (MAVT). These packages offer flexibility and adaptability.

Several MCDM tools are designed to offer multiple methods, providing comprehensive solutions for complex evaluations. For example, D-Sight [39], a commercial soft-

[1] https://www.mcdmsociety.org/content/software-related-mcdm-0

ware, supports the PROMETHEE methods alongside Multi-Attribute Utility Theory (MAUT), offering robust functionality for decision analysis. Similarly, DEFINITE/BOSDA [35], another commercial tool, includes a range of methods such as Weighted Summation, AHP, ELECTRE II, and others, catering to a variety of decision-making scenarios.

LSP-NT (Logic Scoring of Preference Network Tools) [88] is designed specifically for the implementation of the LSP method. This tool supports advanced aggregation functions, such as UGCD.15 in its demo version, and provides features like cost/preference analysis, criteria and suitability tables, and other facilities that assist in the system evaluation and decision-making. It is a valuable resource for users requiring detailed and customizable decision support.

2.5 Conclusion

This chapter underscores the critical role of structured decision-making frameworks to manage complexity across various sectors, providing systematic approaches for comparing and prioritizing alternatives. MCDM methods, such as AHP, ELECTRE, and LSP, assist decision-makers in navigating the complexities of conflicting criteria and uncertainty. The flexibility of these methods, combined with specialized tools like MCDA-ULaval, PriEsT, and LSP-NT, allows for the customization of decision models to fit specific needs, whether in engineering, healthcare, or environmental management. Moreover, the evolution of MCDM through hybrid methods and the incorporation of fuzzy logic and stochastic models reflects ongoing advancements in the field, enhancing the precision and robustness of decision support. Comprehensive platforms, such as the MCDA Package for R and D-Sight exemplify technological advancements, providing flexible and user-friendly solutions that simplify the aggregation and evaluation of diverse criteria. These tools support the integration of both quantitative and qualitative data, ensuring well-informed, transparent, and aligned decisions. As the complexity of decision-making grows, the ongoing evolution of MCDM methodologies and the development of sophisticated software tools will remain critical in enabling effective and efficient decision processes, ultimately contributing to more balanced, objective, and well-supported outcomes.

Chapter 3

Multi-Objective Optimization

3.1 Introduction

While Multi-Criteria Decision Making/Analysis (MCDM/A) is primarily concerned with discrete decision-making problems where alternatives are evaluated based on predefined criteria, Multi-Objective Optimization (MOO) deals with scenarios where the space of possible solutions can be either discrete or continuous. In MOO, the solution space is defined by the objective functions and constraints, rather than by discrete criterion values, making it inherently more complex. Unlike in MCDM/A, where all possible alternatives are pre-defined and finite, MOO involves exploring a continuous and often vast solution space. The exact set of optimal solutions—represented by the Pareto front—is not known in advance and must be discovered through optimization algorithms, which seek to identify the best trade-offs among conflicting objectives. Thus, while the solution space in MOO is theoretically defined, it remains practically unknown until the optimization process is carried out.

Despite the advancements, MOO remains a challenging domain due to the inherent trade-offs and the computational complexity of evaluating multiple objectives simultaneously. Moreover, the choice of appropriate performance metrics is essential, as different metrics can lead to different interpretations of an algorithm's performance. Understanding these nuances is essential for effectively applying MOO techniques to real-world problems.

3.2 Multi-Objective Optimization Problems

Multi-Objective Optimization Problems (MOP) are a class of optimization problems that involve multiple conflicting objectives that need to be optimized simultaneously. For example, in designing a house, one might want to minimize the cost while maximizing comfort and energy efficiency. These objectives are often in conflict; for instance, increasing the number of luxury features might increase the cost or reduce energy efficiency. A solution to a MOP is typically not a single optimal solution but a set of solutions known as the Pareto front.

A **Pareto front** [126] represents the set of non-dominated solutions, where no single objective can be improved without causing a degradation in at least one other objective. In other words, the Pareto front consists of all solutions for which there is no other solution that is better in every objective. On the other hand, an approximation [122] set refers to the set of solutions generated by an optimization algorithm that approximates the Pareto optimal front. Since it is often infeasible to compute the exact Pareto front due to the complexity and number of objectives, the algorithm aims to produce a set of solutions that are as close as possible to the true Pareto front. This set serves as an approximation of the optimal trade-offs among the conflicting objectives, providing decision-makers with a range of potential solutions.

A MOP can be generally formulated as follows [122]:

$$\begin{aligned} \text{minimize} \quad & \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{subject to} \quad & \mathbf{x} \in \Omega \end{aligned} \tag{3.1}$$

where

- $\mathbf{F}(\mathbf{x})$ represents the vector of objective functions $f_1, f_2, \ldots, f_k$,
- $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is the vector of decision variables,
- Ω is the feasible set defined by the problem constraints.

The goal is to approximate this Pareto front as closely and as globally as possible.

3.3 Performance Metrics

A performance metric in the context of MOO is a function or set of functions that provide quantitative measures to evaluate the quality of solutions produced by optimization algorithms. They help determine how well an algorithm approximates the Pareto optimal front and ensures diversity among the solutions [145]. Moreover, they serve as a comparison tool to assess and contrast the strengths and weaknesses of different algorithms.

3.3.1 Classification of Performance Metrics

Convergence metrics

Convergence metrics evaluate how close the solutions produced by an optimization algorithm are to the Pareto optimal front. For example, the Generational Distance (GD) [127] metric calculates the average distance of the solutions in the approximation set from the nearest point on the Pareto front, with a lower GD value indicating better convergence. Another convergence metric, the Epsilon Indicator [145], measures the smallest distance by which the Pareto front must be translated to dominate the entire approximation set, providing a direct measure of how close the solutions are to the true Pareto front. These convergence metrics are critical because they directly reflect the accuracy of the solutions in representing the optimal trade-offs among the objectives.

Diversity metrics

Diversity metrics assess the spread and distribution of solutions across the Pareto front, ensuring that the solutions cover a wide range of trade-offs and provide decision-makers with a comprehensive set of options. For example, the Spacing Metric (Sp) [117] evaluates the uniformity of solution distribution by measuring the variance in the distance between neighboring solutions, with a lower value indicating a more uniform distribution. Other metrics can assess both diversity and convergence, such as the Hypervolume Indicator (HV) [143], which measures the volume in the objective space dominated by the solutions in the approximation set and bounded by a reference point. A higher hypervolume value indicates a better result, reflecting both superior convergence and diversity.

Cardinality metrics

Cardinality metrics help assess whether the algorithm provides a sufficient number of solutions to cover the entire Pareto front, ensuring a detailed exploration of possible trade-offs (Zitzler et al., 2003). While not directly related to the quality of the solutions, cardinality is important for understanding the richness of the solution set. For example, the Overall Non-dominated Vector Generation (ONVG) [127] metric counts the number of non-dominated solutions generated by an algorithm, providing insight into the algorithm's ability to explore the solution space.

While performance metrics are indispensable tools in evaluating and comparing MOA, they are not without their limitations. Each metric comes with its own set of drawbacks, which can sometimes lead to misleading conclusions if not carefully considered.

- **Bias Towards Specific Aspects:** Some metrics, like the Generational Distance, heavily emphasize convergence to the Pareto optimal front. While this is useful for assessing how close the solutions are to the optimal front, it can lead to a bias where algorithms that produce closely clustered solutions near the front are favored, even if those solutions lack diversity. A study by Zitzler et al. [145] highlighted that the use of convergence metrics alone could result in algorithms being judged as superior simply because they generate solutions close to the Pareto front, neglecting how well these solutions are spread across the front.

- **Inadequate Representation of Diversity:** Metrics like the Spacing Metric aim to evaluate how uniformly solutions are spread across the Pareto front. However, these metrics can sometimes fail to capture the true diversity of solutions, especially in problems with complex Pareto fronts where solutions might be well-distributed in some regions but sparse in others [126].

- **Overemphasis on Cardinality:** Metrics like the ONVG provide a straightforward count of solutions, but this can be misleading. A high cardinality might suggest that an algorithm is thoroughly exploring the Pareto front, but without

proper diversity and convergence, these additional solutions may be redundant or similar and add little value [100]. This is especially problematic in cases where the Pareto front is large and complex, as a simple count does not provide insight into the actual utility of the solutions [126].

- **Complexity and Computation:** Some metrics, particularly those that require calculations over the entire Pareto front, can be computationally intensive. This makes them less practical for large-scale problems or real-time applications [126]. The Hypervolume Indicator is an example of a metric that, while effective, can become computationally prohibitive as the number of objectives and solutions increases. This limitation makes it less suitable for problems with high-dimensional objective spaces.

- **Context-Dependency:** The effectiveness of certain metrics can be highly dependent on the specific problem being addressed. A metric that works well for one type of optimization problem might be less suitable for another, leading to inconsistent evaluations across different problem domains [100]. The Epsilon Indicator, for instance, can be very sensitive to scaling and the choice of reference points, making it problematic for problems where objectives have vastly different scales [126].

The drawbacks of the performance metrics must be carefully considered. A balanced approach that combines multiple metrics and considers their limitations is essential for a more accurate assessment of algorithm performance [145].

3.4 Taxonomy of MOO Approaches

The taxonomy of MOO methods outlined by Hwang and Masud in 1979 [67] offers a structured framework for classifying various approaches based on how they incorporate decision-maker preferences and handle the complexity of multiple conflicting objectives. This classification aids in narrowing down algorithm choices to those that best balance different performance criteria, thereby simplifying the decision-making process and reducing problem complexity [108, 83]. While this taxonomy serves as a useful starting point for decision-makers seeking to identify the most appropriate method, it is too simplistic to reliably select the optimal algorithm on its own, necessitating further refinement and additional criteria for a more accurate selection process.

The classification categorizes into four main groups: a priori, a posteriori, interactive, and hybtid approaches.

3.4.1 A priori methods

These methods require the decision-maker to specify their preferences before the optimization process starts. This early specification directs the search towards a particular

region of the solution space, potentially simplifying the decision-making process by focusing only on the most relevant solutions.

The a priori methods transform the MOO problem into a single-objective problem. This transformation typically involves two key steps: firstly, assigning appropriate weights to different objectives to reflect their relative importance and secondly, consolidating these multiple objectives into a singular and composite objective.

The ϵ-constraint Method [45] selects one objective for minimization and transforms others into constraints with specified bounds, exploring various feasible regions to identify all efficient solutions.

Benson's approach [12] iteratively updates bounds on the objectives to generate an outer approximation of the Pareto front before any decisions are made or outcomes are observed. It is closely related to the ϵ-constraint method in that it incorporates similar constraints to identify efficient solutions with conflicting objectives.

3.4.2 A posteriori methods

These methods do not require any preference information beforehand, allowing for an exhaustive exploration of the solution space. Decision-makers are presented with the Pareto optimal set after the completion of the optimization, from which they can select the most suitable solution based on a comprehensive view of the trade-offs involved.

The a posteriori approach is typically categorized into two main classes: scalarization methods and Pareto-based methods.

Scalarization Methods

These methods involve transforming the multi-objective problem into a single-objective problem using objective conversion techniques such as the Weighted Sum Approach and Goal Programming. By doing this, the optimization problem is resolved multiple times with different settings or weights, which results in a collection of solutions that form the Pareto front.

The Weighted Sum approach [95] combines objectives using weighted averages, emphasizing the relative importance of each objective according to predefined weights. This approach can be utilized both a priori, to set goals based on theoretical or expected conditions, and a posteriori, adjusting the weights based on outcomes or new insights. Goal Programming [71] differs by focusing on achieving specific target levels for each objective, minimizing deviations from these targets and treating each deviation as a separate minimization problem. Similarly, Goal Programming can be employed a priori to establish target levels before data collection, or a posteriori to adjust targets based on actual outcomes.

Pareto-based Approaches

Pareto-based approaches focus on identifying and evaluating the trade-offs among various objectives by exploring the Pareto front. These methods utilize the concept of Pareto

dominance to directly search for multiple optimal solutions, seeking to find solutions where no single objective can be improved without worsening another. Unlike scalarization methods, Pareto-based methods do not consolidate the objectives into a single scalar function but instead generate a set of solutions that are non-dominated with respect to each other. This process is concurrent, often leveraging evolutionary algorithms or other population-based approaches to efficiently populate the Pareto front.

Bio-inspired algorithms are frequently employed to generate a diverse set of solutions that effectively populate the Pareto front. Multi-Objective Evolutionary Algorithms (MOEAs) like NSGA-II [32], SPEA2 [144], and MOEA/D [139] harness evolutionary principles such as mutation and selection to efficiently balance conflicting objectives and produce a range of viable solutions. Similarly, Swarm Intelligence algorithms such as Multi-Objective Particle Swarm Optimization (MOPSO) [27] and Multi-Objective Ant Colony Optimization (MOACO) [38] leverage collective behaviors observed in nature, enabling exhaustive exploration of solution spaces and the identification of optimally balanced solutions. Additionally, Artificial Immune Systems (MOIA) [28] emulate the human immune system's diversity-preserving capability, making them ideal for maintaining varied sets of solutions along the Pareto front.

While these approaches are commonly used in an a posteriori manner, they can also be effectively adapted for a priori approaches, where specific goals or constraints are defined upfront.

3.4.3 Interactive methods

Interactive methods in multi-objective optimization facilitate dynamic engagement between the decision-maker and the optimization process, allowing for adjustments in preferences based on intermediate outcomes. This engagement leads to solutions that more closely reflect the decision-maker's evolving objectives.

One classic example is the Geoffrion-Dyer-Feinberg method (GDF) [56], where the decision-maker interacts directly with the optimization process, allowing the decision-maker to iteratively refine the preferences by evaluating the solutions generated by the algorithm.

Another significant interactive method is the Synchronous Nondominated Sorting Algorithm (SNSGA-II) [50], which extends the well-known NSGA-II algorithm. SNSGA-II integrates a real-time interaction mechanism that empowers decision-makers to influence the evolutionary process actively by adjusting the population of solutions according to their preferences. Thus, the decision-makers can steer the search towards more favorable regions of the solution space.

3.4.4 Hybrid methods

Hybrid methods in multi-objective optimization strategically combine different optimization approaches to exploit their individual strengths while mitigating their limitations. For example, a typical hybrid approach might begin with a scalarization technique, such as the Weighted Sum, to initially narrow down the search area within the vast Pareto

front. Subsequently, a Pareto-based method like NSGA-II is utilized to refine and deepen the search in the targeted regions, ensuring a thorough exploration and a well-distributed set of solutions. Additionally, integrating interactive methods allows decision-makers to dynamically engage with the optimization process, adjusting preferences based on intermediate results to steer the solution development in real-time.

3.4.5 Strengths and Limitations

The table 3.1 provides a summary of the strengths and limitations associated with each class of methods. This overview helps to clarify which approaches might be best suited for specific types of decision-making scenarios, facilitating a better understanding of how each class can be effectively applied.

Table 3.1: Strengths and Limitations of MOO Methods

Methods	**Strengths**	**Limitations**
A Priori	- Focused search towards specific interests in the solution space - Efficient use of single-objective optimization techniques	- Lack of flexibility for exploring outside predefined scopes - Dependency on clear and accurate initial preferences
A Posteriori	- Comprehensive exploration of the Pareto front - Flexibility in decision making after generating solutions	- Potential overwhelm due to large number of solutions - High computational intensity for complex problems
Pareto-based	- Effective for non-dominated solution generation - Suitable for complex problems with conflicting objectives	- Computationally demanding for maintaining diverse solutions - Algorithm complexity increases with problem dimensions
Interactive	- Dynamic integration of decision-maker's changing preferences - Increased engagement and potentially more satisfactory outcomes	- Dependency on decision-maker's timely and accurate feedback - Extended process duration due to iterative nature

While a priori methods offer targeted efficiency, their lack of flexibility can be a drawback. Conversely, a posteriori methods provide extensive insights but can overwhelm decision-makers and demand substantial computational resources. Pareto-based approaches excel in generating diverse, optimal solutions for complex challenges, requiring significant computational effort and intricate algorithm management. Interactive methods, by adapting to evolving preferences, foster a dynamic decision-making process but may extend its duration due to the need for continuous input. Therefore, selecting an appropriate MOO method must take into account the specific requirements of the problem, the available resources, and the desired outcomes to ensure optimal performance and effectiveness.

3.5 Problem Landscape Features

Problem landscape features refer to the intrinsic characteristics of the search space within an optimization problem that significantly impact the behavior and performance of optimization algorithms. These features offer critical insights into the problem's structure, complexity, and difficulty, which are essential for designing or selecting the most appropriate algorithms for effective problem-solving [147]. To facilitate this understanding, Exploratory Landscape Analysis (ELA) [93] is introduced as a powerful technique that systematically analyzes and characterizes the landscape of an optimization problem. ELA utilizes a small, strategically selected sample of function evaluations to automatically extract relevant features of the landscape.

- **Modality** refers to the number and distribution of local optima within the search space. A unimodal landscape has a single global optimum and no local optima, making it relatively easier for optimization algorithms to converge on the best solution. In contrast, a multi-modal landscape contains multiple local optima, which can trap optimization algorithms and make the search process more challenging [93].
- **Ruggedness** describes the degree of variability or "roughness" in the fitness values across the search space. A landscape with high ruggedness is characterized by many peaks and valleys, indicating numerous local optima. This increases the difficulty for optimization algorithms to find the global optimum. On the other hand, low ruggedness implies a smoother landscape with fewer variations, facilitating a more straightforward search process [85]. Landscape Smoothing involves techniques used to modify or smooth the landscape to make it easier to find the global optimum. Smoothing can reduce ruggedness or increase basin sizes, thus aiding in the search process [89].
- **Neutrality** in a landscape refers to the presence of flat regions where multiple solutions have the same or very similar fitness values. Neutral landscapes can cause optimization algorithms to stagnate because it becomes challenging to identify a clear direction of improvement, leading to potential inefficiencies in the search process [2].
- **Basin Size and Shape:** The size and shape of basins—regions around local optima that attract solutions during the search process—are critical landscape features. Large basins are generally easier for algorithms to identify and converge upon, while narrow or irregular basins can be more difficult to detect, increasing the complexity of the search process [119].
- **Global Structure** of the landscape can significantly impact the optimization process. A funnel structure is a favorable global feature where multiple local optima lead towards a single global optimum, typically simplifying the problem. Conversely, deceptive structures mislead the search process by directing it away from the global optimum, increasing the problem's difficulty.

- **Constraint Structure:** The nature and distribution of constraints within the problem space define the constraint structure. The feasibility landscape—the distribution of feasible versus infeasible regions—can be particularly challenging when large portions of the search space are infeasible, requiring sophisticated algorithms to navigate effectively .

- **Pareto Front Characteristics (for Multi-Objective Problems):** In MOO,the shape of the Pareto front (convex, concave, or discontinuous) affects the difficulty of finding trade-offs between objectives. The spread and distribution of solutions along the Pareto front influence the effectiveness of optimization algorithms, while disconnected Pareto fronts add to the complexity by requiring algorithms to cover disjoint regions [93].

- **Dimensionality** refers to the number of decision variables and objectives within the problem. Higher dimensionality increases the complexity of the search space, making it more challenging for optimization algorithms to explore effectively and efficiently [72].

- **Scalability** describes how the problem landscape changes as the size of the problem (in terms of variables and constraints) increases. Some landscapes become exponentially more difficult as the problem size grows, necessitating scalable algorithms that can handle these complexities [93].

- **Evolvability** is the ability of small changes in decision variables to produce improvements in the fitness value. High evolvability indicates that the landscape is conducive to iterative improvement, while low evolvability suggests that small changes often do not lead to significant improvements, making the search process more difficult [85].

- **Constraint Violations:** The violation landscape represents the degree to which solutions violate constraints, which is particularly important in constrained optimization problems. Understanding this landscape is crucial for developing algorithms that can efficiently navigate and find feasible solutions .

- **Sampling and Distribution:** The sampling method used to explore the search space can significantly influence the landscape features observed. Clustering refers to the tendency of solutions to group in certain regions of the search space, which may indicate the presence of attractors or basins that influence the search dynamics [89].

Liefooghe *et al.* [85] propose a categorization of problem landscape features into four distinct classes: global, multi-modality, evolvability, and ruggedness. This classification is utilized in our algorithm selection system (See §5.3.3 for more details).

3.6 Challenges in Selecting a MOA

Selecting an appropriate multi-objective algorithm (MOA) for a particular problem is indeed a challenging task, especially given the vast array of algorithms available across various classes of approaches. The complexity arises not only from the sheer number of algorithms but also from the diversity in their design, each tailored to tackle specific aspects of MOPs under different conditions and constraints.

For instance, within the domain of EAs alone, there is a staggering variety of methods developed to efficiently navigate and solve MOPs. For example, the PlatEMO platform [109], a MATLAB tool designed specifically for EAs, illustrates this diversity well. PlatEMO includes a collection of 237 MOEAs in its version 4.7, available on GitHub[1]. This wide array of algorithms underscores the depth and breadth of research and development in this area, offering a plethora of tools but also posing a significant challenge in selecting the most suitable algorithm.

- **Performance Variability**: MOAs can exhibit significant performance differences across various problem types. An algorithm that excels in one setting may perform poorly in another, necessitating a thorough understanding of both the specific problem characteristics and the algorithm's capabilities. This variability demands careful consideration and often empirical testing to determine the best fit for a given problem.

- **Problem Specificity**: MOPs can greatly differ in their objective functions, constraints, dimensionality, and the nature of decision variables. For instance, an algorithm optimized for problems with continuous, smooth landscapes may not perform as well with problems characterized by discrete variables and non-linear, complex landscapes. This specificity requires a nuanced approach to algorithm selection, ensuring compatibility with the problem's unique criteria.

- **User Expertise and Preferences**: The selection of an MOA often hinges on the user's expertise and familiarity with specific algorithms, as well as the available computational resources. Additionally, user preferences regarding the transparency of the algorithm and the nature of the solutions it generates—such as the diversity of solutions along the Pareto front— play a critical role in the choice of algorithm.

- **Evaluation Metrics**: Choosing the most suitable MOA requires the use of appropriate evaluation metrics, such as convergence to the Pareto front, diversity of solutions, computational efficiency, and user-friendliness. The prioritization of these metrics may vary depending on the specific goals and constraints of the project, further complicating the selection process.

Finally, the task of choosing the most appropriate MOA is daunting due to the vast selection available. This variety, while beneficial, demands a strategic approach to

[1] https://github.com/BIMK/PlatEMO

algorithm selection, incorporating considerations of problem specifics, algorithm performance, and user capacity to manage and implement the chosen methods effectively.

3.7 Conclusion

In this chapter, we explored the core principles and challenges of MOO, focusing on the importance of the Pareto front and approximation sets in addressing problems with multiple conflicting objectives. We examined key performance metrics, such as convergence, diversity, and cardinality, that are essential for evaluating how well algorithms approximate the Pareto front and maintain a diverse set of solutions. While these metrics are essential, their limitations—like potential biases and computational complexity—highlight the need for a balanced approach in algorithm evaluation.

A significant discussion centered on problem landscape features, which play a critical role in influencing the performance of optimization algorithms. Understanding these features is vital for selecting the most appropriate algorithm for a given problem, as they affect how effectively an algorithm can navigate the solution space.

We conclude that algorithm selection is significantly influenced by the problem's landscape characteristics, the performance metrics used, and the decision-maker's goals. Recognizing the importance of these factors, we apply this approach in our system to systematically select the most appropriate optimization algorithms, ensuring that they are well-aligned with the specific challenges of the problem at hand and the objectives of the decision-maker.

In the next chapter, we will explore the "Algorithm Selection" paradigm, reviewing relevant work both in general and specifically within the context of multi-objective optimization. This exploration will provide a foundation for understanding how to effectively match algorithms to problem characteristics in such challenging field.

Chapter 4

Algorithm Selection

4.1 Introduction

Algorithm selection refers to the process of choosing the best algorithm from a set of available algorithms for solving a specific problem instance. This concept is essential in areas where different algorithms might excel under different circumstances, based on the characteristics of the problem at hand. The concept was first formalized by John Rice in 1976 [112], and it has since become a fundamental strategy in various domains, particularly in computational sciences where multiple algorithms exist for the same problem type, each with unique strengths and weaknesses.

In the context of multi-objective optimization, the Algorithm selection faces the complexities and strategic importance of choosing appropriate algorithms from a portfolio, especially when multiple objectives are involved [123]. The challenge intensifies as different algorithms may excel in optimizing one objective but perform poorly on others. The concept of a portfolio is crucial, consisting of a diverse set of algorithms that cater to varying optimization challenges [87]. This diversity is vital because no single algorithm typically outperforms all others across every possible scenario.

Selection often depends on specific features of the problem instances, which help predict which algorithm might perform best for a given problem [87]. This process involves computational models that assess and predict each algorithm's performance based on these features. Performance in multi-objective scenarios is not straightforward but involves a set of metrics that need to be optimized simultaneously, including solution quality, computational time, and resource consumption [123].

Both empirical data from benchmark tests and theoretical insights from statistical or machine learning methods are used to guide the selection process [87]. The need for adaptive strategies is highlighted, strategies that dynamically select the best algorithm as more information becomes available or as problem parameters change [62]. Additionally, any methodology for algorithm selection, especially in multi-objective contexts, must be validated across a range of scenarios to ensure it is robust and reliable, often involving comparisons against benchmarks or known standards [62].

4.2 Formulation of the Algorithm Selection Problem

In the domain of AS, the problem can be framed as managing a portfolio $A = \{a_1, a_2, \ldots, a_n\}$ of potential algorithms to address a specific instance p of a problem called a benchmark. The task of the selection function F is to identify the most appropriate algorithm or subset of algorithms from the portfolio that maximizes performance according to predefined criteria [112].

4.2.1 Criteria in Algorithm Selection

The selection function F operates by leveraging a diverse array of criteria, ranging from a few simple ones to a complex multitude, thus impacting the computational requirements. These criteria fall into two broad categories: **Static Criteria**, which are constant throughout the runtime and include inherent characteristics of the problem instance or the algorithms; and **Dynamic Criteria**, which evolve during the execution and may encompass metrics such as the speed of progress or changes within the search space dynamics.

Typically, the criteria employed for making selection decisions encompass attributes of the problem instance, properties of the algorithms, characteristics of the search space, and metrics related to the progress of the algorithms (e.g., speed of convergence).

4.2.2 Portfolio-Based Algorithm Selection

As computational capabilities have advanced, the focus in algorithm selection has shifted towards finding a set of algorithms S, that maximize a set of criteria that evaluate the performance of the algorithm under different features[73]. The set can be composed of different algorithms or by multiple different instances of the same algorithm using different parameter values.

The function F aims to optimize a criterion C, such as runtime or solution quality, based on the current problem features and algorithm performance, as follows:

$$S = \arg\max_{S} F(f_1, f_2, \ldots, f_n)$$

where $C(S)$ represents the criteria (e.g., Solution Quality, Hypervolume,) of employing the selected set S of algorithms given the available resources and strategies [105].

The problem complexity increases when algorithms depend heavily on initial conditions or random strategies, such as those seen in MOPs. Sharing information among algorithms within the portfolio can introduce complexities but also create synergies that may lead to superior solutions, particularly in scenarios that benefit from a mix of global and local search strategies.

4.3 Offline Selection

The offline algorithm selection is made without real-time adaptation to changes in problem characteristics or algorithm performance during execution. Instead, the selection relies on a static decision model built from a predefined dataset of problem instances and corresponding algorithm performances. Most existing methodologies in offline algorithm selection primarily involve choosing between single-objective optimization algorithms, with less focus on MOAs [70].

In the context of boolean satisfiability problem (SAT), statistical models and clustering techniques are used to select algorithms that best match variable characteristics and instance features, enhancing solution accuracy and efficiency. Notable works include SATZilla [133], a sophisticated algorithm selection method for the SAT problem. SATzilla employs empirical hardness models to construct per-instance algorithm portfolios, selecting the most effective solver based on the characteristics of specific SAT instances. SATzilla was tested in the 2007 SAT Competition, showcasing its effectiveness by winning several medals.

Lindauer et al. [86] present AutoFolio, another notable approach for automating the configuration of algorithm selectors, which are crucial for determining the most efficient algorithm for a given problem instance in various AI problem areas like SAT, CSP, and MAXSAT [4]. AutoFolio utilizes algorithm configurators like Sequential Model-based Algorithm Configuration (SMAC) [66] and ParamILS [65] to optimize the selection process by automatically determining the best configurations, thereby improving performance across different algorithm selection scenarios.

Hydra [135] represents a significant advancement in AS by integrating automated algorithm configuration with portfolio-based algorithm selection. This method iteratively configures new algorithms to construct a set of solvers that exhibit complementary strengths, effectively merging the benefits of automated configuration—which requires no domain knowledge and produces a single solver—and portfolio-based selection, which utilizes variation among a diverse set of solvers. Tested on stochastic local search algorithms for SAT problems, Hydra demonstrates considerable potential to improve AS processes. It automates configuration and combines it with strategic portfolio usage, suggesting its applicability across various computational domains.

Kroer and Malitsky [80] present an approach to refining the process of algorithm configuration using Instance-Specific Algorithm Configuration (ISAC). The ISAC approach essentially selects and tunes algorithms based on the characteristics of each problem instance, grouped by similarity into clusters. However, a challenge has been reliance on a comprehensive feature set to categorize instances, which can be suboptimal or overly broad.

For machine learning and classification tasks, techniques like nearest neighbor, decision trees, and bayesian networks help in ranking and selecting the most effective algorithms based on instance and algorithm features. This is illustrated in the works of Mantovani *et al.* [90] and Abdulrahman *et al.* [1], where these methods are employed to determine the most suitable algorithms, leading to improved predictive accuracy and

resource management.

Another significant development was presented in [77], focusing on per-run algorithm selection with warm-starting using trajectory-based features. This method leverages a predefined set of "cheap" features, which do not require additional function evaluations, thus minimizing computational overhead while maintaining robustness in feature selection and algorithm performance prediction.

In fields like vehicle routing and job shop scheduling, genetic algorithms and Bayesian classifiers are employed to optimize routing decisions and scheduling tasks. For instance, Pavelski et al. [106] utilized decision trees and gradient boosting for flowshop scheduling, while Blet et al. [17] applied M5P regression in constraint programming for scheduling to enhance operational efficiency.

A notable study [131] introduced an AS on Large Language Models (AS-LLM), which utilized pre-trained LLMs to enhance the representation and selection of algorithms by analyzing their code snippets or descriptive texts. The primary focus is on overcoming the limitations of traditional algorithm selection methods that rely heavily on problem features, while algorithm features are often overlooked. This study leverages LLMs to extract high-dimensional and discriminative features from algorithm code and description texts, which are then used to better match algorithms to specific problems.

Furthermore, research in 2024 has explored new dimensions of optimization for deep learning, particularly focusing on improving the computational efficiency and effectiveness of back-propagation techniques [57] and using meta-learning and pre-trained deep convolution neural networks [30]. These improvements cater specifically to large-scale optimization tasks that are prevalent in deep learning applications, highlighting a critical intersection of algorithm selection and practical implementation in computational neural networks.

The techniques highlighted in the preceding paragraphs offer an overview of the methodologies employed in offline algorithm selection, encompassing statistical modeling, clustering techniques, and automated algorithm configuration. Through exemplars like SATzilla, AutoFolio, Hydra, and ISAC, these methodologies illustrate the dynamic evolution of offline algorithm selection, spanning from the utilization of empirical hardness models to the intricate precision of instance-specific algorithm configuration. Moreover, recent strides in leveraging LLMs serve to underscore the intersection of AS with emerging computational paradigms.

4.4 Online Selection

Online algorithm selection (OAS) is designed to dynamically select the most suitable algorithm from a pool of candidates based on real-time problem characteristics. This approach is particularly valuable in scenarios where no single algorithm consistently outperforms others across all situations. For instance, in machine learning applications, different algorithms may excel under varying data distributions or computational constraints. OAS addresses this variability by employing mechanisms that assess the performance of algorithms on-the-fly and select the optimal one for the current context

[78].

The effectiveness of online algorithm selection is underscored by its ability to adapt to changing environments and its application in diverse fields such as real-time bidding in online advertising, where decisions must be made quickly and efficiently under varying market conditions [53]. This adaptability is facilitated through a feedback loop where the performance outcomes of the selected algorithms influence future selections, thus continually refining the selection process. Challenges in OAS include the computational overhead of the selection mechanism and the need for robust methods to evaluate and compare the performance of different algorithms under a wide range of conditions.

Gagliolo and Schmidhuber [54] present GambleTA, a framework for online algorithm portfolio selection. The focus of their work is on problems where solution time is the primary performance criterion, encompassing decision-making or search problems, decision versions of optimization problems, and combinatorial optimization. The algorithms considered include both complete algorithms, based on exhaustive search, and incomplete algorithms, based on local search. Users have flexibility in selecting the algorithm set and time allocators, including both model-based and non-model-based approaches.

Researches in OAS have demonstrated significant diversity across various domains, each employing distinct methodologies to enhance computational efficiency and adaptability. In the planning domain, a series of studies by de la Rosa *et al.* [36]leveraged instance features to predict suitable algorithms using case-based reasoning and decision trees. Similarly, Garbajosa *et al.* [55] applied a classifier ensemble to predict algorithms based on instance features.

In evolutionary algorithms, Yuen *et al.* [138] employed past performance metrics to predict algorithms using linear regression, showcasing the application of traditional statistical methods in dynamic settings. Constraint Programming has seen the use of search statistics for instance features for algorithm prediction via reinforcement learning [134], indicating a focus on real-time adaptability.

The domain of SAT and combinatorial problems, including the Quadratic Assignment Problem and Traveling Salesman Problem, also reflects similar trends. For instance, Wei *et al.* [129] and Veerapen *et al.* [128] respectively used search statistics and past performance to guide algorithm selection, employing hand-crafted rules and statistical models.

A pioneering study in 2017 [74] introduced innovative methodologies to dynamically select heuristics for improving the efficiency of tree search algorithms. This approach centers on adapting heuristic strategies based on historical performance data, enabling the algorithms to modify their problem-solving strategies in real time. By learning from past outcomes, the system can predict and deploy the most effective heuristic for a given scenario, thus optimizing the search process.

Kostovska *et al.* introduce a novel "per-run" algorithm selection method that utilizes trajectory-based features from an initial optimization phase to inform the selection of a potentially more effective algorithm for subsequent phases [77]. This approach significantly diverges from traditional static algorithm selection by incorporating warm-starting techniques, which leverage the accumulated data from the initial phase to im-

prove the performance of the subsequent algorithm.

The authors in [48] enhance algorithm selection by effectively merging per-instance and per-set strategies with machine learning. The selection process is dynamically adjuested based on real-time performance feedback. This dynamic strategy is supported by a predictive model that assesses the characteristics of each problem instance to forecast which algorithm will yield the best results. Additionally, Fan *et al.* present a portfolio-based approach, where multiple algorithms are evaluated and the most suitable one is selected for a given problem.

This variety in approach underscores the growing complexity of algorithm selection frameworks, which increasingly rely on a mixture of heuristic, machine learning, and statistical techniques to optimize performance across a broad spectrum of computational challenges.

In conclusion, OAS is exceptionally useful in environments where algorithm performance varies significantly across different conditions, as is common in machine learning and real-time bidding systems in online advertising. OAS adapts to these fluctuations by utilizing real-time performance assessments to inform the selection of the optimal algorithm for each unique situation.

4.5 MCDM and Algorithm Selection

The use of MCDM in algorithm selection provides a systematic methodology for comparing and ranking alternatives based on multiple evaluation criteria. This approach is particularly valuable in complex domains such as machine learning, operations research, and software engineering, where decision-makers often face diverse and conflicting criteria. MCDM helps balance factors such as computational efficiency, accuracy, and resource utilization, thereby facilitating a more informed and systematic choice among potential algorithms.

For instance, Peng *et al.* [107] develop a user preferences-based framework for selecting software defect detection algorithms utilizing four MCDM methods: Data Envelopment Analysis (DEA)[23], TOPSIS [68], ELECTRE[113], and PROMETHEE [19]. This approach facilitates the ranking of 38 classification algorithms across 13 performance measures using ten software defect datasets. The selection problem is effectively framed as an MCDM challenge, incorporating multiple evaluation criteria such as accuracy, computational time, and misclassification rate. This methodology enables stakeholders to systematically compare and rank algorithms, considering decision-makers' preferences and providing a tailored approach to enhancing software quality assurance processes.

Kou *et al.* [79] evaluate and rank classification algorithms across different criteria, which inherently involve conflicts due to their varying importance and nature. Their methodology employs five MCDM techniques: TOPSIS [68], ELECTRE III, VIKOR [101], PROMETHEE II, and Grey Relational Analysis (GRA)[37], which it is a reference-based model. They assess 17 classification algorithms over 11 public-domain binary classification datasets. The primary goal is to resolve discrepancies among the different MCDM rankings using Spearman's rank correlation coefficient, thereby providing a more

consistent ranking of classifiers.

Similarly, Kiran and Kitsuregawa [76] propose a method to select an appropriate data mining algorithm for modeling the compressive strength of high-performance concrete (HPC) using the AHP method [114]. Their approach evaluates various algorithms' performance on a dataset from the University of California, Irvine, and uses multiple performance metrics to determine the best algorithm. This method facilitates more accurate predictions in construction material research by enabling a structured comparison and selection process that incorporates a broad range of influencing factors, unlike traditional methods that typically rely on limited criteria like the water-to-cement ratio.

Barak and Mokfi [8] address the evaluation and selection of clustering algorithms through a hybrid group MCDM approach. The method incorporates three different MCDM algorithms—TOPSIS, COPRAS (COmplex PRoportional ASsessment) [59], and WSM (Weighted sum model) [51]—along with the Borda count method to provide a robust and comprehensive assessment. This study evaluates clustering algorithms across multiple datasets, utilizing both internal and external measures to ensure a thorough comparison.

Panwar and colleagues [105] introduce a qualitative framework designed for the selection of optimization algorithms suitable for addressing multi-objective trade-off problems in construction projects. This framework utilizes the Consistent Fuzzy Preference Relation (CFPR) method to rank various optimization algorithms based on predefined performance parameters informed by expert responses.

Bausys *et al.* [9] apply the Weighted Aggregated Sum Product Assessment (WASPAS) method for algorithm selection in the context of edge detection for satellite images. This method is a functional model based on aggregation operators. The authors address the challenge of selecting the most appropriate edge detection algorithm for satellite images, which varies based on the specific visual features of each image. The study integrates a variety of criteria, including texture and visual perception, to rank edge detection algorithms using the WASPAS method enhanced by neutrosophic logic, which manages uncertainty in the decision-making process effectively.

Wu *et al.* [130] introduce the Decision-Making Support for the Evaluation of Clustering Algorithms (DMSECA), a framework designed to aid in the evaluation and selection of clustering algorithms. DMSECA addresses the complexity of algorithm performance assessment and selection by integrating expert opinions to resolve conflicts among various performance metrics. The framework employs four MCDM approaches—WSM, GRA, TOPSIS, and PROMETHEE II—to rank clustering algorithms.

Das *et al.* [31] investigate the selection of portfolio optimization strategies using MCDM, an indirect application of algorithm selection. They provide an analysis of different optimization algorithms for portfolio management over a specified period. The research compares some optimization algorithms using multiple performance metrics like Sharpe ratio, expected return, and volatility. It integrates MCDM methods, specifically Logarithmic Percentage Change-driven Objective Weighting (LOPCOW) [44] and Compromise Ranking of Alternatives from Distance to Ideal Solution (CRADIS) [22], to evaluate and rank these algorithms.

Zhang *et al.* [141] introduce a Multi-Criteria Group Decision-Making method (MCGDM) tailored for evaluating production scheduling algorithms in complex industrial contexts. The method incorporates both Group AHP and a cloud-model-enhanced TOPSIS, facilitating a detailed assessment of scheduling strategies. This approach blends the deterministic modeling strength of AHP with the cloud model's ability to handle uncertainty, enabling more adaptive and robust decision-making. By considering a comprehensive set of factors—including economic, environmental, and social criteria—the method aims to optimize decision-making processes in dynamic and multifaceted production environments.

The table 4.1 provides a comparative overview of the key studies we have presented before. It details the types of algorithms assessed, the MCDM methods applied, and the specific criteria used in each study. This table demonstrates the application of diverse MCDM techniques — DEA, TOPSIS, ELECTRE, PROMETHEE, Consistent Fuzzy Preference Relations (CFPR), and the neutrosophic WASPAS method—to manage the intricate decision-making challenges in fields ranging from software defect detection to multi-objetive optimization and production scheduling.

The studies clearly demonstrate that Multi-Criteria Decision Making (MCDM) holds significant promise for addressing the challenges inherent in the AS problem. Different MCDM approaches, ranging from relational and reference-based methods to functional techniques, were used in the Algorithm Selection process. This approach is particularly valuable in environments characterized by multiple conflicting criteria, such as computational efficiency, solution quality, and resource utilization. MCDM thus becomes an indispensable tool for navigating the trade-offs and conflicts inherent in these selection processes.

4.6 Algorithm Selection of MOO Algorithms

4.6.1 Misir's Tools

Misir's suite of tools provides solutions for optimizing algorithm selection and tuning across various complex problem domains. These tools enhance the efficiency and adaptability of solving complex optimization problems through approaches like collaborative filtering, real-time adaptive models, and performance-based recommendations. Designed to handle both dense and sparse data, they significantly improve predictions and decision-making processes in various challenging scenarios, from parameter tuning in multi-objective optimization to managing large-scale problem instances with conflicting objectives.

OSCAR

OSCAR [96] is a framework that combines algorithm portfolios with OAS to optimize memetic algorithms (MAs). It begins by extracting features from problem instances, such as size, density, and domain-specific characteristics, to characterize their properties. It continuously monitors algorithm performance across various metrics, including

Paper	**Algorithms**	**Methods**	**Criteria**
Peng *et al.* [107]	Software defect detection	DEA TOPSIS ELECTRE PROMETHEE	User preferences, software quality
Kou *et al.* [79]	Classification	TOPSIS ELECTRE III GRA VIKOR PROMETHEE II Rank correlation	Classification effectiveness, dataset variability
Kiran *et al.* Kitsuregawa [76]	Data mining	AHP	Performance metrics
Barak *et al.* [8]	Clustering	TOPSIS COPRAS WSM Borda count	Internal and external measures
Panwar *et al.* [105]	MOO	CFPR method	Optimization algorithm suitability
Bausys *et al.* [9]	Edge detection	Neutrosophic WASPAS method	Edge detection algorithm performance
Wu *et al.* [130]	Clustering	WSM GRA TOPSIS PROMETHEE II	Algorithm performance metrics
Das *et al.* [31]	Portfolio optimization	LOPCOW CRADIS	Optimization algorithm performance
Zhang *et al.* [141]	Production scheduling	Group AHP Cloud-model-enhanced TOPSIS	Economic, environmental, social criteria

Table 4.1: Comparison of MCDM in AS Papers

solution quality, computational time, and convergence behavior. At its core, OSCAR utilizes an adaptive learning model that leverages the collected data to predict which algorithms will perform best under specific conditions, employing techniques like machine learning or statistical analysis. This model informs a dynamic portfolio management system, allowing OSCAR to select and configure algorithms in real-time based on predicted effectiveness for both current and future problem instances. A critical component of OSCAR's methodology is its feedback loop, where performance results are fed back into the model, refining predictions and selections, enabling continuous improvement and adaptability to new data or changing environments.

ALORS

Misir *el al.* [97] introduce a collaborative filtering approach to algorithm selection, which contrasts with traditional methods that rely on building and learning performance models.

ALORS operates by handling both dense and sparse data scenarios, which allows it to work efficiently without needing a complete performance dataset for all algorithm and problem instance pairs. The recommender system employs a set of steps:

ALORS begins by collecting historical performance data on various algorithms across a range of problem instances, creating a dataset of algorithm-instance performance metrics. The system then applies collaborative filtering techniques, specifically matrix factorization, to decompose this data into lower-dimensional latent factor matrices, which reveal underlying patterns in algorithm performance relative to instance features. To address the sparsity of the data matrix—where not all algorithms are tested on all instances—ALORS employs methods to infer missing data points, enhancing the robustness of its predictions. The system models the latent factors of both algorithms and problem instances non-linearly, capturing complex interactions between them. Using these latent factors, ALORS predicts the performance of each algorithm for a given instance, ranking them to recommend the best-performing algorithm.

ALORS-MOO

The paper [98] presents optimizing algorithm selection for parameter tuning methods. The study's core premise is to enhance the performance of tuning tools for MOO problems by selecting the most suitable tuning method for each instance. It outlines a systematic approach using ALORS [97], which predict performance and recommend the best tuning methods from a set of well-known parameter tuners, such as Random Search, Bayesian Optimization, and SMAC.

The methodology includes generating a performance matrix by applying these tuners to a suite of problem instances from the CEC'2020 benchmark suite. Misir uses ALORS to analyze and map the feature space derived from performance data to a new, latent feature space that better represents the instances and tuners.

ALORS-LSMOP

ALORS-LSMOP [99] explores the application of AS to effectively determine the most suitable algorithms for addressing large-scale multi-objective optimization problems. This study focuses on enhancing algorithm performance by utilizing AS to handle complex optimization tasks that involve conflicting objectives, such as those found in real-world scenarios. The methodology utilized involves a case study with four multi-objective optimization algorithms tested across 63 benchmarks involving problems with two to three objectives and varying numbers of variables (46 to 1006). The study employs the Hypervolume indicator to evaluate algorithm performance and uses a basic set of four instance features to facilitate AS.

4.6.2 AsLib

ASlib [16] is a benchmark library and platform aimed at enhancing the comparability and exchange of algorithm selection (AS) techniques across various research groups. This initiative tackles the challenge of diverse data formats and the absence of a unified repository, which previously impeded the algorithm selection community's ability to perform effective comparative studies.

ASlib[1] provides a standardized format for representing AS scenarios and includes a comprehensive dataset from various domains, including SAT, AI planning, and machine learning. It supports evaluations through a common interface and is designed to be extensible.

Features

ASlib systematically catalogs a broad spectrum of features to describe problem instances. These features encompass instance size, complexity measures, and domain-specific characteristics, such as the clause-to-variable ratio in SAT problems. By standardizing these features, ASlib facilitates their integration into selection models by researchers.

Criteria

The primary evaluation criteria in ASlib include performance metrics such as runtime, solution quality, and the cost associated with deploying particular algorithms. These criteria are essential for assessing the effectiveness of algorithm selection methods in identifying the most appropriate algorithm for given problem instances, based on predefined performance benchmarks.

4.6.3 De Jesus's Frameworks

The methodology proposed by Alexandre de Jesus [70] aims to address AS for MOO problems, particularly focusing on scenarios with uncertain time budgets and desired

[1] https://www.coseal.net/aslib

solution quality. The approach employs predictive modeling techniques, such as regression and classification, to forecast the performance of MOO algorithms over time.

Offline AS Framework: This framework utilizes empirical models to predict the anytime performance of MOO algorithms on new instances. It incorporates a utility function that reflects both time and quality preferences to select the most suitable algorithm before commencing the problem-solving process. The key steps include predicting anytime performance, evaluating utility functions to measure trade-offs, and selecting the algorithm that best aligns with the decision maker's time and quality preferences.

OAS Framework: This framework dynamically selects and switches algorithms during problem-solving based on ongoing performance indicators. It continuously monitors performance in real-time, compares it against predefined criteria or decision maker preferences, decides to switch algorithms if another could offer better performance under current conditions, and seamlessly implements the switch to continue the problem-solving process with the new algorithm.

Both frameworks integrate decision maker preferences and rely heavily on data-driven approaches, utilizing historical data for offline predictions and real-time data for online adjustments. This ensures that the selected algorithms align with specific problem constraints and goals.

The methodology for predicting the anytime performance of algorithms on previously unseen instances of the Multi-Objective Binary Knapsack Problem (MOBKP) involves several key steps. First, features characterizing each problem instance in the training dataset are identified and normalized to a [0, 1] scale. Then, a k-nearest neighbor algorithm is used to compute distances between training data instances and unseen instances, with feature weights determined by their correlation with anytime performance. The k closest instances are selected, and their anytime performance traces are aggregated to construct a performance profile for the unseen instance. Finally, predictive accuracy is validated by splitting known instances into training and testing sets, employing numerical and visual analysis, and optimizing the k value using Leave-One-Out Cross-Validation (LOOCV) to minimize Mean Absolute Error (MAE) across all testing instances.

The experiments were conducted using instances of MOBKP with varying objectives (2, 3, and 5 objectives). This problem involves selecting items with given weights and benefits (extending to multiple objectives), aiming to maximize the benefit while staying within a weight limit. This methodology emphasizes a data-driven approach to algorithm selection, leveraging historical performance data and instance characteristics to predict how algorithms will perform on new, unseen instances.

4.6.4 Tian *et al.* Selector

Tian *et al.* [123] develop an automated method to select the most effective evolutionary algorithm for specific MOO Problems (MOPs). Their approach involves constructing a predictive model that utilizes both explicit and implicit features of MOPs to forecast which algorithm will perform best. Explicit features include measurable criteria like the number of objectives and decision variables, while implicit features relate to the problem's landscape and Pareto front characteristics. The model is trained using data from

the performance of various evolutionary algorithms across benchmark MOPs, employing machine learning techniques such as support vector machines.

The selector begins with data collection, where a set of ten candidate evolutionary algorithms, including AR-MOEA, BCE-IBEA, GFM-MOEA, MOEA/D, MOEA/DD, NMPSO, NSGA-III, NSGA-II/SDR, RVEA, and SPEA2+SDE, is empirically evaluated across multiple benchmark MOPs. The performance of these algorithms is recorded, forming a dataset that reflects their effectiveness across different problem types. Feature extraction follows, where explicit features such as the number of objectives and decision variables, and implicit features related to the problem's landscape and Pareto front, are considered. A predictor model is then developed using techniques like Support Vector Machines and Regression models, with the extracted features as inputs and the identification of the best-performing algorithm as the output. This model is trained on the initial dataset, and its parameters are optimized to enhance prediction accuracy. Finally, the model is validated and tested on new MOPs, including those from the WFG (Walking Fish Group) toolkit, to assess its applicability in predicting the optimal evolutionary algorithm for new problems.

4.6.5 Autofolio

AutoFolio [86] is presented in the Section 4.3 as an automated AS tool.

In optimization problems, the tool facilitates determining the most effective optimization algorithm for specific instances of optimization problems, ensuring that the solution quality and computational resources are optimally balanced.

The portfolio of AutoFolio is not a static list of algorithms but rather a dynamic and extensive framework of AS strategies and machine learning models that can be tailored to any given AS scenario.

The methodology of AutoFolio involves four key steps: First, **Configuration of Algorithm Selectors** involves using algorithm configurators to tailor AS frameworks for specific scenarios, defining a parameterized selector, its configuration space, and a performance metric for evaluation. Second, **Formal Problem Statement** defines an AS scenario including algorithms, problem instances, and relevant performance and feature data, with the selector's effectiveness measured by the average performance of selected algorithms on test data. Third, **Cross-Validation and Optimization** uses cross-validation to optimize the selector's parameters by splitting the training data into multiple folds, ensuring the best configuration is identified without accessing the test set. Finally, **Final Evaluation** involves training the selected configuration on the entire training data and evaluating its performance on a withheld test set to ensure an unbiased assessment.

4.6.6 Liefooghe

Liefooghe et al. [85] explore the application of landscape features, originally designed for multi-objective combinatorial optimization, to multi-objective interpolated continuous optimization problems (MO-ICOPs). They introduce a methodology to assess these

features' predictive power in algorithm selection and performance prediction across 1200 randomly-generated bi-objective problems. Their method relies on fixed-size sampling of the search space, which allows for an efficient evaluation of landscape features while controlling computational costs. The authors find that the landscape features, when used with classification models like random forests, provide comparable predictive accuracy to features derived from problem parameters. This suggests that these landscape features are effective in aiding the selection of the most suitable algorithm for given problem instances.

4.6.7 Comparative study

In the comparative analysis of AS tools for MOO (cf. table 4.2), several factors are essential for comprehensive evaluation. The **methodology** section examines the processes and methods utilized by each tool, such as optimization techniques, machine learning methods, or heuristic algorithms. This examination aids in understanding the operational mechanisms of each tool and the reasons behind its efficacy for particular applications. The **portfolio** highlights the diversity of algorithms or strategies that the tool is capable of managing. The **benchmark** criterion provides details on the datasets or standard tests used in the selection process, to evaluate the tool, offering insights into the operational contexts and performance metrics. **Measures** for evaluation include metrics such as accuracy, efficiency, and hypervolume, essential for gauging the effectiveness of each algorithm or strategy and for their subsequent comparison and ranking. Finally, the **features** section describes specific characteristics that differentiate the algorithms or strategies being compared, such as the number of variables or the modality of the problem instances. Collectively, these components furnish a framework to compare the tools based on detailed operational and functional aspects, thereby facilitating the derivation of guidelines for the development of our tool and positioning it within the broader context.

Table 4.2: Comparison of Various Algorithm Selection Tools

Tool	**Methodology**	**Portfolio**	**Benchmark**	**Measures**	**Features**
OSCAR [96]	Combines algorithm portfolios with OAS.	Memetic algorithms	Quadratic assignment and flowshop scheduling problems	Number of found Best known Solutions	Number of new best solutions, improving solutions, and worsening solutions. Number of moves. Amount of improvement. Amount of worsening. Total spent time.
ALORS [97]	Collaborative filtering approach using matrix factorization to handle sparse data and predict algorithm performance.	SAT, CSP and ML portfolios	ASlib and OpenML.	Latent factors predicting performance Rank. Regret. Penalized average runtime (PAR10). Solved Instances Ratio	65 initial and descriptive features. Latent features
ALORS-MOO [98]	Uses ALORS for optimizing AS for parameter tuning methods in MOO.	Parameter tuners(ParamILS, SMAC, ...) for NSGA-II	NSGA-II	Improved parameter tuning performance	Scalable number of variables and objectives, pareto front geometry, the MOO metrics (Hypervolume, Generational Distance, additive $\epsilon-$indicator)
ALORS-LSMOP [99]	Applies AS to large-scale multi-objective optimization problems using collaborative filtering techniques.	SMPSO, MOEA/DVA, LMEA, WOF-SMPSO	63 LSMOP benchmarks	Hypervolume	Number of Objectives, Number of Variables, Modality, Separability.

Tool	Methodology	Portfolio	Benchmark	Measures	Features
De Jesus [70]	Predictive modeling for AS in MOO with uncertain conditions.	MOO algorithms like PLS and BHV-DP, DFS, IBB.	Multi-Objective Binary Knapsack Problem (MOBKP) with varying objectives	Performance prediction accuracy Utility function effectiveness. Anytime performance. Decision maker's preferences.	Instance features (such as size, complexity, and specific structural properties)
MOSAP [62]	Manages trade-offs in algorithm portfolios for optimization tasks.	Optimal subset contributing to a joint Pareto front	Various real-world benchmark problems for experimental validation.	Efficiency. Accuracy. Resource consumption.	Problem instance characteristics that influence algorithm performance.
Tian et al. [123]	Empirical evaluation of a set of candidate MOEAs across multiple benchmark MOPs.	MOEAs like AR-MOEA, BCE-IBEA, GFM-MOEA, MOEA/D.	WFG4-WFG9 problems from the Walking Fish Group toolkit.	Prediction accuracy Pareto efficiency. Explicit and implicit features of MOPs. Benchmark validation.	Number of objectives and decision variables. Characteristics of the problem's landscape and Pareto front.
AutoFolio [86]	Automated configuration of algorithm selectors to optimize performance across different scenarios.	Dynamic and extensive framework of algorithm selection strategies	Algorithm Selection Library (AsLib) scenarios	Cross-validation performance. Penalized average runtime (PAR10).	Instance features (number of variables in a SAT instance for example)
Liefooghe [85]	Uses landscape features for MO-ICOPs, assessing predictive power through fixed-size sampling and classification models.	MOEAs: NSGA-II, GDE3, MOEA/D-DE-DRA, DECMO2++	1200 bi-objective problems (MO-ICOPs)	Predictive accuracy of features. Hypervolume	49 landscape features from combinatorial MOO applied to continuous problems.

The table 4.2 offers insights into the capabilities and suitability of the tools for different optimization tasks.

Methodologies

Each tool employs methodologies tailored to specific types of optimization problems. For instance, Liefooghe's tool uses landscape features for assessing predictive power through fixed-size sampling and classification models, suitable for continuous problem spaces. OSCAR, on the other hand, focuses on combining algorithm portfolios with OAS to optimize memetic algorithms, indicating a dynamic approach to problem-solving in environments such as scheduling and assignment problems.

Portfolios

The portfolio column reflects the diversity and adaptability of each tool. ALORS, for example, demonstrates versatility by supporting a wide range of portfolios including SAT, CSP, and machine learning problems. This contrasts with ALORS-MOO, which specifically optimizes for parameter tuning methods in MOO, showing a more focused approach.

Benchmarks

Benchmarks provide a context for where these tools have been tested and proven effective. ALORS-LSMOP is evaluated against 63 large-scale multi-objective optimization problem benchmarks, indicating its robustness in handling complex and scalable optimization tasks. Similarly, Tian et al.'s tool are tested on WFG4-WFG9 problems from the Walking Fish Group toolkit, a standard set for evaluating evolutionary algorithms.

Measures

Evaluation Measures such as predictive accuracy, efficiency, and resource consumption are crucial for assessing the effectiveness of these tools. MOSAP emphasizes efficiency, accuracy, and resource consumption, providing a clear metric for performance in real-world optimization tasks. OSCAR focuses on adaptability and execution time, key for environments requiring rapid responses and adaptive strategies.

Features

The features column highlights specific capabilities that differentiate each tool. De Jesus' framework focuses on predictive modeling for algorithm selection with uncertain conditions, valuable in scenarios where data or parameters may be incomplete or changing. AutoFolio offers automated configuration of algorithm selectors to optimize performance across different scenarios, showcasing advanced customization and flexibility.

Collectively, these tools demonstrate a broad spectrum of approaches and capabilities, from general-purpose tools with wide applicability (like ALORS) to highly specialized tools designed for specific optimization scenarios (like ALORS-MOO). This diversity underscores the richness of the field of algorithm selection in MOO, where different tools can be selected based on the complexity of the problem, the nature of the optimization tasks, and the specific performance metrics crucial to the user's needs.

4.7 Conclusion

Algorithm Selection is a diverse and evolving field that encompasses a variety of methodologies tailored to specific optimization needs and problem scenarios. Offline selection strategies, such as SATzilla and AutoFolio, utilize pre-configured datasets to develop static decision models that guide algorithm choices based on robust historical data. These methods are highly effective for well-understood problem types but often struggle to adapt to unforeseen changes during runtime. In contrast, online selection techniques, including dynamic methods like GambleTA and OSCAR, leverage real-time data to adjust algorithm choices based on the immediate characteristics of problem instances. This adaptability is particularly valuable in environments where problem parameters evolve or where real-time decision-making is crucial, such as in online advertising or dynamic resource allocation tasks.

The integration of MCDM methods into AS allows for the systematic evaluation and ranking of algorithms based on multiple conflicting criteria like computational efficiency, accuracy, and resource utilization. This is especially important in domains where balancing multiple objectives is necessary. In MOO, AS becomes even more complex, requiring tools that can accurately predict and select the best algorithm based on intricate problem characteristics. The development of Algorithm Selection Libraries (ASlib) and tools like ALORS and OSCAR underscores a growing trend towards collaborative, data-driven, and dynamically adjustable frameworks.

Our objective is to develop a decision support system specifically designed for selecting algorithms tailored for Evolutionary Circuit Design, a problem characterized by its large-scale nature. Currently, the only available tool for algorithm selection in large-scale problems is ALORS-LSMOP, which uses criteria like the number of objectives, number of variables, modality, and separability. However, this tool lacks the incorporation of other essential metrics such as convergence, diversity, and evolvability. Additionally, tools like ALORS-MOO, which use a comprehensive set of criteria from both algorithms and problem features, are more focused on tuning algorithms (only NSGA-II) rather than selecting them.

Most existing tools in this domain prioritize factors like the number of variables and objectives along with some basic problem characteristics. Some research efforts have concentrated on algorithm selection for inclusion in a portfolio, while others have examined the impact of landscape features on the performance of Multi-Objective Evolutionary Algorithms (MOEAs).

Our goal is to design a decision support that selects algorithms from a portfolio of

MOEAs based on a comprehensive set of criteria, encompassing both the performance metrics of the algorithms themselves and the specific characteristics of the problem. This approach is intended to provide a holistic understanding of how algorithms will perform on a particular problem, enabling more informed and effective algorithm selection for complex, large-scale problems like Evolutionary Circuit Design.

Finally, as algorithms continue to permeate various domains of science and engineering, the strategies for their selection will remain crucial in harnessing their full potential, ensuring that the chosen algorithms are not only fit for purpose but also optimal in performance across diverse scenarios.

Chapter 5

Methodology and Development

5.1 Introduction

In algorithm selection for optimization problems, particularly in multi-objective optimization, several factors are important to consider. While performance metrics are essential, focusing solely on them may not always lead to the best algorithm selection.

Moreover, understanding the problem's characteristics through analysing its landscape features is equally important. Problem landscape features provide insights into the optimization problem's structure, such as ruggedness, multi-modality, and evolvability, which can significantly impact the performance of different algorithms. By considering problem landscape features, one can make a more informed selection of algorithms that are well-suited to the specific problem landscape.

To address this need for an approach in algorithm selection for multi-objective algorithms, we developed a system using the Logic Scoring of Preferences (LSP) method as a MCDM system to perform an online selection of the algorithms. This tool, named AS-LSP-MOEA, stands for Algorithm Selection using LSP for MOEAs. The LSP method distinguishes itself among multi-criteria evaluation methods through its hierarchical logic aggregation structure, which offers considerable flexibility by employing graded logic to represent simultaneity and replaceability. This capability allows LSP to effectively model complex relationships between criteria. Furthermore, LSP excels in handling a large number of inputs without diminishing the significance of any individual criterion, owing to the sophisticated logic expressions it employs. These features make LSP more effective than other methods, such as additive aggregation operators like weighted sum or weighted average. and Analytic Hierarchy Process (AHP), particularly in scenarios requiring a detailed and flexible approach to decision-making, as highlighted in various comparative studies [61].

The development of AS-LSP-MOEA follows a systematic three-step process:

- First, we construct a portfolio of multi-objective optimization algorithms that are potentially suitable for the problem at hand. This portfolio comprises a diverse set of Multi-Objective Evolutionary Algorithms (MOEAs) (see §5.7 for more details

on the MOEAs used). The algorithms are tested against some of well-known benchmarks, such as ZDT [143], CF [140], LSMOP [26], and others. This strategy ensures that the algorithms are evaluated using established benchmarks, providing a robust basis for comparison.

- Next, we build a hierarchy of criteria and define the suitability functions that will be used in elementary criteria to analyse the performance of the algorithms within the portfolio. Then, the most convenient aggregation operation is selected for each intermediate node, in order to model the different situations of simulatneity/replaceability among criteria. Weights are also given to the different elements. Finally, a software tool supporting this structure is implemented.

- Finally, we analyze the results generated by AS-LSP-MOEA on well-known problem benchmarks, taking into account the stochastic nature of multi-objective optimization algorithms and the inherent complexity of managing conflicting objectives. This analysis enables us to fine-tune the tool, ensuring it aligns more closely with the specific requirements of these benchmarks before applying it to a case study in Evolutionary Circuit Design.

5.2 Architecture of AS-LSP-MOEA

The architecture of AS-LSP-MOEA is organized into several interconnected stages, each designed to leverage problem-specific characteristics and empirical performance data. The overall architecture is depicted in Figure 5.1, which illustrates the sequential flow of operations from problem definition to the final ranking of algorithms.

5.2.1 Initialize AS-LSP-MOEA

The first stage of AS-LSP-MOEA involves initialization, which includes the following steps:

- **Selection of Alternatives:** Selecting the alternatives from the portfolio of multi-objective optimization algorithms. This portfolio consists of various MOEAs, each with different characteristics.

- **Definition of the Criteria Tree:** Defining the criteria tree, which includes specifying the hierarchy of the criterion, utility values, weights and aggregators.

- **Setting Operational Parameters:** Establishing the operational parameters, such as the maximum allowed time for algorithm execution or the maximum number of evaluations. These parameters ensure that the algorithm execution is aligned with practical limits of the task.

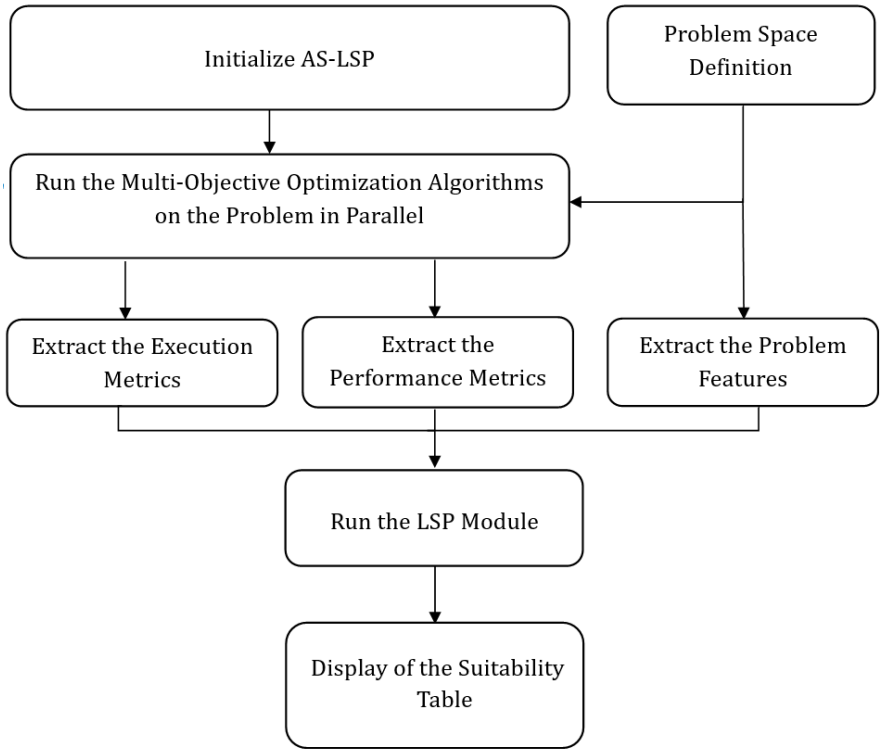


Figure 5.1: Architecture of AS-LSP-MOEA

5.2.2 Problem Space Definition

Parallel to the selection of algorithms, the problem space is defined. This involves identifying the problem, as well as the number of objectives, constraints, and decision variables.

5.2.3 Run the Multi-Objective Optimization Algorithms on the Problem in Parallel

The pre-selected algorithms are then executed in parallel on the defined problem. Running these algorithms concurrently maximizes the use of computational resources and ensures that the allowed maximum time or number of evaluations is not exceeded, even though the algorithms are run in parallel. This parallel execution strategy allows for the execution of the algorithms without extending the overall computational limits set by the decision-maker.

5.2.4 Extract the Execution and Performance Metrics

During the execution of the algorithms, the execution metrics are collected. After the execution is complete, performance metrics for all the alternatives are also extracted.

These metrics are then stored in files as criteria tables structured according to the criteria tree defined in AS-LSP-MOEA. See section §5.3 for more details on the metrics used.

5.2.5 Extract the Problem Features

In this stage, the features of the problem landscape are extracted. These features are added to the criteria table respecting the defined criteria tree. See the section §5.3 for more details on the problem landscape feature.

5.2.6 Run the LSP Module

The collected execution metrics, performance metrics, and problem features are then input into the LSP module. Using the criteria tree and the decision-maker's preferences, the LSP aggregation is performed to produce rankings and overall scores for the pre-selected algorithms on the benchmarks. Different weights and preferences are applied to compare the results across benchmarks, allowing for an evaluation of algorithm quality.

5.2.7 Display of the Suitability Table and Algorithms Ranking

The final step in the AS-LSP-MOEA methodology involves the presentation of the results, where the system provides:

- **Suitability Table:** A detailed table that displays how each algorithm scored on different criteria. This table allows users to see where each algorithm excels or falls short.
- **Evaluation Results:** A table presenting a clear ranking of the algorithms, displaying their aggregated scores alongside the individual scores for each criterion in the criteria tree.

5.3 Criteria

The criteria used in AS-LSP-MOEA are categorized into three main groups: **Execution Metrics**, **Indicators**, and **Landscape Features**. Each group includes specific metrics and indicators to assess different aspects of the algorithm's performance.

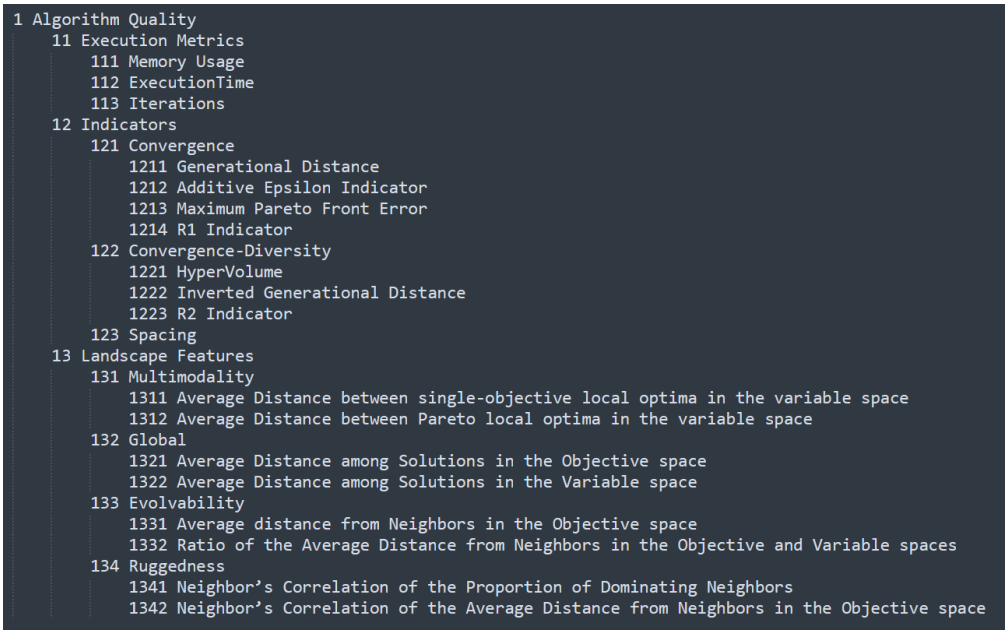


Figure 5.2: Criteria Tree in AS-LSP-MOEA

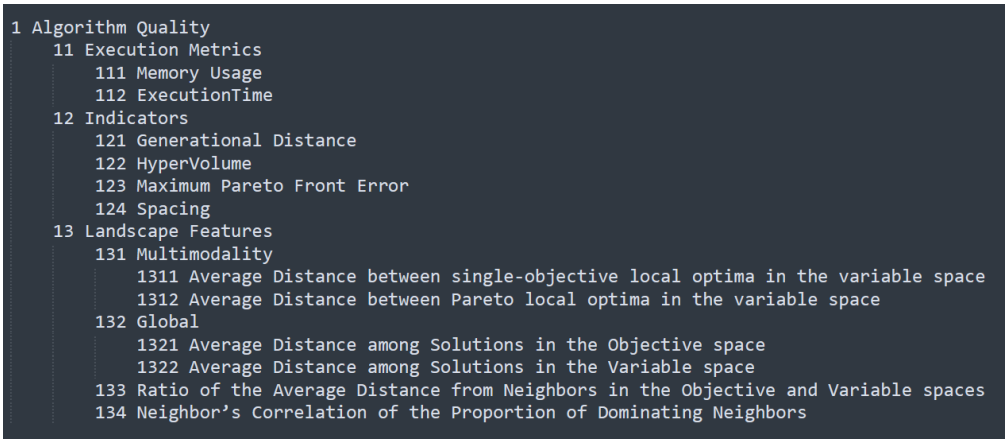


Figure 5.3: Criteria Tree in AS-LSP-MOEA (Non Correlated Criteria)

5.3.1 Execution Metrics

These metrics evaluate the algorithm's operational efficiency and resource consumption. They include:

- **Execution Time**: The total time taken by the algorithm to complete the optimization process within the allocated maximum time.

- **Iterations**: The number of iterations the algorithm completes within the allocated maximum time.
- **Memory Usage**: The amount of memory consumed during the execution of the algorithm.

5.3.2 Performance Indicators

These indicators are context-dependent, meaning their values depend on the specific problem, the reference set, the reference point, and the solutions obtained by the algorithm. To enable comparison between the algorithms, some indicators are normalized. They can be used for both continuous and discontinuous approximations of Pareto sets [7].

We use performance indicators from three aspects: **Convergence** and **Convergence-Diversity**, along with **Diversity**.

Convergence

Indicators that measure how well the algorithm converges to the optimal solutions. These include:

- **Generational Distance (GD) [127]**: It measures the average distance between the solutions obtained S and the true Pareto front P. A lower GD indicates better convergence, with $GD = 0$ representing perfect convergence, as all solutions are exactly on the true Pareto front.

  GD favors algorithms that return a few non-dominated points near the Pareto front over those with a more distributed representation. This can unfairly judge algorithms offering diverse solutions as inferior [7]. It is important to consider additional metrics that evaluate the distribution and coverage of solutions to obtain a more accurate assessment. Therefore, we use a set of complementary metrics in AS-LSP-MOEA to evaluate the algorithms.

- **Additive ϵ-Indicator ($I_{\epsilon+}$)**: It measures the minimum value by which the solution set S needs to be translated to dominate Pareto front. It provides an intuitive understanding of how much the solutions in S need to be adjusted to fully cover the Pareto front represented by the reference set R. A smaller $I_{\epsilon+}$ value indicates that S is closer to dominating R, implying better performance.

  The $I_{\epsilon+}$ in the AS-LSP-MOEA allows for a balanced evaluation of all objectives uniformly, ensuring that no single objective disproportionately influences the assessment. This is particularly useful in contexts where trade-offs between conflicting objectives must be fairly evaluated.

- **Maximum Pareto Front Error ($MPFE$) [127]**: It measures the greatest minimum distance from any vector in the solutions obtained S to the nearest vector in the true Pareto front (P). This metric helps assess how closely S matches P.

If $MPFE = 0$, it means that every vector in S is exactly on P, indicating perfect conformity. Conversely, if $MPFE > 0$, it means that at least one vector in S is not on P.

Using the $MPFE$ as a metric in AS-LSP-MOEA helps identify outliers and ensures that the algorithm performs well even in the worst-case scenarios, highlighting poor solutions. Additionally, $MPFE$ complements other metrics like HyperVolume (HV) and Generational Distance (GD), offering an integrated view of an algorithm's performance.

- **R1 Indicator [60]**: The R1 indicator calculates the probability that approximation A is better than B over an entire set of utility functions. It is a direct comparative indicator that does not induce a total ordering on the set of approximations and is a non-cardinal measure.

  The best values depend on the weighted Chebyshev utility functions[1] reflecting the decision-maker's preferences. The values of this metric range from $[0, 1]$ with 1 preferred.

  The $I_{\epsilon+}$ and the **MPFE** serve different purposes and offer distinct perspectives on algorithm performance. The $I_{\epsilon+}$ focuses on the dominance relationship by measuring the minimum uniform translation required for the solution set to dominate the reference set, making it useful for assessing overall effectiveness across all objectives. In contrast, the MPFE highlights the maximum deviation of any point in the solution set from the true Pareto front, providing a clear measure of the worst-case error and the precision of the solutions. Using both metrics in AS-LSP-MOEA together can offer a well-rounded assessment, enabling more informed and balanced decisions in algorithm selection and comparison.

Convergence-Diversity

Indicators that evaluate both the convergence and the diversity of the solutions:

- **HyperVolume (HV) [143]**: Also known as the $\mathcal{S}$ measure, the HyperVolume metric measures the volume of the dominated region in the objective space dominated by a solution set S relative to a reference point, providing an evaluation of solution quality across all objectives.

  The HV evaluates both the convergence to the optimal set and the spread across the objective space, making it a preferred metric. While it has advantageous mathematical properties and is maximized only with Pareto-optimal solutions, it is sensitive to the choice of reference point and computationally intensive to calculate, though fast approximation algorithms exist.

[1] The Chebyshev utility function is a scalarizing function to convert multiple objectives into a single aggregated objective by considering the weighted maximum deviation from the ideal point

A higher HV value indicates that the solution set covers a larger portion of the objective space, suggesting that the solutions are closer to the Pareto front and more optimal, thereby reflecting better performance.

Using the HyperVolume indicator in AS-LSP-MOEA is beneficial due to its ability to uniformly evaluate all objectives and its sensitivity to the Pareto front shape.

- **Inverted Generational Distance (IGD) [24]**: It measures the extent of convergence by evaluating how close the obtained solutions S are to the true Pareto front P. It also considers the distribution of solutions along the Pareto front, as it involves all points on the front.

  For the IGD metric to provide accurate results, the number of solutions in S should be sufficiently large. IGD can be applied in both solution space and objective space. When using the Euclidean distance, IGD evaluates convergence-diversity of the algorithms. Conversely, when using the Manhattan distance, IGD measures only the convergence and is also known as the convergence metric γ.

  An IGD of 0 means that the Pareto front is perfectly covered by the obtained solutions, indicating ideal convergence and diversity.

  Using IGD in AS-LSP-MOEA helps in identifying algorithms that offer a more complete and diverse representation of the Pareto front. Its ability to penalize solutions that miss parts of the Pareto front ensures a more accurate evaluation of algorithm performance.

- **$R2$ Indicator [60]**: It calculates the expected utility difference between approximations A and B. The $R2$ Indicator is computationally efficient, and unlike the HV indicator, it uses an ideal or utopian reference point[2] instead of an anti-optimal[3] one, making it suitable for applications like error or cost minimization. Although the $R2$ indicator is only weakly Pareto-compliant[4], it generally performs well in practice, especially in continuous unconstrained optimization problems [47].

  The $R2$ values are normalized between $[-1, 1]$ with -1 preferred.

Diversity

- **Spacing (Sp) [117]**: Spacing is intended to assess the uniformity of distribution among the members of an approximation set. An Sp value of 0 indicates that all members are evenly spaced from one another.

  The main limitation of the Spacing indicator is that it provides limited insight when the solutions generated by the algorithm are distinctly separated but clustered into multiple groups. In such cases, the indicator may not accurately reflect

[2] An ideal or utopian reference point is a hypothetical point in the objective space where all objective functions reach their optimal values.

[3] An anti-optimal reference point is a point in the objective space where all objective functions reach their worst values.

[4] Weakly Pareto-compliant means that while a better approximation should result in a better indicator value, this is not guaranteed for all cases.

the overall distribution of the solutions, as it primarily measures the evenness of spacing among nearby points. Despite this drawback, the Spacing indicator is simple and easy to compute, making it a convenient tool for basic assessments of solution distribution [7].

5.3.3 Landscape Features

We utilize the research conducted by Liefooghe *et al.* [85] to identify the most relevant landscape features for predicting the performance of multi-objective algorithms. This study selected a subset of the most significant features from among 49 examined, based on their correlation with algorithm performance. By analyzing these correlations, the study pinpointed which features are most indicative of algorithmic success.

This enabled us to choose the following landscape features for AS-LSP-MOEA, considering those that are less resource-intensive. The features are divided into four subcategories: **multi-modality**, **Global**, **Evolvability**, and **Ruggedness**. Each category encompasses specific features that provide different insights into the landscape's properties.

Multi-modality Features

multi-modality refers to the presence of multiple local optima within the search space, indicating several Pareto-optimal fronts with different trade-off solutions between objectives. This complexity requires algorithms that can maintain diversity within the population to effectively explore various regions of the Pareto front and avoid getting trapped in local optima. Techniques such as niching or crowding are often employed to manage multi-modality.

- **Average Distance between Single-objective Local Optima in the variable space (Slo_DistAvg)** This is the average Euclidean distance between local optima considering one objective at a time. This metric is used to characterize the landscape of optimization problems by quantifying how far apart the local optima are from each other. A **larger average distance** suggests that the local optima are more spread out, which can require significant exploration across the variable space to avoid getting stuck in widely spaced local optima, as they might need to search across larger regions of to find the global optimum. Conversely, **a smaller average distance** indicates that the local optima are clustered more closely together, which might make the landscape easier to explore for algorithms designed to avoid getting trapped in local optima.

  The "best" value for the "Average Distance between Single-objective Local Optima in the variable space" depends on the specific goals and context of the optimization problem. A **small average distance** is beneficial for encouraging exploration and making the landscape easier to navigate, allowing algorithms to quickly find a stable, consistent solution, especially in less complex or more predictable environments.

Conversely, a **large average distance** is preferred for exploring the solution space more thoroughly to identify diverse and widely spaced solutions, particularly in complex, highly multi-modal problems.

In AS-LSP-MOEA, the decision-maker must choose between a quick, potentially less diverse solution or a more diversified solution that takes longer, depending on whether the priority is speed and efficiency or thorough exploration of the solution space.

- **Average Distance between Pareto Local Optima in the variable space (Plo_DistAvg)** It measures the average distance between Pareto local optima with respect to multiple objectives simultaneously but may not represent the global Pareto front. This feature provides insight into how spread out or clustered these trade-off solutions are within the decision variable space.

  The best values follow the same logic as the previously discussed feature.

Global Features

These features describe the overall structure of the optimization landscape without focusing on specific local neighborhoods. They include measures such as the correlation among objective values, the average and maximum distances among solutions in both the variable and objective spaces, the proportion of non-dominated solutions, and the hypervolume covered by these solutions.

The selected global features for AS-LSP-MOEA are as follows.

- **Average Distance among Solutions in the Objective Space (DistAvg_Obj)**: This feature measures the average of distances between any pair of solutions in the objective space. It helps to understand the spread of solutions in terms of their objective values.

  Higher values are favored when maintaining diversity and thoroughly exploring the objective space is important. Conversely, lower values are preferred when the focus is on convergence and fine-tuning solutions around the Pareto front, often in the later stages of optimization. This tighter clustering of solutions helps in refining the most promising areas of the objective space, supporting a more targeted and intensive search.

- **Average Distance among Solutions in the Variable Space (DistAvg_Var)**: This feature calculates the average of distances between any pair of solutions in the variable space, indicating the spread of solutions in terms of their decision variables.

  The preferred values follow the same logic as the feature previously described.

Evolvability Features

Evolvability measures the responsiveness of the optimization landscape to small changes in decision variables, indicating how these changes can lead to improvements in ob-

jective values. High evolvability suggests that small variation operators can efficiently explore the search space, leading to faster convergence. Algorithms that leverage high evolvability are often more efficient in finding high-quality solutions quickly.

- **Average Distance from Neighbors in the Objective Space (DistAvgNeig_Obj)**: This feature measures the average distance between each solution and its neighboring solutions in the objective space. It provides insights into the local distribution of solutions and the density of the Pareto front.

  The preferred value depends on the desired balance between exploration and exploitation in the optimization process. A **smaller distance** is favored when the goal is to fine-tune solutions and achieve convergence. In contrast, a **larger distance** is preferred for maintaining diversity, allowing the algorithm to explore a wider range of trade-offs and avoid premature convergence. The choice of preferred value reflects whether the priority is on precision or broad exploration.

- **Ratio of the Average Distance from Neighbors in the Objective and Variable Spaces (DistAvgNeig_ObjVar)**: It measures how changes in decision variables translate into changes in objective values by comparing the average distances between neighboring solutions in both spaces. A **higher ratio** suggests that small changes in variables lead to significant shifts in objectives, promoting diverse solution discovery but potentially increasing ruggedness. A **lower ratio** implies a smoother landscape, where large variable changes cause only minor objective adjustments, supporting stable, gradual improvements.

Ruggedness Features

Ruggedness refers to the degree of irregularity or roughness in the fitness landscape, characterized by numerous peaks and valleys indicating a high level of complexity with many local optima. High ruggedness increases the difficulty of the search process, requiring a careful balance between exploration and exploitation to avoid premature convergence and effectively navigate the landscape. Robust algorithms that incorporate adaptive mechanisms or hybrid approaches are often necessary to handle rugged landscapes, ensuring a comprehensive search that can identify global optima despite the challenging terrain.

- **Neighbor's Correlation of the Proportion of Dominating Neighbors (Dominating_CorNeig)**: This feature evaluates the correlation between a solution's neighbors and the proportion of those neighbors that dominate the solution.

  The preferred correlation values, ranging from $[-1, 1]$, depend on the decision-maker's goals and the landscape's nature. A **higher positive correlation** (close to 1) is ideal for smoother landscapes where consistent patterns among neighboring solutions aid in predictability, stability, and gradual refinement toward the Pareto front. **Negative correlation** (closer to -1) is favored in rugged landscapes where diversity and exploration are crucial, as it promotes variability and helps avoid

local optima by encouraging broader exploration of the search space. **Near-zero correlation** indicates little to no relationship between neighbors, which can be advantageous in highly complex or unpredictable landscapes, allowing for flexibility in both exploration and exploitation strategies.

- **Neighbor's Correlation of the Average Distance from Neighbors in the Objective Space (DistCorNeig_Obj)**: This feature measures how consistently the distances between a solution and its neighbors in the objective space are correlated across the landscape. In simpler terms, it assesses whether the relative positions of solutions in the objective space are similarly structured among different regions of the landscape.

  The decision-maker's preference depends on whether the priority is on stability and refinement, flexibility in exploration, or ensuring diversity in challenging landscapes.

  Higher positive values are preferred for smooth, stable landscapes with goals of predictability and efficient convergence. **Values near zero** are favored in complex or unpredictable landscapes where flexibility and broad exploration are important. **Negative values** are best suited for highly rugged landscapes, emphasizing diversity and thorough exploration.

5.4 Correlated Criteria

We recognize that some of the criteria within the groups of performance indicators and landscape features are correlated with each other. This correlation is logical because we use the obtained solutions and the same reference set for calculations. However, they each apply different formulas and highlight different aspects to provide an overall performance assessment.

In the context of LSP and to address this correlation, we define two different configurations of the criteria: one comprising the set of non-correlated criteria (cf. Figure 5.3), and another including all the criteria from the Figure 5.2.

The first configuration aims to ensure that only the independent criteria are considered, reducing redundancy and potential bias in the evaluation process. In contrast, the second configuration includes all relevant criteria to provide a comprehensive assessment, despite the inherent correlations.

Additionally, the performance indicators are divided into three groups—convergence, diversity, and convergence-diversity—that may overlap. Despite some redundancy and overlap, these indicators are complementary, and no single indicator or group can fully assess an algorithm's performance on its own.

To minimize the disproportionate influence of correlated and overlapping criteria on the overall score, they are grouped together. This approach ensures that no single criterion within the correlated group dominates the evaluation, maintaining a balanced and fair scoring system without overemphasizing any particular aspect.

Group	Sub-Group	Criterion	Preferred Values
Execution Metrics		Execution Time **t**	Lower
		Memory Usage **m**	Lower
		Iterations **i**	100%
Performance Indicators	Convergence	Generational Distance **GD**	Lower, $GD = 0$
		Additive ϵ-Indicator $\mathbf{I}_{\epsilon+}$	Smaller
		Maximum Pareto Front Error **MPFE**	Lower, $MPFE = 0$
		R1 **Indicator**	1 (values from [0, 1])
	Convergence -Diversity	HyperVolume **HV**	Higher
		Inverted Generational Distance **IGD**	Lower, $IGD = 0$
		R2 **Indicator**	-1 (values from [-1, 1])
	Diversity	Spacing **Sp**	Lower, $Sp = 0$
Landscape Features	multi-modality	Avg. Dist. between Single-obj. Local Optima in the Var. space **Slo_DistAvg**	- Higher for diversity, - Lower for fine-tuning and fast convergence
		Avg. Dist. between Pareto Local Optima in the Var. space **Plo_DistAvg**	
	Global	Avg. Dist. among Solutions in the Obj. Space **DistAvg_Obj**	
		Avg. Dist. among Solutions in the Var. Space **DistAvg_Var**	
	Evolvability	Avg. Dist. from Neighbors in the Obj. Space **DistAvgNeig_Obj**	
		Ratio of the Avg. Dist. from Neighbors in the Obj. and Var. Spaces **DistAvgNeig_ObjVar**	
	Ruggedness	Neighbor's Correlation of the Proportion of Dominating Neighbors **Dominating_CorNeig**	- Close to 1: Gradual refinement. - Close to -1: Broader exploration. - Near 0: Flexibility in Exploration and exploitation.
		Neighbor's Correlation of the Avg. Dist. from Neighbors in the Obj. Space **DistCorNeig_Obj**	

Table 5.1: Preferred Values for Evaluation Criteria

5.5 Utility Values

For the metrics that follow a single direction, either minimizing or maximizing, we employ a direct evaluation approach to determine their impact on the overall outcome incorporating variations for testing of different scenarios. However, when it comes to landscape features, where the decision-maker's preferences may vary between fine-tuning, broad exploration, or a combination of both, a more nuanced approach is required. In general, we experiment with different utility values to reflect these varying preferences.

After collecting the various metrics and features used as criteria in AS-LSP-MOEA, we applied both normalization and non-normalization to the metrics. The reason is that some metrics, such as $R1$ indicators, are already normalized. For other metrics, preferred values can be known independently of the problem, such as Execution Time metric. However, some metrics and features are context-dependent, as discussed earlier, like the $DistAvgNeig\_Obj$. For these context-dependent metrics, we may understand the trends in the decision-maker's preferences—such as favoring fast convergence, broader exploration, or a balance between them—but the exact ranges cannot be precisely defined due to the stochastic nature and the complexity of variable and objective spaces.

To address this, we use two approaches. The first approach involves normalizing all the metrics using min-max normalization and applying the decision-maker's preference trends (whether minimizing or maximizing a metric is preferred). This approach has the disadvantage of being influenced by outliers. In this specific case, we cannot simply remove outliers, as they might represent the best-performing algorithms for the problem. Normalization in the presence of outliers tends to separate algorithms into groups—those including the outliers and the rest. The problem with this kind of separation is that it leads to similar results within each group, which can be interpreted more as a classification rather than a true ranking.

The second approach involves running the algorithms prior to applying our system to collect some historical data on each problem, providing an idea of the value ranges. This could be disadvantageous for algorithms that explore new regions of the design space, as they might be penalized for deviating from established ranges. We will examine both approaches in the Tests and Results chapter, but here we present two utility setups that illustrate the preferred trends using normalized values.

5.5.1 Utility Setup 1

In this setup, depicted in the Figure 5.4, the utility values are designed to favor broader exploration and diversity, which is evident from the utility curves that increase with metrics associated with exploration, such as $DistAvgNeig\_Obj$. The positive slope in these plots suggests that higher values of these metrics are considered more beneficial. This aligns with a strategy that values diversity in the solution space and promotes a wide-ranging search for optimal solutions.

For $Dominating\_CorNeig$, the utility values decrease as the input increases. This indicates that lower values, particularly around -1, are preferred for diversity. The

negative slope suggests that the system favors solutions where diversity, as captured by *Dominating_CorNeig*, is maximized when the metric is closer to −1.

For the execution metrics and performance indicators, the trends follow a single direction, either minimizing or maximizing, depending on the specific metric. These trends are consistent and do not change from one setup to another in the optimization process. Metrics like Memory Usage and Execution Time the preference is for more efficient algorithms that minimize resource consumption.

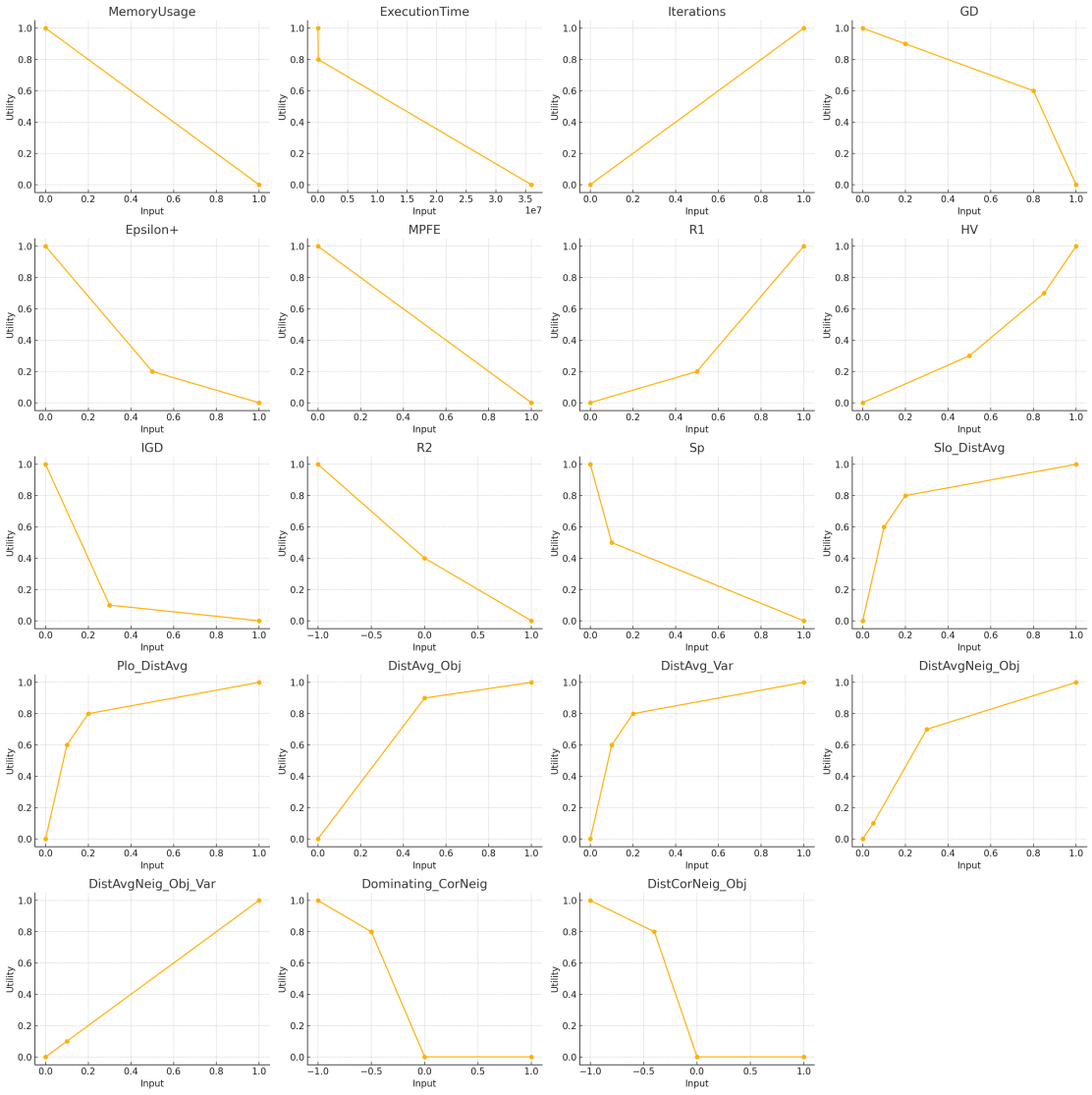


Figure 5.4: Utility values For Diversity

5.5.2 Utility Setup 2

Figure 5.5shows the utility values preferred for fine-tuning and convergence. The multi-modality, global, and evolvability features have decreased utility values because smaller values are preferred in these contexts. In contrast, the utility values for ruggedness features increase, as higher values closer to 1 are favored.

5.6 Aggregators

Two sets of aggregators are defined: Ag.1 and Ag.2. Ag.1 is less restrictive compared to Ag.2 at the "algorithm" level, as well as in the "execution metrics" and sub-groups of the "performance indicators" features. However, Ag.1 is more restrictive in the four sub-groups of "landscape features" compared to Ag.2. These two sets of aggregators will be applied in different configurations and their impact will be compared to understand how the varying levels of restrictiveness or simultaneity/replaceability influence algorithm selection and performance across different scenarios.

Table 5.2: Operators

			Ag.1	**Ag.2**
Algorithm			CA	$C-$
	Execution Metrics		$D+$	$C-+$
	Performance Indicators		$C--$	$C--$
		Convergence	DA	$C--$
		Conv-Div	DA	$C--$
	Features Landscape		$C--$	$C--$
		multi-modality	$C+$	$D-+$
		Global	$C+$	$D-+$
		Evolvability	$C+$	$D-+$
		Ruggedness	$C+$	$D-+$

5.7 Alternatives

The portfolio presented to the decision-maker is composed of 14 Multi-Objective Evolutionary Algorithms (MOEAs). Each algorithm in this portfolio offers different mechanisms for navigating the complex search spaces characteristic of multi-objective problems, such as decomposition, indicator-based selection, reference point guidance, and others. This diverse portfolio provides the decision-maker with a wide range of options to pre-select algorithms that will be evaluated by AS-LSP-MOEA.

- **AGEMOEA-II**: Age-Layered Population Structure for Multi-Objective Evolutionary Algorithms II [104]. AGEMOEA-II introduces an age-based population

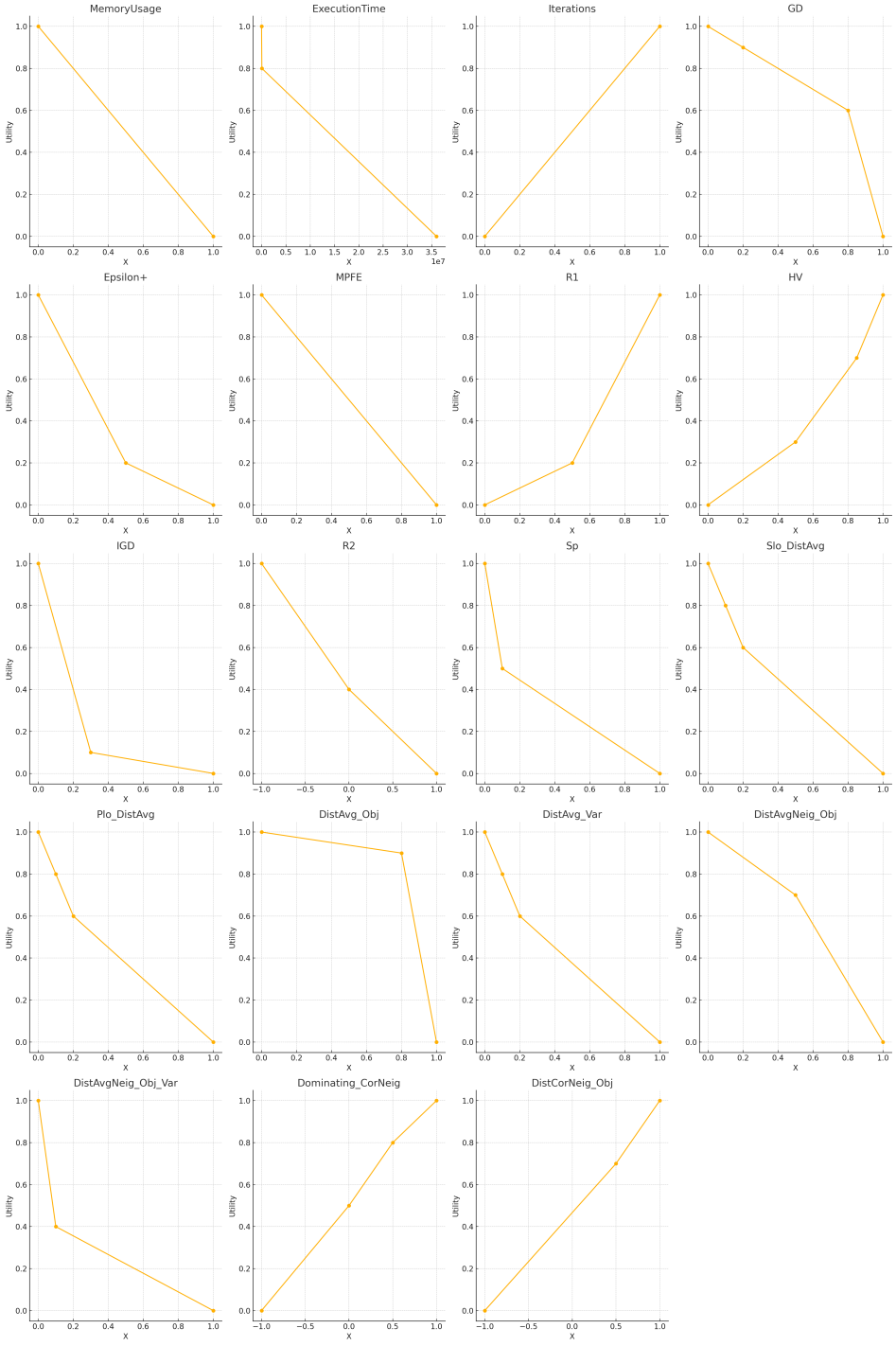


Figure 5.5: Utility values For Fast Convergence

structure to maintain diversity by segregating individuals into different age layers. This approach helps to preserve older, potentially more diverse solutions while allowing younger, fitter solutions to emerge and compete within the population.

- **DBEA**: Decomposition-Based Evolutionary Algorithm [6]. DBEA decomposes a multi-objective optimization problem into a set of scalar optimization problems, each of which is optimized individually. This approach leverages problem decomposition strategies, such as Chebyshev or weighted sum methods, to guide the search towards Pareto-optimal solutions by focusing on different regions of the Pareto front.

- **EpsilonMOEA**: Epsilon Dominance Multi-Objective Evolutionary Algorithm [33]. EpsilonMOEA uses the concept of epsilon-dominance to maintain a diverse set of solutions. By introducing an epsilon threshold, this algorithm controls the resolution of the Pareto front, allowing the decision-maker to specify the desired level of diversity and convergence in the final set of solutions.

- **GDE3**: Generalized Differential Evolution 3 [81]. GDE3 is an extension of the Differential Evolution (DE) algorithm tailored for multi-objective optimization. It combines DE's powerful mutation and crossover strategies with non-dominated sorting to effectively handle the trade-offs between multiple conflicting objectives.

- **MSOPS**: Multiple Single Objective Pareto Sampling [64]. MSOPS transforms a multi-objective problem into several single-objective problems by sampling different weights or objectives. This allows the algorithm to tackle the multi-objective problem from various perspectives, ultimately combining the results to form a global Pareto front.

- **NSGA-II**: Non-dominated Sorting Genetic Algorithm II [32]. NSGA-II is one of the most widely used multi-objective evolutionary algorithms. It ranks individuals in the population using a fast non-dominated sorting approach and applies a crowding distance mechanism to maintain diversity, ensuring that the solutions are well-distributed along the Pareto front.

- **NSGA-III**: Non-dominated Sorting Genetic Algorithm III [34]. NSGA-III is an extension of NSGA-II designed to handle many-objective optimization problems (problems with more than three objectives). It uses a reference point-based approach to maintain diversity in higher-dimensional objective spaces, making it suitable for more complex optimization tasks.

- **U-NSGA-III**: Unified NSGA-III [118]. U-NSGA-III extends NSGA-III by unifying concepts from NSGA-II and NSGA-III, making it versatile for both multi-objective and many-objective optimization problems. This approach allows for a more consistent performance across problems with varying numbers of objectives.

- **PESA2**: Pareto Envelope-based Selection Algorithm II [29]. PESA2 uses a grid-based approach to maintain diversity within the population. By dividing the objective space into hypergrid cells, PESA2 ensures that the selection process favors solutions that contribute to a well-distributed Pareto front.
- **RVEA**: Reference Vector Guided Evolutionary Algorithm [25]. RVEA employs reference vectors to guide the search process, particularly in problems with many objectives. These vectors help to maintain a balance between convergence and diversity, guiding the population towards different regions of the Pareto front based on the reference points.
- **SPEA2**: Strength Pareto Evolutionary Algorithm II [144]. SPEA2 improves upon its predecessor by using a fitness assignment strategy that combines dominance rank and density estimation. This approach helps to maintain a well-distributed and high-quality set of solutions along the Pareto front.
- **VEGA**: Vector Evaluated Genetic Algorithm [116]. VEGA assigns different objectives to different individuals during the selection process, effectively creating subpopulations that focus on different aspects of the multi-objective problem. This approach helps to maintain diversity but can sometimes lead to bias towards certain objectives.

Chapter 6

Tests and Results

In this section, we explore the use of LSP method in evaluation and ranking of the multi-objective optimization algorithms via the AS-LSP-MOEA tool.

6.1 Tests

As described in the methodology and development chapter 5, we constructed a portfolio of a variety of MOEAs. We use the MOEA Framework [58], a Java-based framework that includes a comprehensive list of problem benchmarks and MOEA implementations. We begin by testing AS-LSP-MOEA on the ZDT problem family [143], as these problems are relatively simple and increase in complexity. Following this, we test AS-LSP-MOEA on the LSMOP problem family [26] to assess its performance in large-scale optimization scenarios, which are relevant to the large-scale nature of the third case study, Evolutionary Circuit Design, as will be discussed later. All the tested problems are bi-objective.

In a stochastic context, algorithms include elements of randomness, which causes their behavior and outcomes to vary between runs, even when applied to the same problem instance. To capture this variability and ensure a reliable assessment of the algorithm's performance, it is necessary to execute the algorithm multiple times. This approach allows us to average the performance metrics, reducing the impact of any outliers or extreme results that might occur due to randomness in a single run. Additionally, repeated executions enable the evaluation of the algorithm's robustness by showing how consistently it performs under different random conditions. By running the algorithm multiple times, we also increase the chances of exploring different parts of the solution space, leading to the discovery of high-quality, diverse solutions. For the tests, we execute the same portfolio on the same problem 10 times. Ideally, it should be run at least 100 times to obtain more statistically robust results, but due to limited resources, we were only able to perform 10 runs.

We test, for all the problems, the non-normalized version of the metrics.

6.1.1 Problems

ZDT Problems

The ZDT family [143] of test problems, named after their creators Zitzler, Deb, and Thiele, is a set of benchmark functions widely used in the field of MOO, particularly for evaluating the performance of evolutionary algorithms. These problems are two-objective optimization tasks and are designed to present a broad range of challenges to optimization algorithms. We use ZDT1, ZDT2, ZDT3, and ZDT6 from MOEA framework problems in our tests.

- **ZDT1** is characterized by a simple convex Pareto front, making it one of the easier benchmark problems. The primary challenge of ZDT1 lies in the algorithm's ability to converge to the Pareto front and maintain a well-distributed set of solutions across this front. Since the Pareto front is convex, the problem primarily tests an algorithm's effectiveness in achieving a uniform distribution of solutions along a smooth and continuous front. 30 decision variables are used.
- **ZDT2** is similar to ZDT1 in structure but differs in that it features a non-convex Pareto front. This non-convexity introduces additional complexity, requiring the optimization algorithm to handle non-linearities in the objective space. 30 decision variables are used.
- **ZDT3** presents a more complex scenario by featuring a discontinuous Pareto front. The front is made up of several disconnected segments, which tests an algorithm's ability to not only converge to these distinct regions but also maintain diversity within each segment. 30 decision variables are used.
- **ZDT6** features a non-uniform Pareto front, where the solutions are densely packed in one region and sparse in another. This problem is particularly challenging because it requires the algorithm to maintain diversity across the Pareto front, even in regions where solutions are harder to find or are less densely populated. 10 decision variables are used.

For the tests of AS-LSP-MOEA, we use the following alternatives from the initial portfolio: AGEMOEAII, DBEA, EpsilonMOEA, GDE3, MSOPS, NSGA-II, NSGA-III, PESA2, RVEA, SPEA2, UNSGA-III, and VEGA.

LSMOP Problems

The LSMOP (Large-Scale Multi-Objective Problems) family [26] of test problems was designed to address the challenges associated with large-scale multi-objective and many-objective optimization. These problems are specifically constructed to test the performance of optimization algorithms in environments with a large number of decision variables and complex fitness landscapes. We use LSMOP1 and LSMOP2 from MOEA framework in our tests.

- **LSMOP1:** LSMOP1 typically involves a unimodal fitness landscape with linear Pareto fronts, making it one of the simpler problems in the LSMOP suite. It serves as a baseline for testing basic algorithm performance. With 1,000 decision variables, LSMOP1 is still considered a large-scale problem, effectively testing the scalability of optimization algorithms in handling significant numbers of variables.
- **LSMOP2:** LSMOP2 introduces a partially separable fitness landscape with mixed modality, adding a layer of complexity to the optimization process. This problem demands more sophisticated handling by algorithms due to its increased complexity. With 2,000 decision variables, LSMOP2 further amplifies the problem's difficulty, making it a more challenging test for algorithms as they navigate the intricacies of its fitness landscape.

6.1.2 Case study: Evolutionary Circuit Design Problem

Evolutionary Circuit Design (ECD) [124] is a technique that uses evolutionary algorithms to automate the design and optimization of analog and digital circuits. Unlike traditional design methodologies, such as Karnaugh maps and Quine-McCluskey methods, which rely on predefined topologies and manual optimization, ECD enables the exploration of unconventional and innovative circuit topologies. By generating, evaluating, and evolving potential solutions iteratively, ECD explores a vast design space, allowing for the discovery of circuit designs that might be overlooked by more conventional approaches.

Each potential circuit design is represented by a set of parameters, including component sizes (such as gates and transistors), connections, and overall circuit topology. This design space is high-dimensional, often containing thousands of possible configurations, making exhaustive search impractical. The design space involves both continuous variables, such as component values, and discrete variables, like circuit topologies.

In ECD, multiple objectives—such as minimizing power consumption, reducing time, and maximizing operating frequency—must be optimized simultaneously. A population-based search is employed, where potential solutions evolve over time through various operators, depending on the specific algorithm used.

For our case study, we focus on the design at the gate level, where the circuit is constructed based on individual logic gates and their interconnections. We represent the circuit as a matrix of vectors, with each vector corresponding to a gate, including its inputs and outputs, as depicted in Figure 6.2. Initially, the representation involves both integers and doubles; however, to avoid mixing variable types, we employ a real-valued representation for all variables. For the sake of visualization, the values are shown as integers in the figure.

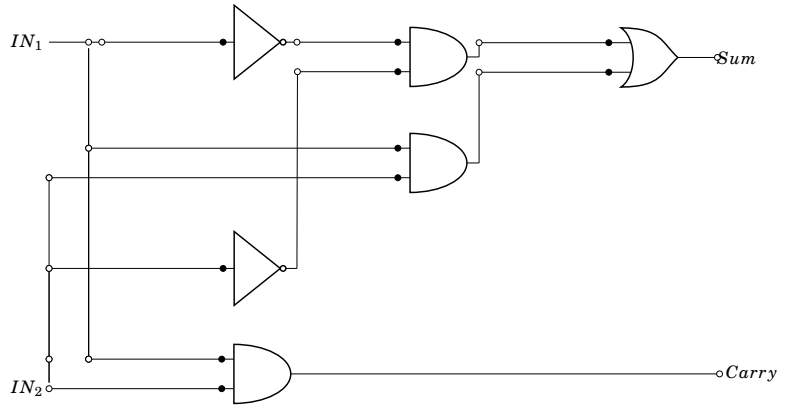


Figure 6.1: Two-bit Half Adder

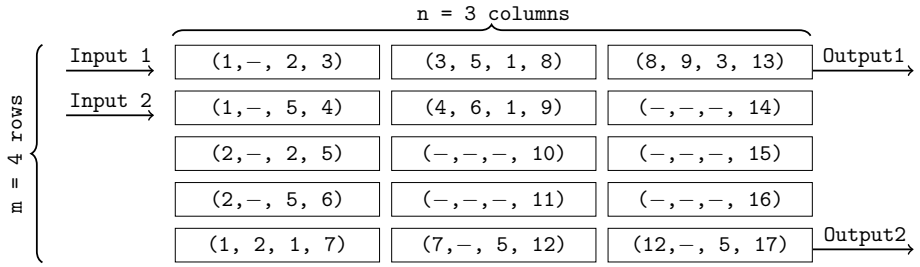


Figure 6.2: Matrix Representation of a 2-bit Half Adder

Figure 6.2 represents the 2-bit Half Adder of the Figure 6.1, where each cell corresponds to a gate. The first two positions in the vector of each cell represent the inputs, the third position is the identifier of the gate, and the last position is the output of the gate to be connected with the rest of the circuit. For instance, the first cell (1, −, 2, 3) indicates that we have an inverter (gate identifier 2) with input identifier 1 and output identifier 3. The list of gates is as follows: And = 1, Not = 2, Or = 3, Xor = 4, and Link = 5.

In this case study, we aim to maximize the correctness of the circuit and minimize its runtime. Correctness refers to the accuracy with which the circuit performs its intended function. Specifically, correctness is defined by the degree to which the outputs generated by the circuit match the expected results as described in the truth table. A circuit is considered correct if, for every possible input combination, the circuit's output aligns with the output specified in the truth table. This ensures that the circuit performs its logical operations accurately and consistently under all conditions.

The ECD problem can be classified as a large-scale problem depending on the size of this matrix. For instance, if the matrix size is 20x20 and each vector representing a gate has a length of 4 positions, the total number of variables would be 20x20x4,

equating to 1600. This large number of variables clearly classifies the problem as a Large-Scale problem, which is why we tested AS-LSP-MOEA on LSMOP benchmarks for its assessment.

For testing purposes, we considered ECD problem as a continuous problem with real-valued variables (null decimals). However, in reality, ECD can be perfectly represented using integer or binary variables. Commonly, continuous optimization algorithms are used on ECD [75, 46, 63] and other real-world problems [18, 3], with discretization techniques applied for the final representations.

6.1.3 Weights of Criteria

In this section, we provide different configurations of the weights assigned to each criterion used in calculating the overall score of the alternatives. The process involves experimenting with various weight assignments for the same criterion to observe and compare how these changes impact the final ranking of the alternatives.

Comprehensive Criteria Weights

In all the configurations, there is a balanced distribution of emphasis across execution metrics, performance indicators, and features landscape.

Within the execution metrics, time (t) receives weight of 0.3, indicating a moderate focus on how quickly the algorithms perform, while memory (m) is weighted at 0.2, showing less emphasis on memory usage, as memory is a critical resource but can be enhanced relatively easily. The number of iterations (i) completed within the allocated time provides valuable insight into the computational complexity of an optimization algorithm. For example, if the evaluation function of the generated solutions or the operators used by the algorithm are complex, this increases the algorithm's overall complexity, leading to fewer iterations being completed than preferred by the decision-maker. the iteration metric has a weight of 0.5.

Overall, execution metrics are given a total weight of 0.2, reflecting a low emphasis because they cannot justify a higher weight. A complex algorithm may consume more resources but deliver superior solutions, whereas a simpler algorithm might use fewer resources but fail to converge effectively. Thus, execution metrics should not heavily influence the assessment of an algorithm's overall quality.

The Performance Indicators group is attributed an equal weight with the landscape Features which is 0.4 for each one. Within this group, the criteria of Spacing (Sp) hold the higher levels of importance than the Convergence Group and have a weight of 0.4 each one. The Convergence-Diversity group is assigned relatively low weight (0.2) to minimize redundancy, as it overlaps with other sub-groups within the Performance Indicators group. Convergence indicators such as GD and $R1$ indicator have a weight of 0.2 each one, while $I_\epsilon+$, and $MPFE$ have a weight of 0.3 each one. The Convergence-Diversity indicators, IGD and HV, have the higher weights within the group, 0.45 each one. The $R2$ indicator is assigned low weight of 0.1 because it is considered somewhat

redundant; however, it remains in the criteria tree as its complement the metrics within its sub-group.

The Features Landscape group includes several sub-groups: Multi-modality, Global, Evolvability, and Ruggedness, each one with an equal weight of 0.25. Each of these features is further divided into two elementary criteria receiving equal weights of 0.5.

Weights for Non-Correlated Criteria

For performance indicators, the emphasis is slightly more diverse. Generational Distance (GD) has a lower emphasis with a weight of 0.2, while MPFFE and Hypervolume (HV) both receive a weight of 0.3, suggesting a moderate focus on the accuracy of Pareto fronts and the quality of the solution set. Spread (Sp) also has a lower emphasis with a weight of 0.2. The total weight for performance indicators is 0.4, indicating a moderate level of importance in the overall evaluation.

In the features landscape group, the multi-modality criterion is divided equally between its child nodes $Slo\_DistAvg$ and $Plo\_DistAvg$, each receiving a weight of 0.5 within their parent node. This balance indicates that both aspects of multi-modality are equally important. The overall weight for multi-modality is 0.25, suggesting a moderate emphasis on this feature. Similarly, the global features are equally weighted between $DistAvg\_Obj$ and $DistAvg\_Var$ at 0.5 each, with the overall global feature also receiving a moderate emphasis with a weight of 0.25.

6.1.4 Configurations

We use the alternatives defined in the chapter 5 and pre-select algorithms for each family of problems. We then launch AS-LSP-MOEA with different configurations, as shown in Table 6.1.

Table 6.1: Configurations of AS-LSP-MOEA

Utility 1	**Utility 1**	**Utility 2**	**Utility 2**
Aggregators 1	**Aggregators 2**	**Aggregators 1**	**Aggregators 2**
Configuration 1	Configuration 2	Configuration 3	Configuration 4

We utilize the complete list of criteria as defined in Chapter 5, for all problems.

6.1.5 Operational Parameters

For the MOEAs, each algorithm was run for 100,000 evaluations with a population size of 100 individuals. The maximum allowed runtime for each algorithm was set to 5 minutes. All other settings were configured according to the default parameters provided by the MOEA Framework.

6.2 Results

This section details the outcomes of our experiments, presenting the evaluation and ranking of the selected MOEAs when applied to the various problem benchmarks using AS-LSP-MOEA. We analyze the results to determine which algorithm meets the decision-maker preferences under different configurations.

6.2.1 ZDT Problems

ZDT1 Problem

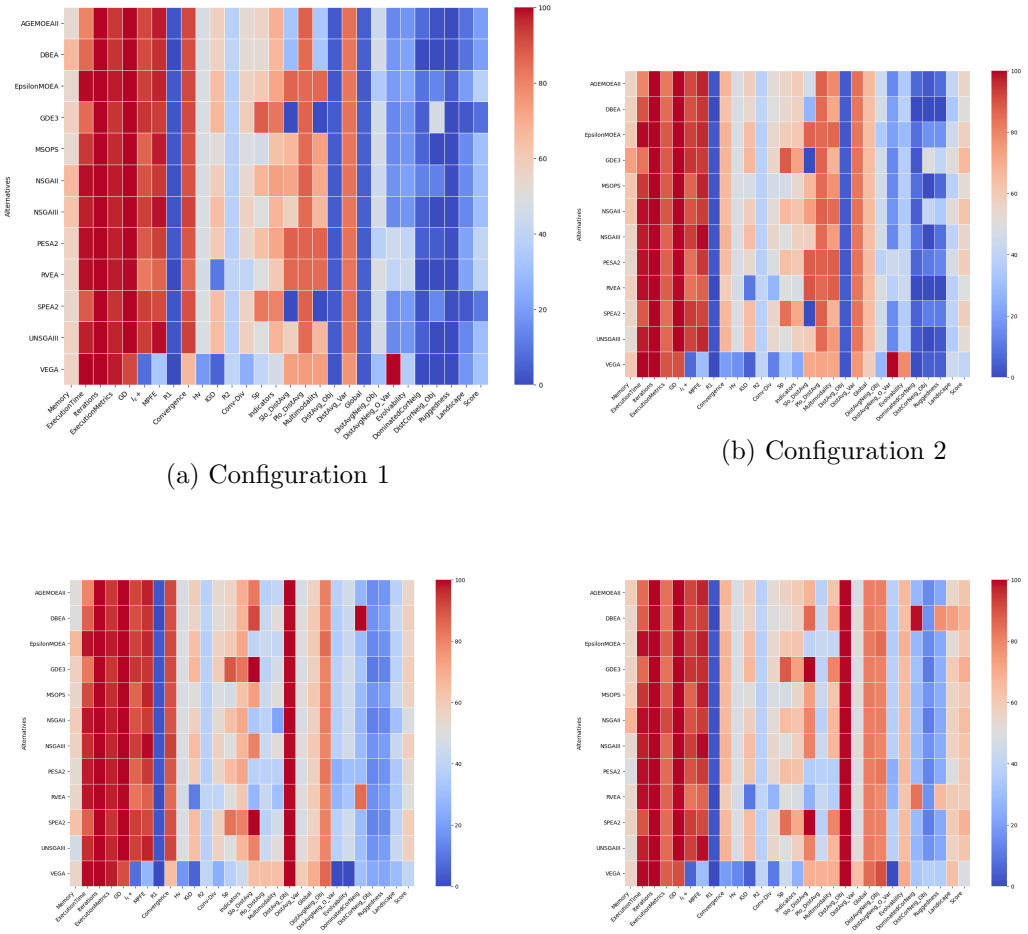
Analysis The heatmaps in Figure 6.3 illustrate the evaluation results of the MOEAs on the ZDT1 problem. Each row corresponds to an algorithm, and each column represents a different metric or feature. The color intensity indicates the relative performance of the algorithms on each criterion, with darker red signifying higher values and darker blue representing lower values. This visualization allows for a direct comparison of algorithm performance across different aspects of the optimization task.

In Configuration C1, PESA2, RVEA, and EpsilonMOEA achieve overall scores of 40.44%, 38.74%, and 37.72%, respectively. Memory usage and execution time show considerable variation across algorithms, with NSGA-II, NSGA-III, and VEGA displaying best efficiency in managing resources, while AGEMOEA-II and GDE3 demonstrate lower efficiency. All algorithms reach the maximum allowed iterations (100,000), making the Iterations metric less distinctive.

UNSGA-III and NSGA-III exhibit strong performance in convergence metrics, achieving 93% and 92.74%, respectively, while in terms of landscape features, RVEA and PESA2 perform well.

In C2, GDE3 and NSGA-II take the top spots, each securing more than 62% of the overall score. They also excel in both performance metrics and landscape features. In terms of convergence group metrics, NSGA-III and EpsilonMOEA take the top positions.

In C3 and C4, GDE3 and SPEA2 are the top performers. NSGA-III and DBEA aslo emerge as strong choices.



(a) Configuration 1

(b) Configuration 2

(c) Configuration 3

(d) Configuration 4

Figure 6.3: ZDT1 Results 1/2

In terms of overall scores depicted in Figure 6.4, configurations, C3 and C4 generally yield higher scores for most algorithms compared to C1 and C2. This suggests that the settings in C3 and C4 are more favorable to the strengths of these algorithms and the problems. Both C3 and C4 share the same utility value trends, which favor fast convergence and fine-tuning over the diversity of solutions. For instance, GDE3 and SPEA2 show significant improvements in C3 and C4, indicating that these configurations align better with algorithms that are effective at quickly converging to a set of optimal or near-optimal solutions. These configurations prioritize convergence, so algorithms designed to reach the Pareto front quickly and accurately perform well here.

Conversely, C1 and C2 configurations emphasize maintaining diversity. The lower scores in these configurations suggest that the same algorithms may not be as effective at maintaining a diverse set of solutions. This means that while they reach the Pareto front quickly, the solutions are clustered or concentrated in certain areas of the front,

rather than being spread out across the entire front. Among these, C1 is the most restrictive configuration, with conjunctive aggregators and placing the greatest emphasis on diversity, which further challenges algorithms focused on fast convergence.

Meanwhile, VEGA performs moderately well across all configurations but does not reach the top scores, indicating consistent yet not outstanding performance. This suggests that while VEGA is capable of balancing between convergence and diversity, it may lack the necessary evolvability and techniques to effectively handle rugged landscapes, such as escaping from local optima traps.

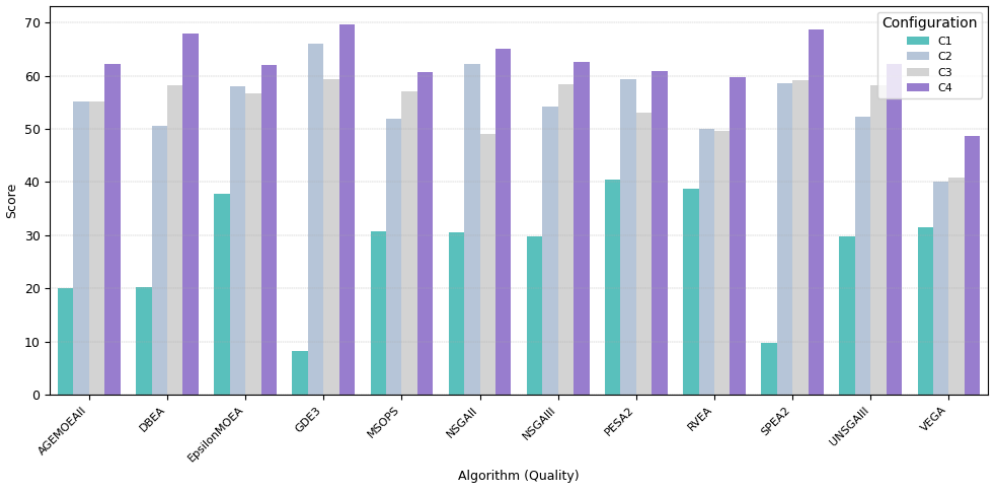


Figure 6.4: Algorithm Quality - ZDT1

Regarding Performance Indicators, Figure 6.5, all algorithms perform from above average to good, with GDE3 achieving 83.57% and SPEA2 80.72% in C3, except for VEGA, which scores below the mean. The Configurations C1 and C3 display the highest values.

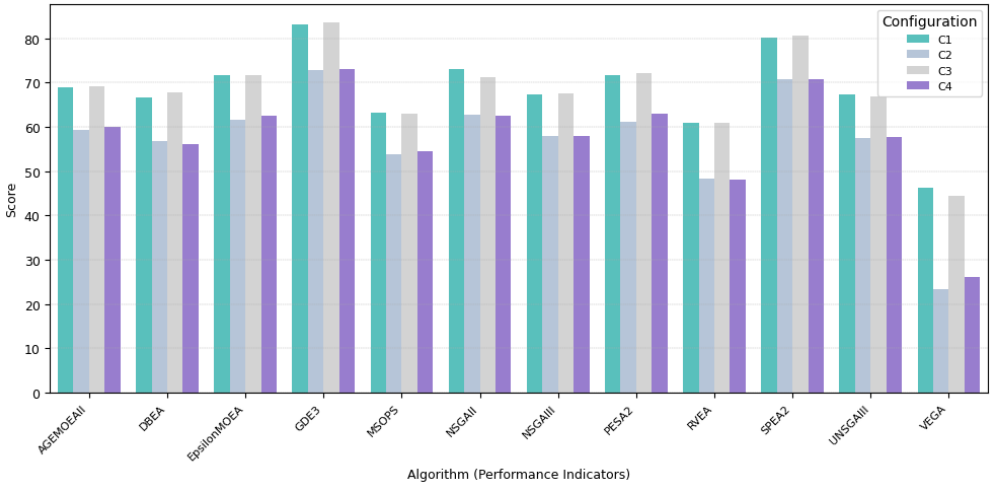


Figure 6.5: Performances Indicators - ZDT1

Figure 6.6 displays the scores of landscape features. DBEA, GDE3, and SPEA2 demonstrate a significant performance boost from C1 to C4, with DBEA improving from 9.12% to 73.52%, GDE3 from 2.77% to 60.37%, and SPEA2 from 3.36% to 59.7%. Overall, the algorithms perform well in C4, likely due to the $D-+$ aggregators applied across all the sub-groups of the landscape features, which seem to favor these configurations.

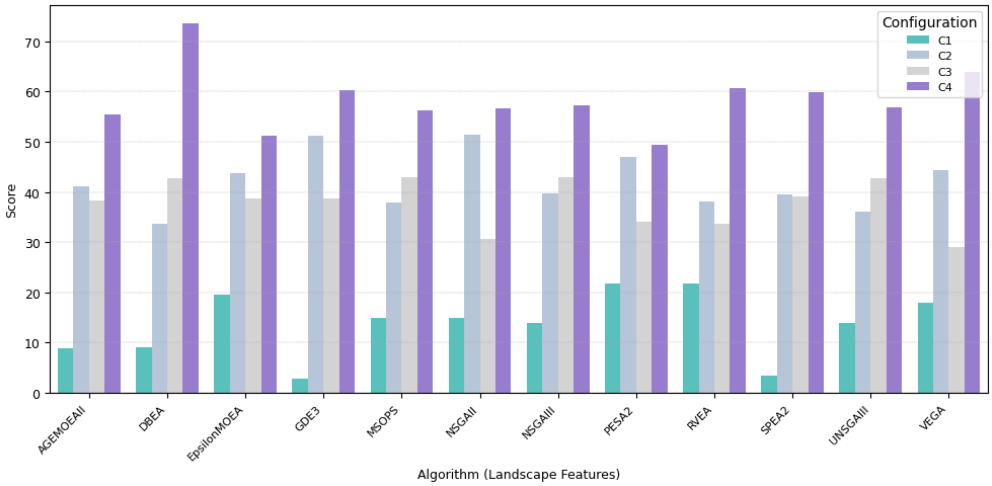


Figure 6.6: Landscape Features - ZDT1

In C1, PESA2 and EpsilonMOEA lead the results, particularly due to their strong ability to handle multi-modal landscapes and their effective exploration in both the variable and objective spaces, achieving scores of 87.03% and 86.07%, respectively.

Discussion The results from Configuration C1 indicate that although PSEA2, RVEA, and EpsilonMOEA perform relatively well compared to other algorithms, their overall scores still fall below the halfway mark. This is likely due to the use of conjunctive aggregators in C1, which tend to penalize algorithms that do not excel across all metrics.

The variation observed in Memory Usage and Execution Time among algorithms like UNSGA-III and NSGA-III because of their high computational complexity, which could impact their scalability in larger problem instances. Conversely, the efficiency of AGEMOEA-II and GDE3 makes them more suitable for resource-constrained environments.

The superior convergence of NSGA-II highlights its effectiveness in reaching the Pareto front, while the performance of GDE3 and SPEA2 in the HV metric suggests they provide a better balance between convergence and diversity.

The strong performance of DBEA in landscape features indicates its ability in handling complex search spaces. VEGA and RVEA's success in the $DistAvg\_Obj$ and $DistAvg\_Var$ metrics points to their ability to maintain diversity.

The high scores, in general, in Configurations C1 and C3 is due to the disjunctive nature of the aggregators used in these configurations, particularly the DA aggregator applied to both Convergence and Convergence-Diversity metrics. This contrasts with the $C--$ aggregator used in C2 and C4, which may not favor such high performance.

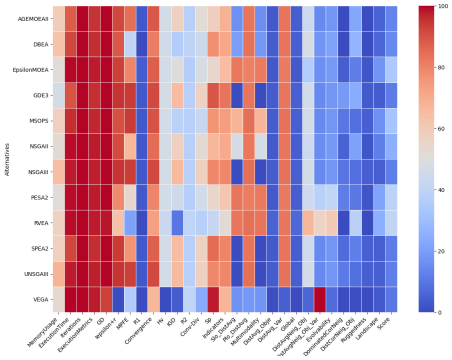
ZDT2 Problem

Analysis In configuration C1, RVEA shows a moderate performance across various criteria, with a mix of light red and blue shades in the heatmap of Figure 6.7a. RVEA, along with PESA2, leads the ranking of the MOEA for ZDT2 under C1, with scores of 40.41% and 38.84%, respectively, followed by EpsilonMOEA with 34.05%. These three algorithms perform very well in multi-modality, with scores of 83.46%, 82.20%, and 81.73%, respectively.

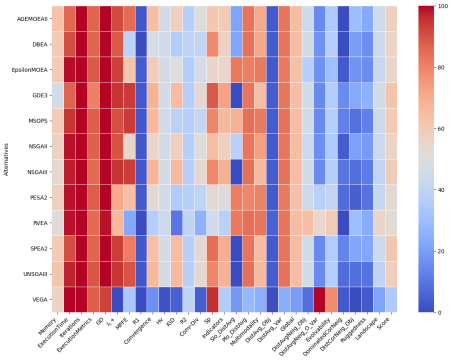
GDE3 exhibits a significant improvement in performance in configurations C2, C3, and C4, with darker red shades across many criteria, particularly in areas related to convergence and landscape features. It leads the list with 62.27% in C2, 60% in C3, and 68.07% in C4. SPEA2 and NSGA-III stand out as strong MOEA performers for ZDT2.

DBEA shows a significant performance boost in C3 and C4 compared to C1 and C2, evident in the deep red colors across several metrics in the heatmaps. Meanwhile, PESA2 stands out in C1, with high scores indicating its effectiveness in maintaining a diverse set of solutions.

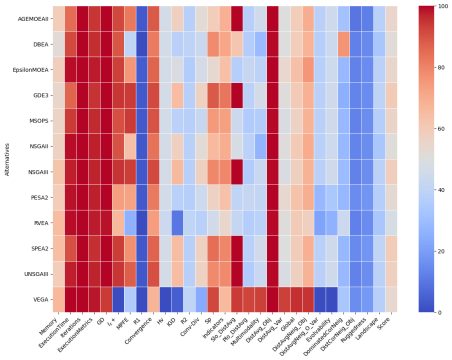
VEGA's performance is consistently below average across all configurations as depicted in Figure 6.7.



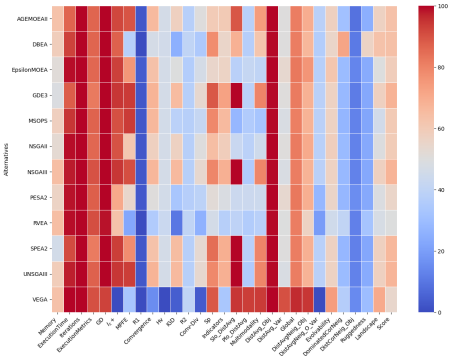
(a) Configuration 1



(b) Configuration 2



(c) Configuration 3



(d) Configuration 4

Figure 6.7: ZDT2 Results

The algorithms are generally performing better in C2, C3, and C4 configurations compared to C1, as depicted in Figure 6.8. However, PESA2 and RVEA stand out by showcasing some consistency across configurations. This suggests that while most algorithms benefit from the conditions set by C2, C3, and C4, PESA2 and RVEA are more versatile, maintaining stable performance regardless of the configuration.

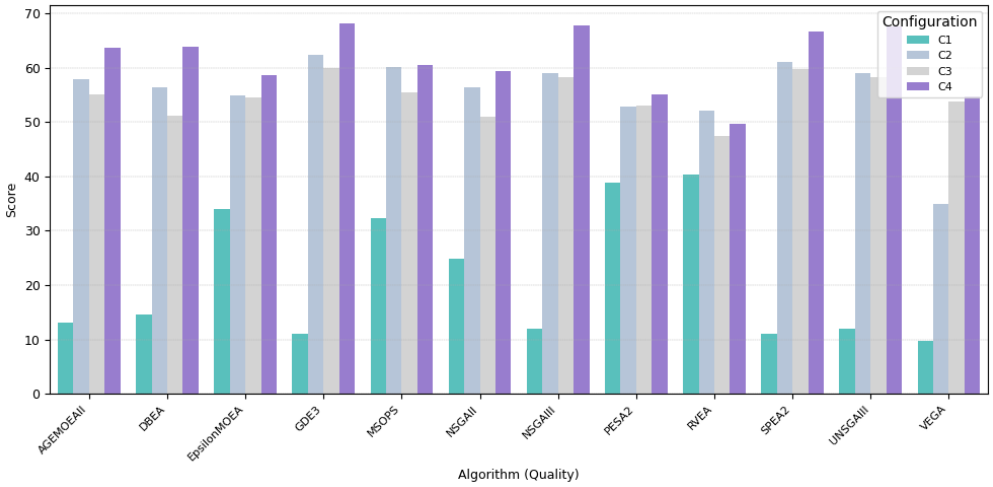


Figure 6.8: Algorithm Quality - ZDT2

GDE3, SPEA2, NSGA-III, and UNSGA-III exhibit higher performance indicators (*cf.* Figure 6.9), particularly in configurations like C1 and C3.

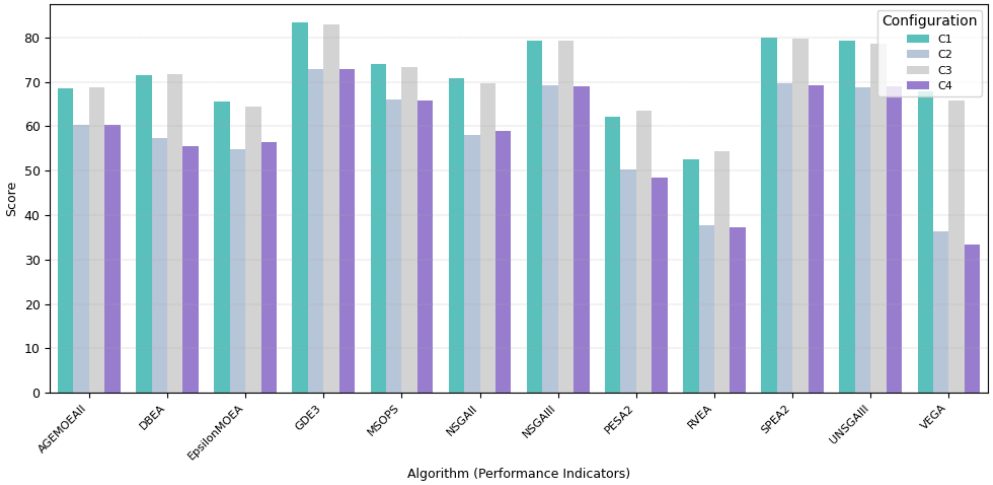


Figure 6.9: Performances Indicators - ZDT2

The algorithm VEGA demonstrates a very high score in the multi-modality criterion under configuration C4 as shown in Figure 6.10. Interestingly, C4 is designed to favor unimodal landscapes rather than multi-modal ones, yet VEGA excels in this area, achieving a score of 94.12%. This performance is notably better, with a 14% lead over GDE3.

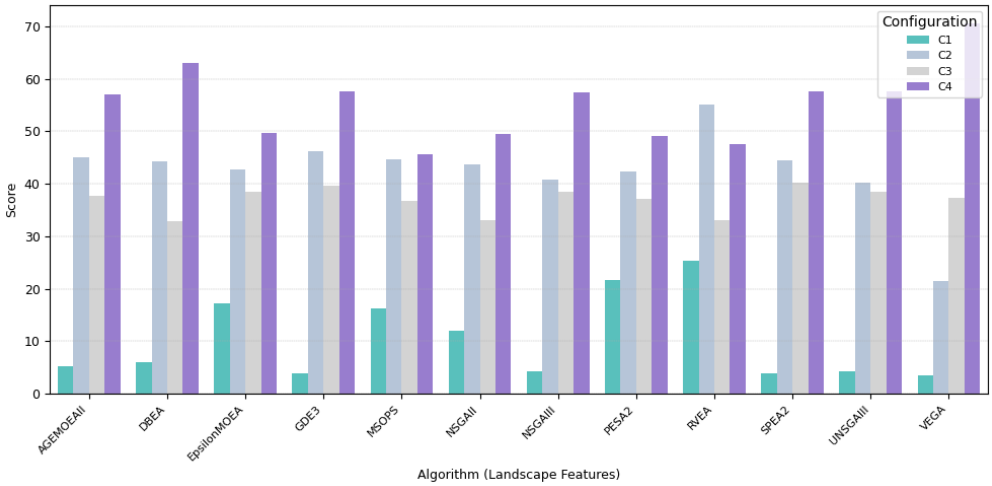


Figure 6.10: Landscape Features - ZDT2

Discussion The moderate performance of RVEA in C1 suggests that it is neither exceptionally strong nor particularly weak under these conditions. The strong performance in multi-modality by PESA2, RVEA, and EpsilonMOEA indicates their ability to handle a spread Pareto front and the existence of multiple single local optima.

The improvement of GDE3 in C3, and C4 suggests that GDE3 is more effective when evaluation criteria focus on convergence efficiency rather than solely on maintaining solution diversity. The strengths of GDE3 as a differential evolution algorithm are highlighted in configurations that prioritize fast convergence and fine-tuning.

The significant performance boost of DBEA in C3 and C4 can be attributed to its decomposition-based approach, which aligns well with the utility functions and aggregators used in these configurations. This alignment allows DBEA to excel in situations prioritizing convergence and efficient exploration of the Pareto front.

PESA2's high performance in C1 suggests that it is particularly adept at maintaining diversity, which aligns well with C1's priorities, allowing it to outperform many other algorithms in this setup.

VEGA's consistent underperformance across all configurations indicates it may struggle with balancing the trade-offs between diversity and convergence. This performance shortfall may be due to its foundational nature and lack of advanced mechanisms compared to more modern algorithms like NSGA-II or GDE3.

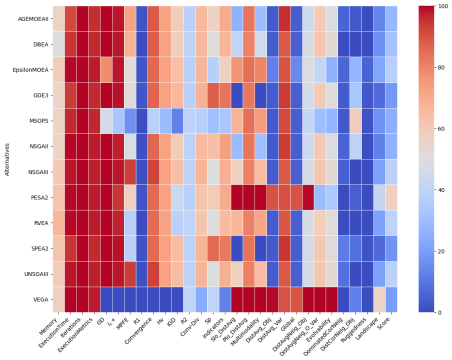
ZDT4 Problem

Analysis In Configuration C1 (see Figure 6.11a), PESA2 and RVEA generally perform well across most metrics, demonstrating high convergence rates and execution efficiency. In contrast, GDE3 and SPEA2 present more mixed results, struggling with landscape features handling.

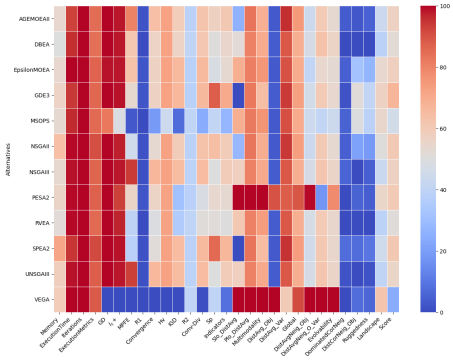
In Configuration C2, GDE3 and RVEA dominate, excelling particularly in convergence and execution metrics, with an overall score of 67.41% and 61.28%, respectively. NSGA-II ranks third but surpasses RVEA in landscape features scores, as depicted in Figure 6.11b.

In Configuration C3 (Figure 6.11c), NSGA-III and UNSGA-III slightly outperform DBEA and RVEA. All four algorithms exhibit similar behavior across the three groups: execution metrics, performance indicators, and landscape features. They have an overall score between 47.31% and 48.27%.

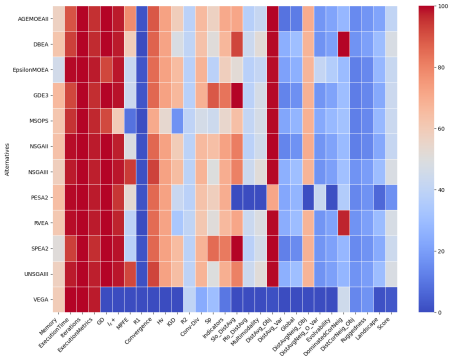
Finally, in Configuration C4, shown in Figure 6.11d, DBEA, GDE3, and SPEA2 stand out with outstanding performance across the board, achieving strong scores in all metrics. RVEA remains a strong contender but shows slight weaknesses in some areas. VEGA continues to underperform in most metrics except in execution metrics and ruggedness scores, where it shows relative strength.



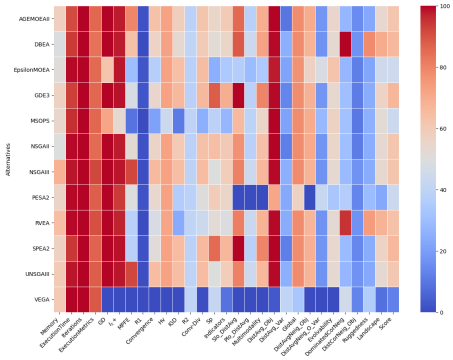
(a) Configuration 1



(b) Configuration 2



(c) Configuration 3



(d) Configuration 4

Figure 6.11: ZDT4 Results

The Figure 6.12 reveals considerable differences in scores across the various configu-

rations (C1 to C4) for each algorithm. For instance, GDE3 exhibits a notable surge in performance under configuration C2, showing a considerable improvement from C1. In contrast, MSOPS maintains consistent performance across C2, C3, and C4.

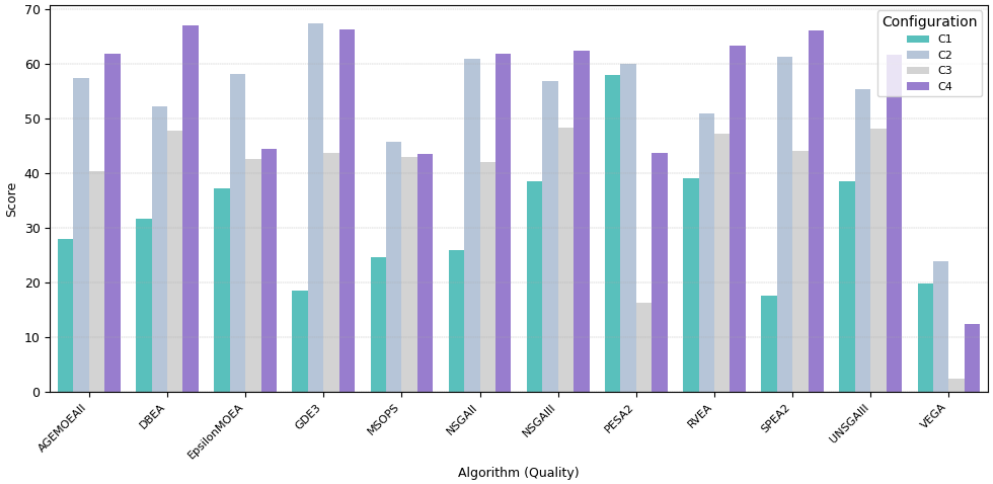


Figure 6.12: Algorithm Quality - ZDT4

In terms of Performance Indicators, Figure 6.13, most of algoithms show consistent scores across all configurations, while MSOPS shows a shift in C3. GDE3, like the others, excels in configurations C1 and C2 but experiences a drop in C3 and C4. SPEA2 stands out as a strong performer also. In contrast, EpsilonMOEA consistently underperforms, achieving notably lower scores across the board, while MSOPS shows a sharp decline in performance under configurations, C3 and C4.

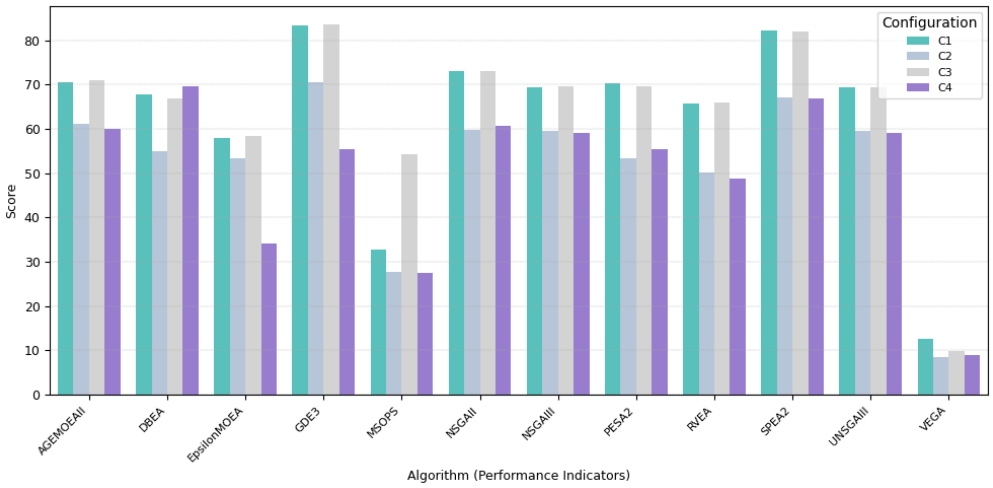


Figure 6.13: Performances Indicators - ZDT4

VEGA outperforms in landscape features scores in C1 and C2 but experiences a significant drop in C3 and C4. GDE3 and RVEA achieve the highest scores in C4, while the others maintain comparable scores across C2, C3, and C4 (see Figure 6.14).

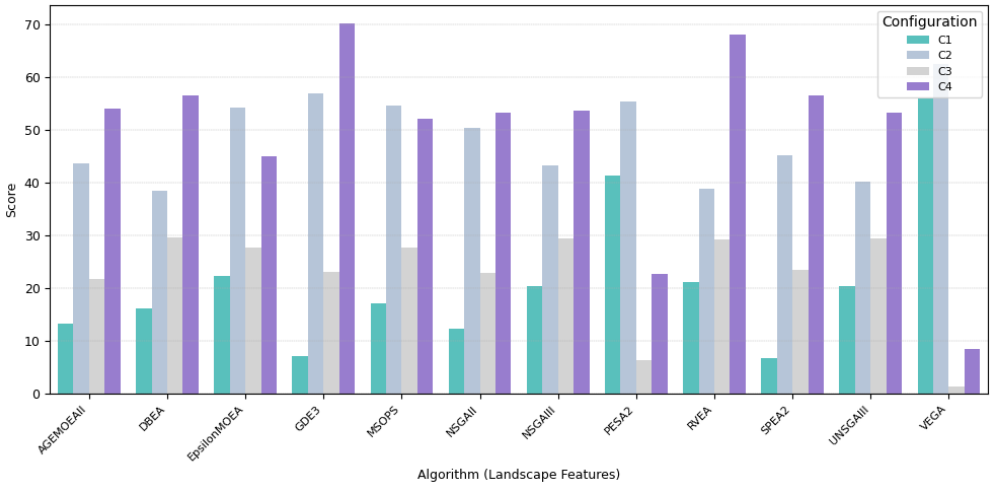


Figure 6.14: Landscape Features - ZDT4
land

Discussion The strong performance of PESA2 and RVEA in Configuration C1 suggests that these algorithms are particularly well-suited to scenarios where convergence and execution efficiency are prioritized. In contrast, GDE3's struggles with memory usage and VEGA's poor overall score indicate that these algorithms may not be as efficient in the C1 configuration.

GDE3 and RVEA's dominance in Configuration C2 highlights their robustness in handling convergence and execution tasks. The consistent underperformance of MSOPS, particularly in convergence, indicates a mismatch between this algorithm's strengths and the C2 configuration's demands on convergence.

The success of NSGA-III in Configuration C3, particularly in convergence and unimodality, suggests that NSGA-III is well-suited for tasks that require quick and effective convergence. PESA2's struggle in C3 is due to its inability to effectively handle evolvability and unimodality as required in this configuration. VEGA's continued poor performance, especially in convergence and landscape features, reaffirms its limitations in more demanding configurations.

In Configuration C4, DBEA's outstanding performance across all metrics suggests that it is capable of handling diversity and convergence as well. SPEA2's slight weaknesses compared to GDE3 indicates that while it is still a strong performer, it has limitations in landscape features maintenance. The consistent underperformance of MSOPS and VEGA suggests that these algorithms are not suitable for more complex or resource-intensive tasks.

ZDT6 Problem

Analysis The heatmaps for the ZDT6 problem, depicted in Figure 6.15, show varying algorithm performances. In configuration C1, the top three performers are MSOPS, PESA2, and EpsilonMOEA, with scores of 42.54%, 37.22%, and 23.40%, respectively, with particularly high scores in execution metrics, convergence, and hypervolume. The heatmap reveals that while MSOPS and PESA2 perform well in some areas, they are not performing well in global, evolvabitlity and ruggedness metrics.

In Configuration C2, MSOPS, PESA2, and SPEA2 lead with scores of 62.62%, 58.55%, and 51.20%. NSGA-III and UNSGA-III emerge as strong contenders, particularly in landscape-related metrics such as diversity and multi-modality. The heatmap suggests that these algorithms are well-suited to handle the complexities of the ZDT6 landscape in the C2, though AGEMOEA-II and DBEA show more varied performance, possibly indicating a need for better tuning or adaptation to this specific landscape.

Configuration C3 sees DBEA, UNSGA-III, and GDE3 as the top performers, scoring 78.4%, 74.12%, and 74.10%, respectively. The performance of PESA2 continues to be inconsistent, indicating possible difficulties in maintaining performance in more complex or diverse landscapes. AGEMOEA-II also shows competitive performance but lags slightly behind in certain metrics like spacing and hypervolume.

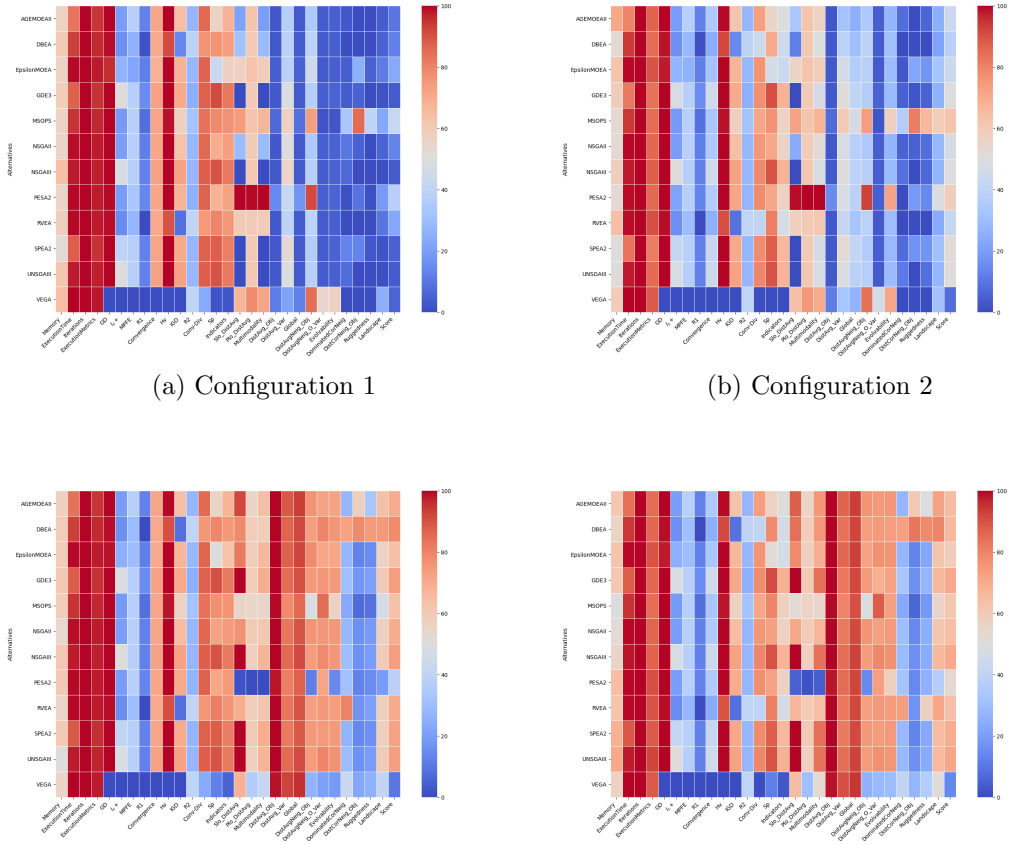
Finally, in C4, UNSGA-III shows high performance of 71.61%, particularly in convergence-related metrics and execution metrics. GDE3 remains a strong contender with 70.83%, although its performance is slightly lower than in previous configurations. The heatmap shows that VEGA struggles significantly in this configuration, further highlighting its limitations in handling the complexity of the ZDT6 in the configuration C4.

All the algorithms, except for MSOPS, PESA2, RVEA, and VEGA, show a considerable shift between C1 and C4 in terms of landscape features, as seen in Figure 6.16. SPEA2, for instance, sees its score increase from 1.62% in C1 to 65.21% in C4. This significant shift highlights the impact of configuration changes on the algorithms' ability to navigate the landscape features.

In terms of performance indicators, GDE3 and NSGA-III lead with over 87% in C1 and over 68% in C2. In C3 and C4, UNSGA-III emerges as the top performer with an impressive 82.35%.

In terms on the landscape features, DBEA (Configuration C4) leads with the highest Landscape score of 79.86%, followed closely by . AGEMOEA-II (Configuration C4) which performs strongly with a score of 72.58%. In Configuration C3, DBEA also performs well at 74.95%. Among the C2 configuration, MSOPS stands out with a Landscape score of 59.36%, followed by PESA2 (Configuration C2) at 49.24%.

Discussion In Configuration C1, MSOPS, PESA2, and EpsilonMOEA emerge as the top performers. These algorithms excel particularly in execution metrics, convergence, and hypervolume, which indicates their capability to handle the initial, perhaps simpler, configuration effectively. However, the results also reveals that these algorithms struggle with more complex landscape features, such as evolvability and ruggedness. This implies



(a) Configuration 1

(b) Configuration 2

(c) Configuration 3

(d) Configuration 4

Figure 6.15: ZDT6 Results

that they tend to provide more concentrated solutions in specific regions of the search space, lacking the ability to maintain diversity and effectively escape local optima.

The performance in Configuration C2 highlights a shift in the leading algorithms, with MSOPS, PESA2, and SPEA2 taking the forefront. The emergence of NSGA-III and UNSGA-III as strong contenders, particularly in landscape metrics like diversity and multi-modality, suggests that these algorithms are better suited for the complexities introduced in this configuration and for this problem.

Configuration C3 showcases the dominance of DBEA, UNSGA-III, and GDE3, all of which perform exceptionally well, scoring above 74%. This configuration appears to challenge PESA2 significantly, as evidenced by its inconsistent performance. The strong showing by DBEA and AGEMOEA-II in this setup, especially in ruggedness and spacing metrics, suggests that these algorithms can adapt well to increased multi-modality and diversity maintenance, though they may still lag behind in certain critical performance indicators.

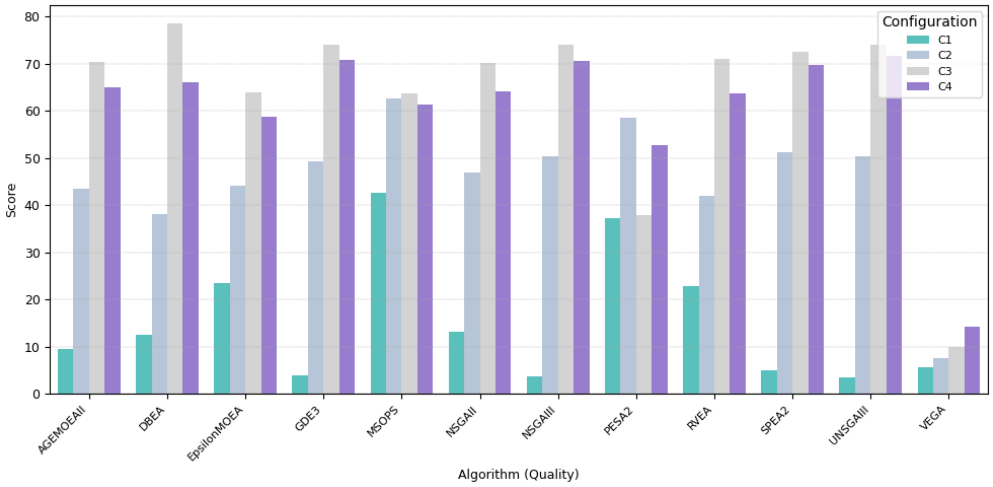


Figure 6.16: Algorithm Quality - ZDT6

Finally, in Configuration C4, the consistent performance of UNSGA-III and GDE3, despite the increased complexity, highlights their reliability across different configurations. Their ability to maintain high scores in both convergence and execution metrics across diverse landscapes underscores their versatility. On the other hand, VEGA's struggle in this configuration underlines its limitations.

The significant shifts in landscape feature scores between C1 and C4 further emphasize the impact of changing configurations on algorithm performance. The drastic improvement of algorithms like SPEA2, from a low score in C1 to a high score in C4, illustrates how certain algorithms may only realize their full potential under specific problem conditions, suggesting a need for a more nuanced understanding of when and how to deploy these algorithms effectively.

Given that the remaining problems are Large Scale Problems, it is judicious to eliminate VEGA at the beginning to ensure the focus remains on more competitive and capable algorithms. This decision is justified by VEGA's consistently poor performance across all ZDT problems, indicating its inability to handle complex landscapes and diverse optimization tasks effectively.

6.2.2 LSMOP Problems

LSMOP1 Problem

Analysis In Configuration C1, the scores of the MOEAs for LSMOP1 range from 4.6% to 39.81%. The top three performers are PESA2, AGEMOEAII, and GDE3 where PESA2 is shwoing 92.43% in the handling the global landscape features. All the algorithms consume varying amounts of memory, with scores ranging from 54% for UNSGA-III to 69% for DBEA. This suggests that DBEA is utilizing memory resources more effectively, than UNSGA-III which uses a reference point-based approach for diversity

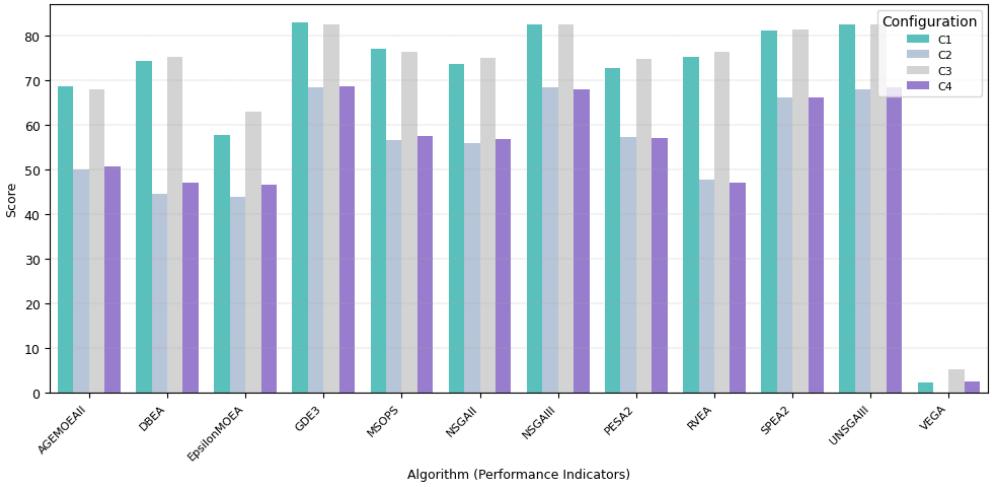


Figure 6.17: Performances Indicators - ZDT6

preservation and convergence. This approach typically requires maintaining a larger population of solutions to adequately cover the objective space, and involve complex calculations related to reference point management and the maintenance of multiple fronts.

While memory is generally not a bottleneck in modern computational environments, poor memory management can still lead to significant issues, such as heap overflow, which may result in the loss of progress made by the algorithm, particularly when dealing with large-scale problems.

In C2, AGEMOEA-II achieves a score of 53.13%. It shows consistent performance across configurations, achieving high scores in global and evolvability features in C1 and C3, but it falls short in convergence scores across several configurations.

DBEA excels in global features scores but struggles with handling complex landscapes.

EpsilonMOEA performs well in ruggedness, particularly in C1, with competitive Global and Evolvability scores in C2.

GDE3 consistently performs well in $MPFE$ across configurations, with particular robustness in C3. However, it has lower GD, landscape, and global features scores in some configurations, which affects its adaptability.

MSOPS excels in landscape features in C2 but has lower GD scores in C1 and C3, facing challenges in evolvability, which impacts long-term adaptability.

In C4, NSGA-III and UNSGA-III take top positions with a score of 56.75% and 55.28%. They have the highest landscape features scores (62%) and the convergence score with over 28%.

Figure 6.20 shows that DBEA, MSOPS, and RVEA perform poorly across all configurations compared to the other tested MOEAs where UNSGA-III shows a significant shift from C1 to C4.

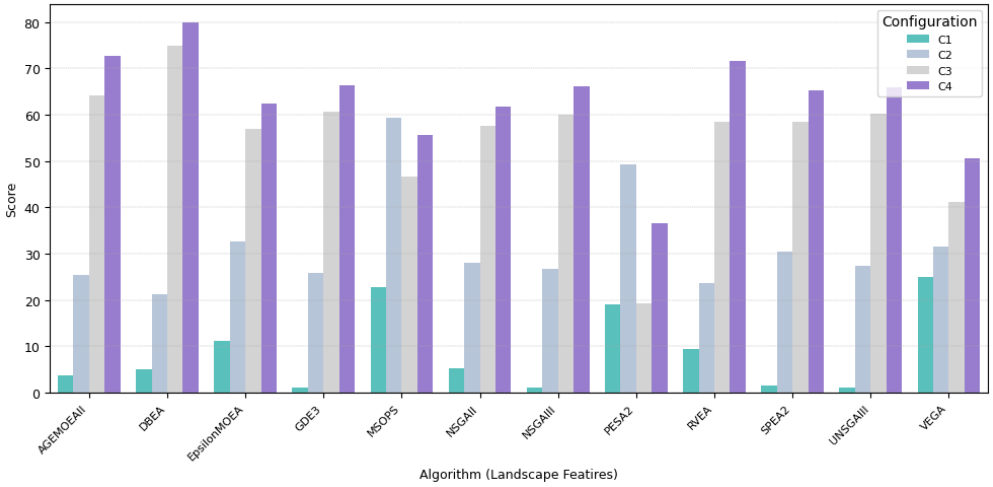


Figure 6.18: Landscape Features - ZDT6

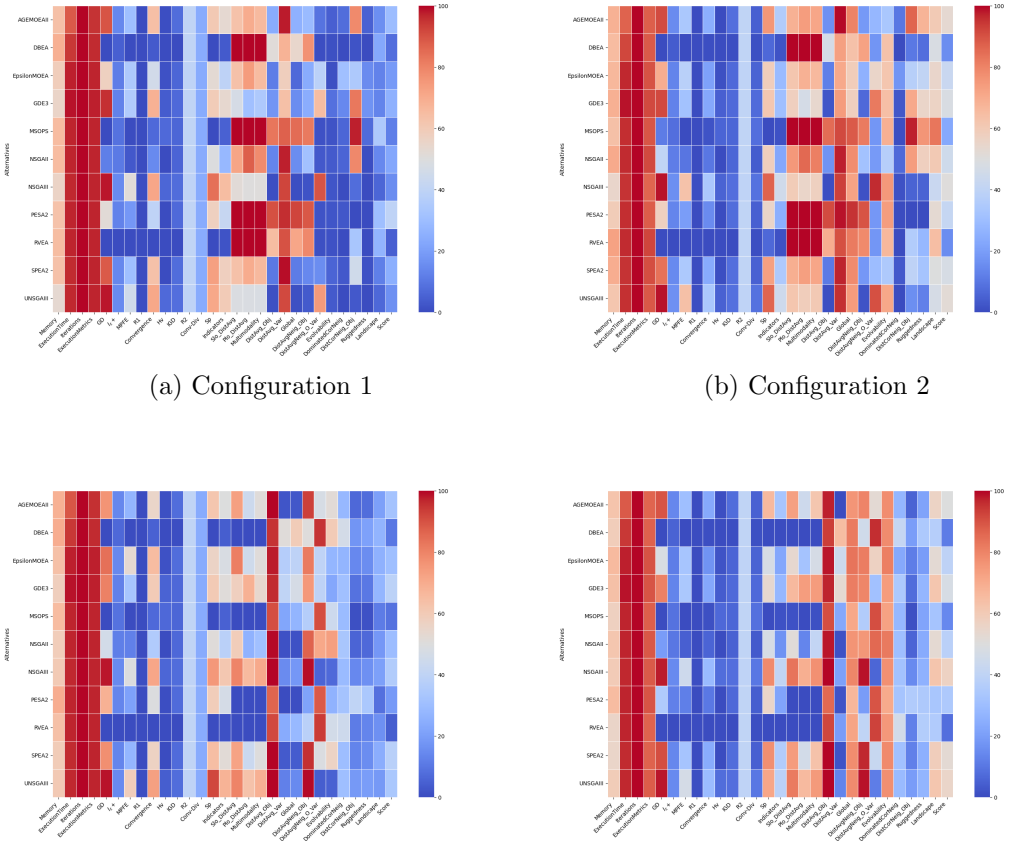
The algorithms DBEA, MSOPS, and RVEA fall significantly short in performance indicators, struggling to achieve competitive results as depicted in Figure 6.21. In contrast, UNSGA-III and NSGA-III perform very well across all four configurations, demonstrating strong and consistent performance in various scenarios.

MSOPS stands out significantly in C2, excelling in handling complex landscapes (se eFigure 6.22. NSGA-III and SPEA2 also perform well across multiple configurations, demonstrating adaptability to different problem landscapes. AGEMOEA-II and EpsilonMOEA show moderate performance, with some variability depending on the configuration.

Discussion The results of AS-LSP-MOEA clearly demonstrate that different MOEAs are suited to the LSMOP1 problem on different configurations, depending on the specific landscape features and performance indicators emphasized by each configuration.

AGEMOEA-II [104] excels in large-scale multi-objective optimization problems due to its adaptive mechanisms that dynamically adjust search strategies based on the problem landscape, efficient diversity maintenance to prevent premature convergence, and scalable approaches that handle a large number of decision variables. Its use of efficient fitness evaluation methods, including surrogate models, and strategic problem decomposition further enhance its ability to navigate vast search spaces quickly and robustly.

MSOPS While MSOPS is not designed for large-scale problems, its Pareto sampling helps in large-scale problems like LSMOP1, where the search space is vast, an effective sampling method can help in identifying and maintaining a diverse set of high-quality solutions.



(a) Configuration 1

(b) Configuration 2

(c) Configuration 3

(d) Configuration 4

Figure 6.19: LSMOP1 Results

LSMOP2 Problem

Analysis In the configuration C1 (see Figure 6.23a, GDE3 and DBEA exhibit the lowest values in terms of memory usage, indicating higher computational costs, while MSOPS and UNSGA-III show lower values, suggesting better efficiency. NSGAIII and UNSGA-III demonstrate superior performance in convergence, while DBEA and RVEA have higher values in overall performance indicators. However, in terms of landscape features, algorithms like PESA2 and MSOPS perform particularly well.

The top performers in C1 are PESA2 and EpsilonMOEA with overall scores of 52.04% and 45.35%, respectively.

In C2, as depicted in Figure 6.23b, AGEMOEAII and DBEA demonstrate higher computational costs in execution metrics. However, NSGAIII and UNSGA-III continue showing high scores in convergence, performance indicators and the overall scores in general. The heatmap also highlights the variability in the performance of the algorithms across different scenarios, emphasizing that no single algorithm consistently outperforms

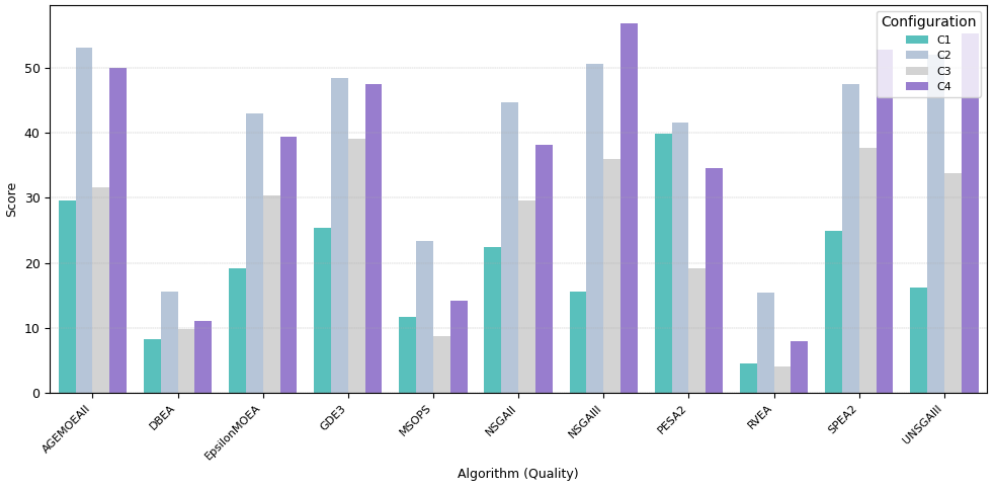


Figure 6.20: Algorithm Quality - LSMOP1

the others. The top three performaers are: NSGA-III (55.34%), GDE3 (53.77%), and UNSGA-III (53.07%).

In configuration C3, DBEA and SPEA2 lead with strong performance metrics. At the same time, NSGA-II emerges as a good performer in landscape features, highlighting its adaptability in fine-tuning and handling uni-modality. Moreover, NSGA-II surpasses AGEMOEA-II, which scored 45.75%, in the overall ranking by achieving a score of 48.09%, positioning itself just behind SPEA2, which has a score of 51.52%.

In C4, AGEMOEA-II rejoins the ranks of the top performers, securing a position just after SPEA2 with overall scores of 46.08% and 51.39%, respectively. DBEA and SPEA2 demonstrate high scores in performance indicators, while RVEA shows strong performance in the Spacing metric. Overall, the C4 heatmap reveals a similar pattern to the previous ones, where certain algorithms consistently perform well across various metrics, while others excel in specific areas but struggle in others. For instance, GDE3 scores 0% in the evolvability metric despite performing well in the convergence metric with a score of 70.36%.

The highest score is achieved by NSGA-III in C2 (55.34%), marking a significant shift from its performance in C1 (15.86%). On the other hand, NSGA-II shows consistent scores across the configurations, indicating its adaptability to different configurations, but without a significant peak in performance in any particular configuration.

Interestingly, LSMOP2 is the first problem among those tested to achieve high scores not in C4 but in C2, mainly, and C3, sometimes. This indicates that LSMOP2 performs well across a range of configurations, particularly those that emphasize a balance between diversity and are less restrictive.

The performance indicators for each algorithm remain relatively consistent across C1, C3, and C4, with similar scores observed in these configurations. However, these indicators experience a significant drop in C2. For example, SPEA2 scores 65.92%,

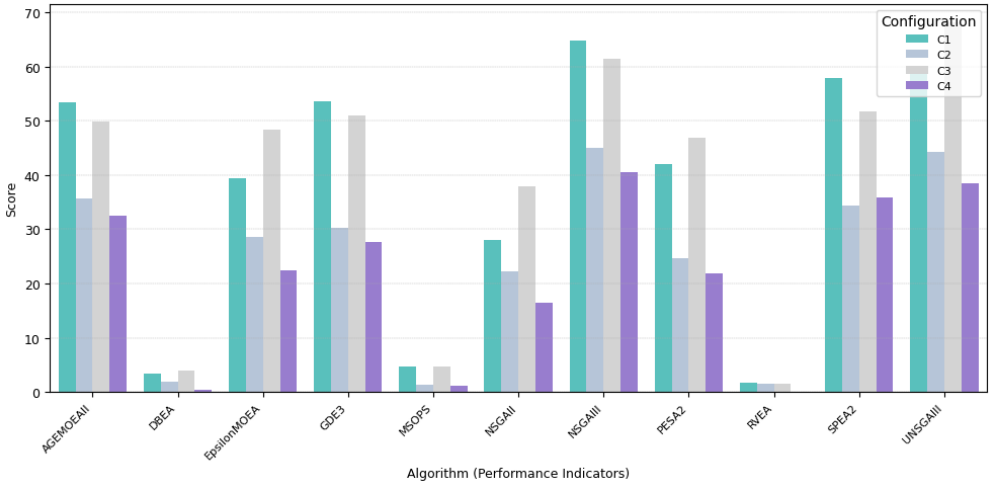


Figure 6.21: Performances Indicators - LSMOP1

65.51%, and 66.04% in C1, C3, and C4, respectively, but drops to 37.57% in C2.

DBEA stands out as a top performer in performance indicators for the three configurations C1, C3 and C4 with scores of 68.94%, 71.61%, and 70.62%, respectively . RVEA and SPEA also perform well, though there is some variability depending on the configuration. PESA2 and MSOPS show lower scores across the four configurations.

PESA2 dominates in this category, showing superior performance across different landscape features in C1 and C2, configuration preferring diversity among convergence.

NSGA-III and DBEA also perform well in LSMOP2, indicating their avility in managing complex landscapes. These algorithms show their effective exploration of challenging problem spaces, which involve a mix of simple and complex regions, as seen in LSMOP2. EpsilonMOEA and MSOPS, on the other hand, demonstrate good performance in handling multi-modality, especially in the C2 configuration. This indicates that these algorithms are adept at maintaining a diverse set of solutions, which is crucial when dealing with problems like LSMOP2, where multiple optimal solutions exist across different regions of the search space. Their ability to navigate through and effectively optimize such complex landscapes makes them particularly well-suited for problems that require a nuanced approach to multi-modality, as emphasized in C2.

Discussion LSMOP2 in C1 has PESA2 and EpsilonMOEA as the top performers. PESA2 handles diversity in LSMOP2 through a combination of grid-based selection, which ensures that solutions are selected from various regions of the Pareto front, and a prioritized replay mechanism that periodically reintroduces top-performing individuals from memory. On the other hand, EpsilonMOEA is able to maintain diversity, by adapting its epsilon parameter, and preserving niches. By using epsilon-dominance, the algorithm can effectively explore and exploit a search space having pultiple local and global optima such in LSMOP2.

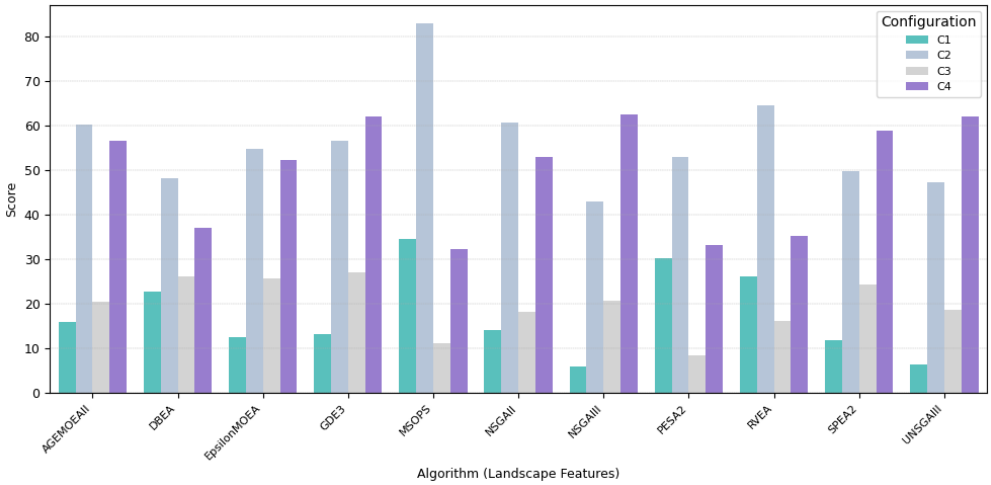


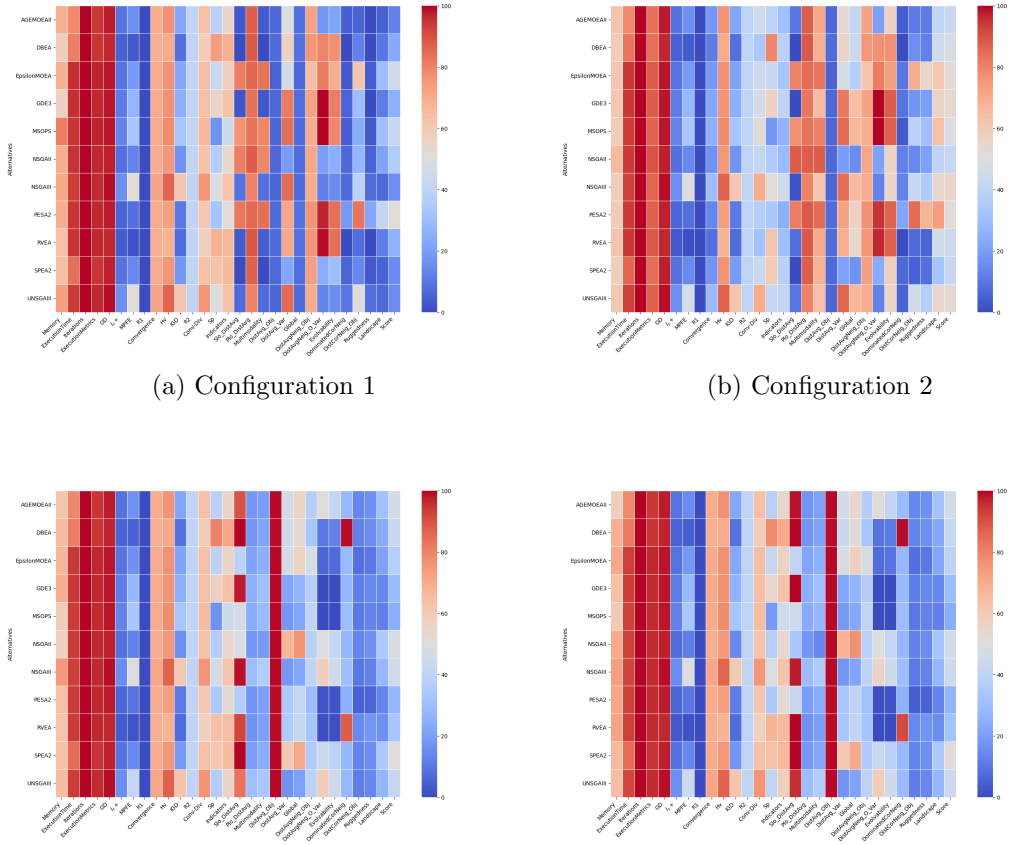
Figure 6.22: Landscape Features - LSMOP1

Configuration C2 highlights a notable shift in performance dynamics, where NSGA-III, GDE3, and UNSGA-III demonstrate strong results in handling LSMOP2. NSGA-III employs a reference point-based selection process to decompose the problem into sub-problems, ensuring a well-distributed population across the Pareto front, while also preserving diversity through niche management. UNSGA-III enhances this approach by dynamically adapting reference points and using a utility-based selection mechanism to efficiently maintain diversity. Meanwhile, GDE3 extends traditional Differential Evolution by initiating with a diverse population and dynamically adjusting crossover and mutation rates and using a dominance-based sorting mechanism and crowding distance-based selection.

Configuration C3 shifts the focus towards fine-tuning and convergence. Here, DBEA and SPEA2 lead in performance metrics, while NSGA-II shows strong adaptability in handling landscape features, particularly uni-modality through its effective combination of crowded tournament selection, fast non-dominated sorting, crowding distance computation, and elite preservation. These mechanisms allow NSGA-II to converge to the global optimum in simpler uni-modal landscapes.

Configuration C4 brings AGEMOEA-II back into the spotlight, securing a position just behind SPEA2. This configuration, which employs $D - +$ aggregators for all landscape sub-groups, appears to favor algorithms that can efficiently balance performance indicators and landscape features.

Overall, LSMOP2's performance across these configurations reveals that while certain algorithms excel in specific areas, others demonstrate broader adaptability across different configurations. This underscores the importance of considering multiple configurations and metrics when evaluating algorithm performance, as different scenarios can dramatically shift the rankings and effectiveness of the algorithms tested.



(a) Configuration 1

(b) Configuration 2

(c) Configuration 3

(d) Configuration 4

Figure 6.23: LSMOP2 Results

6.2.3 ECD Problem

Analysis In Configuration C1, as depicted in Figure 6.27a, the heatmap shows that RVEA (47.46%) and DBEA (26.56%) are the top performers across several metrics, especially in terms of execution, convergence, and multi-modality. SPEA2 also performs well, especially on indicators like spacing metric with an overall score of 13.66%.

Figure 6.27b shows the performance of the algorithms in C2. RVEA continues to show strong performance with a score of 57.77%, but we also see NSGA-III and SPEA2 emerging as competitive algorithms, with scores of 45% and 44.26%, respectively. These algorithms perform particularly well on landscape metrics, highlighting their performance on landscape metrics.

In configuration C3, as shown in Figure 6.27c, GDE3 (74.58%), PESA2 (67.42%), and EpsilonMOEA (63.64%) show improved performance in landscape indicators, particularly on ruggedness features. Notably, NSGA-II also emerges as a strong performer in this configuration, while RVEA begins to show weaknesses in several areas, indicating

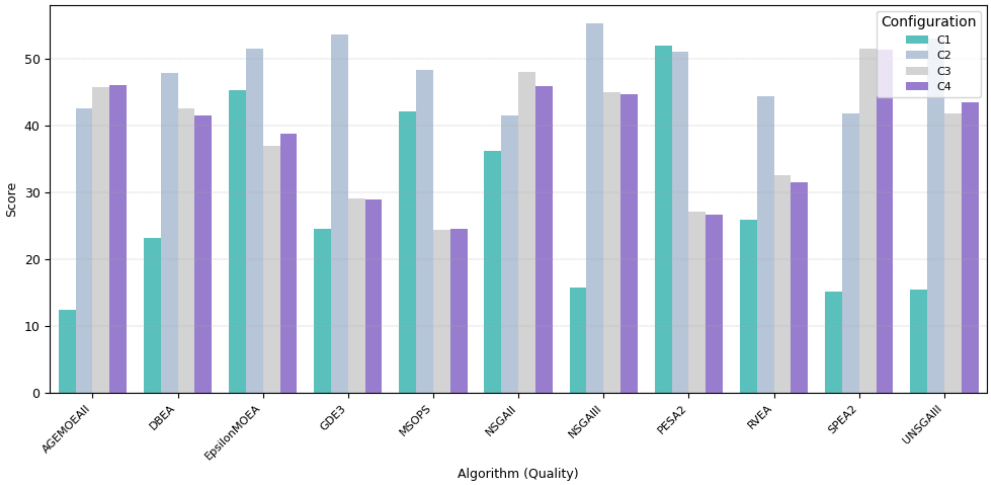


Figure 6.24: Algorithm Quality - LSMOP2

a decline in its ability to handle the increased complexity of the landscape features in C3.

In C4, we observe a strong performance from GDE3 (75.87%) , and EpsilonMOEA (68.26%), with each of them scoring high in performance indicators and landscape metrics. Other algorithms like MSOPS show inconsistent results, especially in terms of convergence metrics.

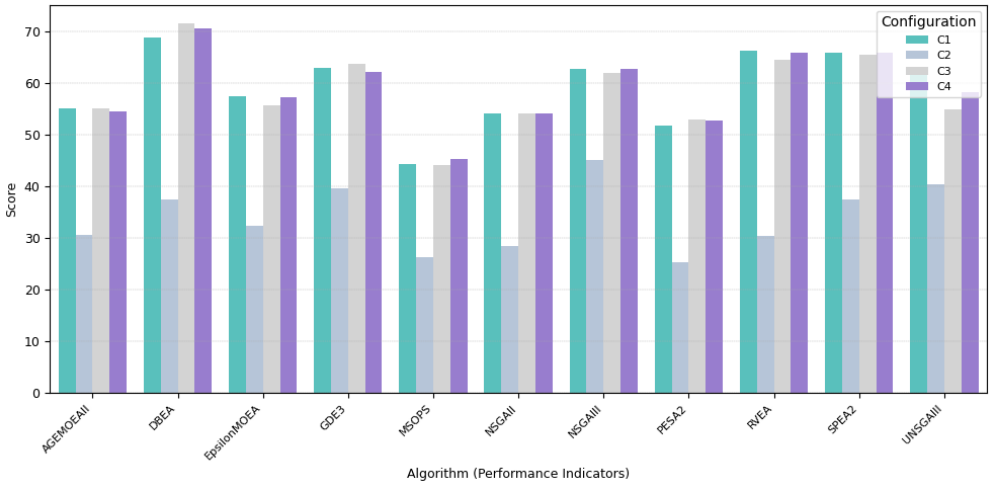
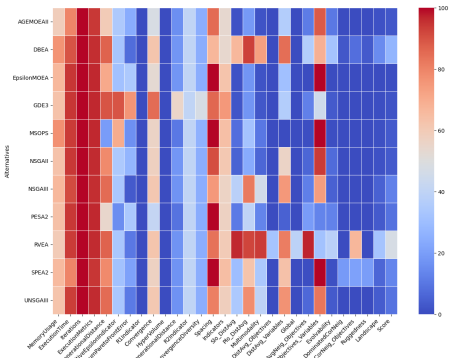
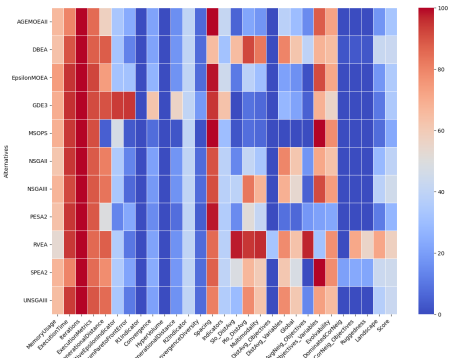


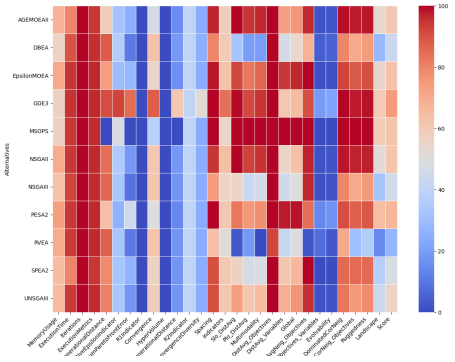
Figure 6.25: Performances Indicators - LSMOP2



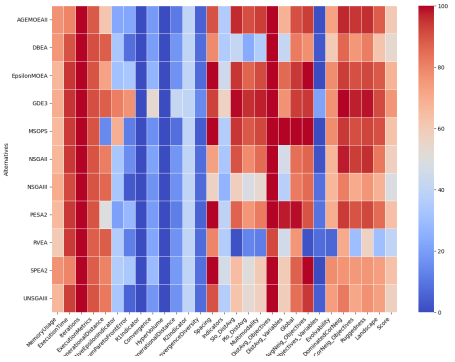
(a) Configuration 1



(b) Configuration 2



(c) Configuration 3



(d) Configuration 4

Figure 6.27: ECD Results

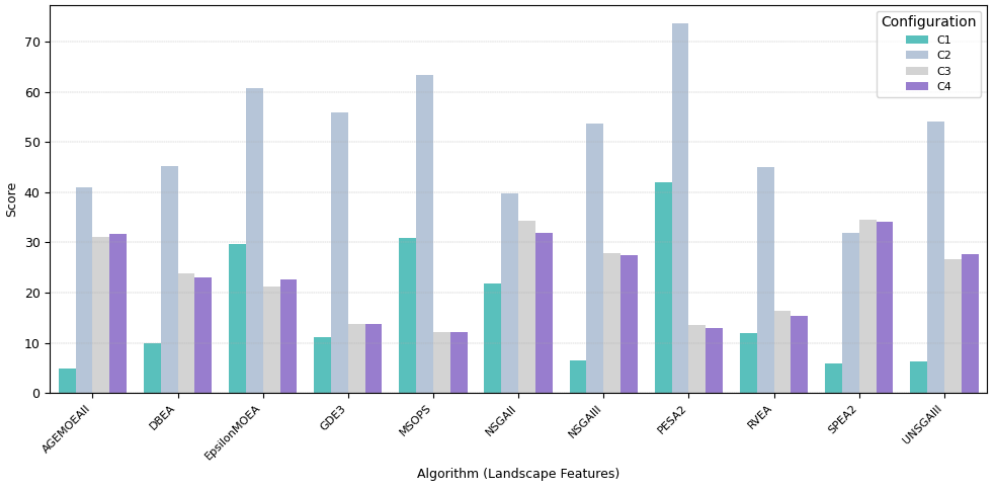


Figure 6.26: Landscape Features - LSMOP2

The scores in Figure 6.28 reflect the trends observed in the overall performance, with GDE3 and EpsilonMOEA displaying drastic shifts in their scores from C1 to C4. Conversely, NSGA-III exhibits consistent and harmonious scores across C2, C3, and C4, indicating its steady performance across these configurations.

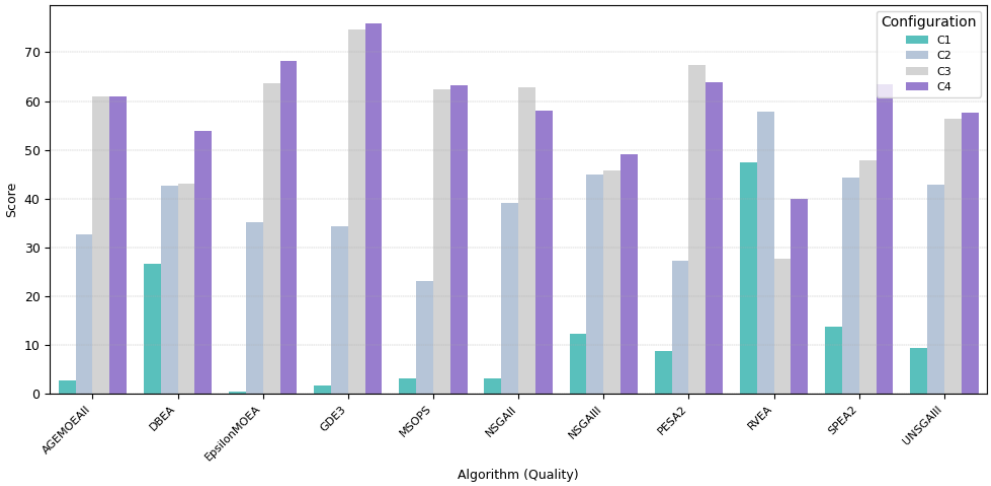


Figure 6.28: Algorithm Quality - ECD

GDE3 consistently ranks high across all configurations in performance indicators scores, see Figure 6.29, with notable peaks in C1 and C3. MSOPS and SPEA2 also show strong performance, while algorithms like AGEMOEA-II and DBEA have more varied performance.

There is a clear distinction in landscape feature scores across the configurations,

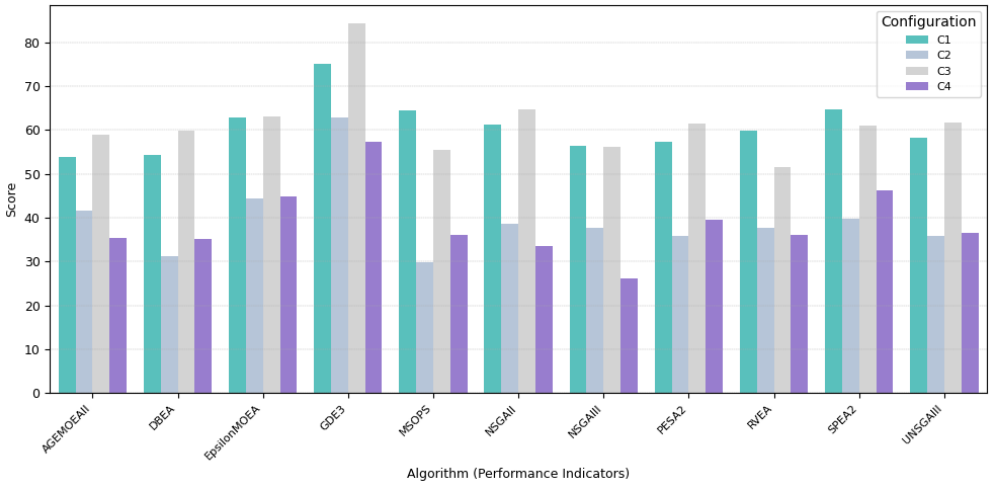


Figure 6.29: Performances Indicators - ECD

as depicted in Figure 6.30. All the algorithms, except RVEA, struggled to handle the restrictive diversity requirements imposed by C1. RVEA, however, shows comparable performance in both the diversity requirement configuration (C1) and the convergence requirement configuration (C4), with scores of 32.09% and 30.34%, respectively.

GDE3, MSOPS, EpsilonMOEA, NSGA-II, PESA2, and AGEMOEA-II demonstrate an upward trend in performance across landscape features as the configuration progresses.

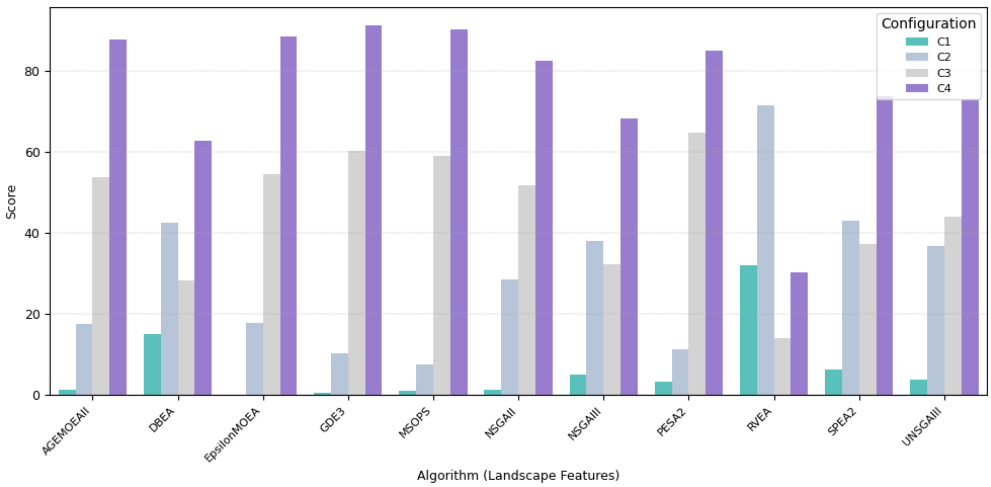


Figure 6.30: Landscape Features - ECD

Discussion The analysis of the ECD problem across different configurations reveals the varying strengths and weaknesses of the algorithms under different scenarios. In Configuration C1, RVEA and DBEA emerge as the top performers, suggesting that they are well-suited for problems requiring efficient execution and stable convergence, alongside handling diversified trade-offs. SPEA2 also shows a solid performance, especially in spacing metrics, indicating its capability in maintaining a well-distributed set of solutions.

As we move to Configuration C2, the landscape shifts slightly with NSGA-III and SPEA2 joining RVEA as top contenders. Their strong performance in landscape metrics suggests that these algorithms are better equipped to handle the diversity and complexity of the problem's landscape under this configuration. The emergence of NSGA-III highlight the importance of its reference point-based approach for diversity preservation.

In Configuration C3, the landscape becomes more challenging, with ruggedness features playing a significant role. Here, GDE3, PESA2, and EpsilonMOEA demonstrate improved performance, particularly in navigating the ruggedness of the landscape. NSGA-II also proves to be a strong competitor in this configuration, indicating its resilience in handling increased number of local optima. Conversely, RVEA's performance begins to decline, suggesting that it struggles with the maintain of a gradual refinement of the solutions introduced in C3.

Finally, in Configuration C4, GDE3 and EpsilonMOEA stand out with strong performances, scoring high in both performance indicators and landscape metrics. GDE3's consistent high ranking across all configurations, especially in C1 and C3, underscores its balancing techniques of ecloration and exploitation. EpsilonMOEA, while showing a significant shift in performance from C1 to C4, proves its capability in convergnce and diversity. On the other hand, MSOPS exhibits inconsistency, particularly in convergence metrics, highlighting potential limitations in its simple across different problem configurations. Overall, the scores depicted in Figure 6.28 align with these trends, with GDE3 and EpsilonMOEA showing significant shifts in performance across configurations, while NSGA-III maintains a steady and harmonious performance, particularly in C2, C3, and C4. The performance indicators in Figure 6.29 further reinforce GDE3's dominance, particularly in C1 and C3, while also acknowledging the strong performance of MSOPS and SPEA2. The varied performance of algorithms like AGEMOEA-II and DBEA across different configurations suggests that while they may excel in specific scenarios, they may require further tuning or adaptation to maintain consistency across diverse problem landscapes.

6.3 Synthesis

6.3.1 Consistency of the Results:

In deterministic scenarios, where the processes and conditions are fully controlled and repeatable, running the same evaluation method like LSP on the same problem with

identical criteria and configuration should consistently yield the exact same results. This consistency arises because the inputs remain constant, ensuring that the output is identical with every evaluation.

However, in the context of MOEAs, which are inherently stochastic, the situation is quite different. Stochastic algorithms incorporate randomness in their processes, and due to this randomness, each run of an algorithm can generate different solutions and yield varying performance metrics. Consequently, when these varying results are fed into the LSP module for evaluation, the scores produced by LSP may differ between runs—an expected outcome, given that the input data has changed.

The Consistency Table 6.2 is designed to quantify the impact of this stochastic nature on the final evaluation scores produced by AS-LSP-MOEA. We measure the Coefficient of Variation (CV) of the overall scores across different configurations (C1, C2, C3, C4) for MOEAs on the tested problems. A lower CV indicates more consistent or stable data, whereas a higher CV signals less consistency. We consider a CV greater than 20% ($CV > 20\%$) as a penalizing threshold. This means that any algorithm or configuration yielding a CV above this threshold is considered to have excessive variability, which could indicate instability in the results produced by that particular setup.

Table 6.2: Consistency of AS-LSP-MOEA Results

Problem	Config.	AGEMOEA-II	DBEA	EpsilonMOEA	GDE3	MSOPS	NSGA-II	NSGA-III	PESA2	RVEA	SPEA2	UNSGA-III	VEGA
ZDT1	C1	56.6	57.18	10.59	0.17	0.41	37.79	0.50	5.67	4.24	19.58	0.30	10.62
	C2	2.78	2.38	7.78	4.97	6.70	6.75	4.45	5.30	1.45	7.77	4.02	8.52
	C3	8.96	1.12	1.85	0.84	0.23	15.70	0.22	3.23	7.67	1.57	0.99	7.29
	C4	3.85	4.08	1.40	1.84	2.03	2.00	1.75	2.24	1.51	1.65	2.01	3.57
ZDT2	C1	78.63	75.52	9.59	31.44	37.68	55.57	31.50	10.59	2.28	45.99	28.02	19.08
	C2	7.30	1.80	7.21	2.91	3.47	8.07	2.85	8.94	4.32	6.16	2.23	17.54
	C3	1.37	15.10	2.76	0.65	2.04	13.98	0.06	3.34	3.77	1.37	0.75	7.01
	C4	4.00	5.82	2.35	1.25	2.40	4.75	1.46	2.43	2.33	1.33	2.19	20.59
ZDT4	C1	32.20	33.35	13.36	21.77	49.00	46.91	0.05	7.85	2.97	20.60	0.02	99.49
	C2	4.20	1.62	16.98	3.99	10.59	5.03	2.59	7.67	2.67	5.74	3.34	68.44
	C3	3.91	17.53	16.80	5.38	3.57	6.09	6.70	5.60	3.51	8.85	5.96	6.32
	C4	3.47	2.65	35.58	2.51	25.71	3.66	0.75	4.30	4.56	1.02	1.85	92.21
ZDT6	C1	102.22	79.25	20.96	11.32	4.52	74.31	9.64	8.55	4.69	60.44	8.66	68.73
	C2	3.91	17.53	16.80	5.38	3.57	6.09	6.70	5.60	3.51	8.85	5.96	6.32
	C3	1.89	3.09	15.86	1.00	2.00	2.43	1.66	7.86	1.61	0.68	0.75	119.17
	C4	3.76	2.89	18.53	2.24	2.97	3.04	1.53	4.29	2.24	1.65	1.11	70.93
LSMOP1	C1	23.82	24.15	37.99	32.51	2.63	11.73	12.39	16.74	19.80	24.19	7.98	-
	C2	13.55	60.90	49.71	24.03	9.15	19.48	10.94	32.44	66.67	23.15	20.40	-
	C3	21.16	10.16	40.99	29.94	6.02	30.04	27.29	10.72	13.23	7.50	39.24	-
	C4	6.44	13.68	43.44	23.79	2.98	16.80	3.40	16.13	7.97	5.82	7.61	-
LSMOP2	C1	30.65	3.53	8.24	16.57	6.63	26.31	43.29	8.65	19.93	26.99	48.20	-
	C2	5.78	5.92	4.68	2.76	4.71	5.60	3.97	2.78	4.25	5.31	13.73	-
	C3	7.04	5.14	8.96	1.70	4.17	5.26	2.16	16.07	8.95	2.57	11.84	-
	C4	3.75	7.64	4.65	3.26	4.66	11.13	4.93	8.11	5.17	2.02	6.56	-
ECD	**C1**	20.80	48.00	31.60	16.90	13.20	15.50	34.00	75.00	14.00	99.00	50.00	-
	C2	29.00	23.00	25.00	18.00	25.00	22.00	13.00	33.00	11.00	31.00	21.00	-
	C3	7.00	32.00	3.00	7.00	6.00	11.00	34.00	7.00	23.00	46.00	12.00	-
	C4	20.00	20.00	5.00	10.00	6.00	26.00	28.00	8.00	22.00	13.00	18.00	-

Configurations C2 and C3 consistently show lower CV values for most algorithms. This suggests that these configurations are generally more stable and provide more reliable evaluations across different runs. C2, which focuses on diversity, and C3, which emphasizes fast convergence, seem to mitigate the variability introduced by the stochastic nature of the algorithms.

Additionally, configuration C1 consistently exhibits higher CV values across many algorithms and problems. C1 is designed to emphasize diversity, which requires algorithms to explore more of the solution space. The restrictive nature of the aggregators used in C1 amplifies the impact of small differences in performance metrics.

The table reveals that the algorithm VEGA exhibits highly variable scores across different runs, highlighting its sensitivity to the stochastic elements inherent in its operation. This variability suggests that VEGA's performance can fluctuate significantly during the optimization process. As a result, VEGA's performance is highly unpredictable, making it less reliable in consistently delivering optimal solutions compared to other algorithms with more stable behavior.

In contrast, some algorithms exhibit much less variation, indicating they are more stable and produce more consistent results despite the inherent stochastic processes. For instance, NSGA-II and NSGA-III generally demonstrate more consistent performance across different configurations. NSGA-II, in particular, maintains relatively low Coefficient of Variation (CV) values across most configurations for the ZDT problems, reflecting its robustness and reliability. NSGA-III also shows consistent performance; however, it tends to exhibit higher variability in C1 for certain problems, such as LSMOP2.

On the light of these analysis, we discard the Configuration C1 for the test knwoing that the instability of the results is high for all the studied probelms and for almost all the algorithms. We also discard the algorithm VEGA from the portfolio to present to the future decision-makers.

The results of the ECD reveal the highest instability in configurations C1 and C2, while configurations, C3 and C4 demonstrate more stable and consistent outcomes, with significantly lower CV values. Configuration C3, in particular, stands out as the most reliable, providing the lowest variability and thus the most consistent performance across runs.

To sort the configurations for each problem according to the consistency values, we use the Weighted Sum Model:

$$\text{Weighted Sum} = 0.2 \times v + 0.3 \times v_{10} + 0.5 \times v_{20}$$

where:

- v is the number of occurrences where $CV < 10$.
- $v10$ is the number of occurrences where $10 \leq CV < 20$.
- $v20$ is the number of occurrences where $CV \geq 20$.

Lower weighted sum values indicate better performance and more consistent results.

We eliminate C1 for all problems since it consistently represents the least consistent configuration across all scenarios. We then rank the remaining configurations, but only those that have a weighted sum value lower than C1. If any configuration has a weighted sum equal to or greater than C1, that configuration is also eliminated. Using this approach, the configurations for each problem would be ranked as follows based on their consistency:

- ZDT1: Configurations ranked as (C2, C4) and then C3, indicating that C2 and C4 are the best configurations for this problem.
- ZDT2: Configurations ranked as C2, C3, and then C4.
- ZDT4: The ranking is C3, C4, and then C2.
- ZDT6: Configurations ranked as C2 then (C3, C4).
- LSMOP1: C4 is the most consistent configuration.
- LSMOP2: Configurations ranked as (C2, C4) and then C3, with C2 and C4 being the best.
- ECD: Configurations C4 and C3 show the best consistency.

These rankings suggest that C2, C3, and C4 are generally the most consistent configurations across different problems, with C4 often providing the best performance for complex problems like LSMOPs and ECD.

In the section 6.3.2, we analyze the most consistent results of AS-LSP-MOEA. To enhance the reliability of our findings, we eliminated the configuration C1 and the VEGA algorithm due to their high inconsistency.

6.3.2 Analysis

The results from our tests using the AS-LSP-MOEA on the ZDT and LSMOP problem families, as well as the Evolutionary Circuit Design case study, provide insights into the performance of various MOEAs across different configurations and problem complexities.

For the relatively simple ZDT1 and ZDT2 problems, GDE3 emerges as the most suitable algorithm in the context of continuous optimization problems. NSGA-II also proves to be a strong choice, showing comparable performance. However, when considering problem landscape features, GDE3 outperforms NSGA-II, particularly in terms of evolvability and ruggedness measures, as well as in convergence metrics.

GDE3's differential evolution strategy provides a significant advantage in exploring and exploiting continuous search spaces. Its adaptability to complex landscapes makes it more robust in finding well-distributed Pareto-optimal solutions, particularly with many continuous variables. While NSGA-II is effective and widely used, it can struggle with maintaining diversity in complex search spaces and may converge more slowly than GDE3, especially where GDE3's mutation strategy allows for a faster, more effective

Table 6.3: Best Algorithms per Problem

	Var.	Features	**C2**	**C3**	**C4**
ZDT1	30	Convex pf. Simple front For testing convergence	GDE3	GDE3	GDE3
ZDT2	30	Non-convex pf. Challenging non-linearities	GDE3	GDE3	GDE3
ZDT4	10	Convex pf. Highly multi-modal, 219 local optima Challenges exploration	GDE3	NSGA-III	DBEA
ZDT6	10	Non-convex pf. Non-uniforally distributed pf. Multi-modal Challenging exploration	MSOPS	DBEA	UNSGA-III
LSMOP1	1000	Unimodal pf. Linear Simple landscape	–	–	NSGA-III
LSMOP2	2000	Pf. of Mixed modality, Partially separable Increased complexity.	NSGA-III	SPEA2	SPEA2
ECD	242	Non-convex pf. multi-modal Feasibility constraints	–	GDE3	GDE3

search. Studies have shown GDE3's superior performance over NSGA-II for the ZDT problem family [5] and in other types of problems as well [84, 10].

For ZDT4, in the most consistent configuration, C3, NSGA-III emerges as the top performer. A study by Seada et al. [118] shows that NSGA-III outperforms NSGA-II in ZDT4, particularly at smaller population sizes, which aligns with our case where the population size is 100. Another study [14] confirms that NSGA-III surpasses both NSGA-II and SPEA2 in the ZDT problem family, including ZDT4. NSGA-III addresses the limitations of NSGA-II, such as its difficulty in maintaining diversity among population members, by adaptively updating a set of well-spread reference points [14]. GDE3 also performs well on ZDT4 in C2, as demonstrated in the study conducted by Kukkonen et al. [82].

In the LSMOP problems, the increased complexity and scale posed significant challenges. The best performer for both LSMOP1 and LSMOP2 is NSGA-III. It has been shown that NSGA-III outperforms many large-scale multi-objective algorithms in LSMOPs in a study by Pang *et al.* [103]. Additionally, research by Zille *et al.* [142] highlights that NSGA-III surpasses NSGA-II, RVEA, and GDE3 in LSMOPs, further solidifying its effectiveness in tackling these complex optimization problems.

SPEA2 also performs well in LSMOP2. As noted by Bui and Nguyen [20], in Pareto-based algorithms such as SPEA2, convergence is emphasized first, with diversity as a secondary consideration. This characteristic makes SPEA2 the best algorithm in C3 and C4, configurations that emphasize convergence more than diversity. Additionally, research by Yang *et al.* [137] demonstrates that SPEA2 outperforms fifteen other multi-objective algorithms in LSMOPs, further validating its effectiveness in LSMOP2.

In the ECD case study, which involved a large-scale design space with multiple objectives, the versatility and scalability of the top-performing algorithms were thoroughly tested. For the ECD problem, GDE3 once again demonstrated strong performance across most metrics, particularly in complex and diverse landscapes and when dealing with a large number of continuous variables, as previously noted.

6.4 Conclusion

In this chapter, we applied the LSP method using the AS-LSP-MOEA system to evaluate and rank a diverse set of multi-objective evolutionary algorithms across various problem families, including the ZDT and LSMOP problems, as well as a complex case study on Evolutionary Circuit Design (ECD). AS-LSP-MOEA integrates performance metrics, landscape features, and configurability to provide a comprehensive assessment of algorithms, making it a useful resource for decision-makers. Through extensive testing, we identified key patterns in algorithm performance across different configurations, highlighting the strengths and weaknesses of each algorithm within the context of continuous optimization and large-scale problems.

AS-LSP-MOEA succeeds in handling both simple and complex optimization problems by using a multi-criteria approach, which captures subtle differences in algorithm performance. The tool's flexibility allows users to adjust criteria weightings based on specific needs, ensuring it can be tailored to various problem domains. By considering the stochastic nature of MOEAs and analyzing consistency through metrics like the Coefficient of Variation (CV), AS-LSP-MOEA ensures reliable results.

Chapter 7

Conclusions and Future Work

7.1 Summary of Key Findings

This research explored the evaluation, ranking, and selection of multi-objective evolutionary algorithms (MOEAs) using the AS-LSP-MOEA system across a range of benchmark problems, including the ZDT family, LSMOP problems, and a real-world case study on Evolutionary Circuit Design (ECD). The study revealed significant insights into how different MOEAs perform under varying problem complexities and landscape features.

1. GDE3's Superior Performance in Simpler Problems: GDE3 consistently outperformed other algorithms in simpler, continuous optimization problems like ZDT1 and ZDT2. Its strength lies in its effective balance between exploration and exploitation, allowing it to navigate the search space efficiently and converge on high-quality solutions. Although NSGA-II is a well-established algorithm, it struggled with diversity maintenance and exhibited slower convergence compared to GDE3.

2. NSGA-III's Excellence in Complex Landscapes: For more challenging problems like ZDT4 and the large-scale LSMOP family, NSGA-III emerged as the top performer. Its ability to maintain diversity across complex, multi-modal landscapes was crucial in these scenarios. NSGA-III's reference point-based selection process provided a robust framework for handling the non-convex, multi-modal, and high-dimensional nature of these problems.

3. SPEA2 and its Role in Convergence: SPEA2 demonstrated strong performance in scenarios where convergence was a priority, particularly in the LSMOP2 problem. Its strength lies in efficiently balancing convergence and diversity, making it well-suited for complex, large-scale optimization tasks where maintaining a diverse set of solutions is essential.

4. GDE3's Versatility in Real-World Applications: The ECD case study, which involved a highly complex, large-scale optimization problem, highlighted GDE3's

versatility. GDE3 delivered robust performance across various metrics and configurations, proving its adaptability to real-world challenges where problem landscapes are non-convex, multi-modal, and feature diverse feasibility constraints.

7.2 Adaptability to Landscape Features and Problem Complexity

One of the core strengths of the AS-LSP-MOEA system is its ability to tailor the algorithm selection process based on the specific landscape features of the problem at hand. The results consistently demonstrated that AS-LSP-MOEA could identify and prioritize algorithms that excel in the landscape features relevant to each configuration:

- Simple Problems (ZDT1, ZDT2): For relatively straightforward problems, AS-LSP-MOEA favored algorithms like GDE3 and PESA2, which are designed to converge quickly while effectively exploring the solution space. The simplicity of these problems allowed AS-LSP-MOEA to focus on algorithms that prioritize efficiency and speed without the need for overly complex mechanisms.
- Complex Problems (ZDT4, LSMOPs): As the complexity of the problems increased, particularly in cases like ZDT4 and LSMOP2, AS-LSP-MOEA adapted by selecting algorithms that maintain high performance across both convergence and diversity metrics. NSGA-III and SPEA2 were often chosen for their ability to handle the intricate balance between these two aspects, especially in multi-modal and high-dimensional landscapes.
- Real-World Application (ECD): In the ECD case study, which presented a large-scale, multi-objective optimization challenge, AS-LSP-MOEA's adaptability was again demonstrated. The system successfully identified GDE3 as the best performer, highlighting its capability to manage real-world complexities where both precision and scalability are critical.

7.3 Evaluation of AS-LSP-MOEA Configurations

The study also revealed important insights into the effectiveness of different AS-LSP-MOEA configurations:

1. Configuration C2: This configuration, which emphasized diversity, was particularly effective in problems like LSMOP1 and LSMOP2, where maintaining a wide range of high-quality solutions across the search space was critical. Algorithms like NSGA-III and SPEA2 thrived in this environment, demonstrating their ability to balance exploration with effective convergence.
2. Configuration C3: In scenarios where quick convergence was prioritized, as seen in the ZDT4 and ECD problems, Configuration C3 proved most effective. This

configuration allowed algorithms like GDE3 and DBEA to leverage their strengths in fine-tuning and fast convergence, resulting in high performance across a range of metrics.

3. Configuration C4: The robustness of Configuration C4 was particularly evident in complex problems like LSMOP2 and ECD. This configuration favored algorithms that could balance performance across both indicators and landscape features, making it ideal for scenarios with an emphasis on the convergence.

7.4 Challenges and Limitations

While the findings of this study are promising, several challenges and limitations were encountered:

- Limited Number of AS-LSP-MOEA Runs: Due to resource constraints, the number of AS-LSP-MOEA runs was limited to 10 per problem. Ideally, a larger number of runs would yield more statistically robust results, allowing for a deeper understanding of each algorithm's performance across different stochastic scenarios.
- Complexity of the ECD Problem: The ECD problem, with its large-scale nature and intricate design space, highlighted the limitations of current MOEAs. The complexity of the problem underscored the need for more advanced and scalable optimization techniques that can handle real-world challenges with greater efficiency.
- Execution Metrics as a Limitation: The execution metrics, which reflect the computational resources required by each algorithm, were identified as a potential limiting factor. These metrics could be reconsidered as the "cost" of achieving high-quality solutions, providing a more holistic view of algorithm efficiency.

7.5 Perspectives

Building on the findings of this research, several avenues for future work can be pursued:

Currently, we are using an Online Algorithm Selection approach that requires running the algorithms to collect criteria. However, it could be a valuable enhancement to incorporate machine learning techniques to predict performance indicators based on historical data, which would shift the approach to Offline Selection. This change would allow us to estimate an algorithm's performance without the need to execute it, thereby saving time and computational resources while still making informed selections.

We would also present the consistency analysis to the decision-maker to showcase the stability of the results, thereby enhancing the system's transparency. This feature could guide users in selecting algorithms that not only perform well but also deliver stable outcomes across multiple runs, ensuring more dependable decision-making in practice. Alternatively, consistency could be incorporated as a penalizing factor in the scoring

system, further refining the selection process by adjusting the scores of alternatives based on their stability.

Similarly, it could be very intersting to incorporate mandatory, desired, and optional criteria to enhance the evaluation. For example, convergence could be considered mandatory, diversity desired, and evolvability optional. To accommodate such modifications, it may be necessary to adopt aggregators other than GCD aggregators with Weighted Powered Mean. A comparison between these versions could provide deeper insights into the selection process and better flexibility in the prioritization of the criteria.

Furthermore, we could remove execution time and memory usage from the list of criteria and give the decision-maker the option to consider them as the "cost" of achieving a certain level of solution quality. This approach allows for a more balanced assessment, where an algorithm that delivers high-quality solutions might be less desirable if it incurs significant computational costs, while a slightly less effective algorithm could be preferred if it operates more efficiently.

Further testing could involve the use of normalized utility values and non-correlated metrics to explore how these adjustments impact the accuracy of the AS-LSP-MOEA results. The normalized utility values allow for the consideration of a wide range of problems without requiring specific knowledge of each when defining preferences; the decision-maker can simply define general trends, such as fast convergence, high diversity, or a balance of both. The use of non-correlated metrics helps reduce the redundancy of criteria, leading to a more accurate suitability table.

Another perspective involves enlarging the portfolio by broadening the categories of MOAs tested within the system, incorporating a wider variety of algorithmic strategies such as swarm intelligence techniques and decomposition-based methods. This approach would provide the decision-maker with a broader range of choices.

Finally, we would develop a graphical framework to facilitate the use of the AS-LSP-MOEA system, providing an intuitive and user-friendly interface that allows users to easily interact with and configure the system for various algorithm selections.

Bibliography

[1] S. M. Abdulrahman, P. Brazdil, W. M. N. W. Zainon, and A. Adamu, "Simplifying the algorithm selection using reduction of rankings of classification algorithms," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, 2019, pp. 140–148.

[2] H. Aguirre and K. Tanaka, "Working principles, behavior, and performance of MOEAs on MNK-landscapes," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1670–1690, 2007.

[3] R. Amara, M. B. Aissa, R. Kemcha, M. Bougherara, and N. Louam, "Geographical information system for air traffic optimization using genetic algorithm," *GeoInformatica*, vol. 27, no. 3, pp. 593-617, 2023.

[4] C. Ansótegui, J. Gabas, Y. Malitsky, and M. Sellmann, "MaxSAT by improved instance-specific algorithm configuration," *Artificial Intelligence*, vol. 235, pp. 26–39, 2016, Elsevier.

[5] L. M. Antonio and C. A. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 2758-2765. doi: 10.1109/CEC.2013.6557903.

[6] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition based evolutionary algorithm for many objective optimization with systematic sampling and adaptive epsilon control," in *Evolutionary Multi-Criterion Optimization: 7th International Conference, EMO 2013, Sheffield, UK, March 19-22, 2013. Proceedings*, vol. 7, Springer Berlin Heidelberg, 2013, pp. 413-427.

[7] C. Audet, J. Bigeon, D. Cartier, S. Le Digabel, and L. Salomon, "Performance indicators in multiobjective optimization," *European Journal of Operational Research*, vol. 292, no. 2, pp. 397–422, 2021. Elsevier.

[8] S. Barak and T. Mokfi, "Evaluation and selection of clustering methods using a hybrid group MCDM," *Expert Systems with Applications*, vol. 138, p. 112817, 2019.

[9] R. Bausys, G. Kazakeviciute-Januskeviciene, F. Cavallaro, and A. Usovaite, "Algorithm selection for edge detection in satellite images by neutrosophic WASPAS method," *Sustainability*, vol. 12, no. 2, p. 548, 2020.

[10] M. M. Bech, C. Noergaard, D. B. Roemer, and S. Kukkonen, "A global multi-objective optimization tool for design of mechatronic components using generalized differential evolution," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2016, pp. 475-481.

[11] T. Beielstein, J. Dienstuhl, C. Feist, and M. Pompl, "Circuit design using evolutionary algorithms," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, vol. 2, IEEE, 2002, pp. 1904–1909.

[12] H. P. Benson, "Existence of efficient solutions for vector maximization problems," *Journal of Optimization Theory and Applications*, vol. 26, pp. 569–580, 1978.

[13] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653-1669, 2007.

[14] R. H. Bhesdadiya, I. N. Trivedi, P. Jangir, N. Jangir, and A. Kumar, "An NSGA-III algorithm for solving multi-objective economic/environmental dispatch problem," *Cogent Engineering*, vol. 3, no. 1, p. 1269383, 2016.

[15] S. Bigaret, R. E. Hodgett, P. Meyer, T. Mironova, and A.-L. Olteanu, "Supporting the multi-criteria decision aiding process: R and the MCDA package," *EURO Journal on Decision Processes*, vol. 5, no. 1-4, pp. 169–194, 2017. Available: https://cran.r-project.org/web/packages/MCDA/index.html.

[16] B. Bischl, P. Kerschke, L. Kotthoff, M. Lindauer, Y. Malitsky, A. Fréchette, H. Hoos, et al., "Aslib: A benchmark library for algorithm selection," *Artificial Intelligence*, vol. 237, pp. 41-58, 2016.

[17] L. Blet, S. N. Ndojh, and C. Solnon, "Experimental comparison of BTD and intelligent backtracking: Towards an automatic per-instance algorithm selector," in *Principles and Practice of Constraint Programming: 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings 20*, pp. 190–206, 2014, Springer.

[18] M. Bougherara, R. Kemcha, N. Nedjah, D. Bennouar, and L. de Macedo Mourelle, "IP assignment optimization for an efficient NOC-based system using multi-objective differential evolution," in *International Conference on Metaheuristics and Nature Inspired Computing (META)*, 2018, pp. 435-444.

[19] J. P. Brans, "L'ingénierie de la décision: élaboration d'instruments d'aide à la décision. La méthode PROMETHEE," *Presses de l'Université Laval*, 1982.

[20] L. T. Bui and T. T. Nguyen, "A modified dual-population approach for solving multi-objective problems," in *2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES)*, IEEE, Nov. 2017, pp. 89-94.

[21] S. Chakraborty and E. K. Zavadskas, "Applications of WASPAS Method in Manufacturing Decision Making," *Informatica*, vol. 25, no. 1, pp. 1–20, 2014. doi: 10.15388/Informatica.2014.01.

[22] S. Chakraborty, P. Chatterjee, and P. Das, "Compromise Ranking of Alternatives from Distance to Ideal Solution (CRADIS) Method," in *Book Title (if applicable)*, 2023. doi: 10.1201/9781003377030-32.

[23] A. Charnes, W. W. Cooper, and E. Rhodes, "Measuring the efficiency of decision making units," *European Journal of Operational Research*, vol. 2, no. 6, pp. 429-444, 1978. doi: 10.1016/0377-2217(78)90138-8.

[24] S. Cheng, Y. Shi, and Q. Qin, "On the performance metrics of multiobjective optimization," in *International Conference in Swarm Intelligence*, Springer, 2012, pp. 504–512.

[25] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, 2016. doi: 10.1109/TEVC.2016.2519378.

[26] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 7, no. 12, pp. 4108–4121, 2017.

[27] C. A. Coello Coello and M. Salazar Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, pp. 1051–1056, IEEE, 2002.

[28] C. A. Coello Coello and N. Cruz Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, pp. 163–190, 2005.

[29] D. W. Corne, J. D. Knowles, and M. J. Oates, "PESA-II: Region-based selection in evolutionary multiobjective optimization," in *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pp. 283–290, Morgan Kaufmann Publishers Inc., 2001.

[30] I. Dagan, R. Vainshtein, G. Katz, and L. Rokach, "Automated Algorithm Selection Using Meta-Learning and Pre-trained Deep Convolution Neural Networks," *Information Fusion*, vol. 105, p. 102210, 2024, Elsevier.

[31] A. Das, T. Chaudhuri, S. S. Roy, S. Biswas, and B. Guha, "Selection of appropriate portfolio optimization strategy," *Theoretical and Applied Computational Intelligence*, vol. 1, no. 1, pp. 58-81, 2023.

[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002.

[33] K. Deb, M. Mohan, and S. Mishra, "A Fast Multi-Objective Evolutionary Algorithm Using Epsilon-Dominance and Its Application to Real-World Problems," *Evolutionary Computation*, vol. 10, no. 5, pp. 537-561, 2003.

[34] K. Deb and H. Jain, "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*, vol. 18, no. 4, pp. 577-601, 2013.

[35] "Definite / BOSDA," Available: https://spinlab.vu.nl/support/tools/definite-bosda/.

[36] T. de la Rosa, S. Jiménez, R. Fuentetaja, and D. Borrajo, "Scaling up heuristic planning with relational decision trees," *Journal of Artificial Intelligence Research*, vol. 40, no. 1, pp. 767–813, 2011.

[37] D. Ju-Long, "Control problems of grey systems," *Journal of Huazhong Institute of Technology*, Department of Automation, Huazhong Institute of Technology, Wuhan, China, 1981.

[38] K. Doerner, W. J. Gutjahr, R. F. Hartl, M. Karall, and M. Reimann, "Ant colony optimization and its application to the multiobjective shortest path problem," *Asia-Pacific Journal of Operational Research*, vol. 21, no. 02, pp. 225–247, 2004, World Scientific.

[39] "D-Sight Software for Decision-Making," Available: https://www.d-sight.com/.

[40] J. J. Dujmovic, "Mixed averaging by levels (MAL)–A system and computer evaluation method," in *Proceedings of the Informatica Conference, Bled, Yugoslavia*, 1973.

[41] J. J. Dujmovic, "Preferential Neural Networks," in *Neural Networks: Concepts, Applications, and Implementations, Volume II*, P. Antognetti and V. Milutinović, Eds., 1991, pp. 7-XX.

[42] J. J. Dujmović and H. Nagashima, "LSP method and its use for evaluation of Java IDEs," *International Journal of Approximate Reasoning*, vol. 41, no. 1, pp. 3–22, 2006.

[43] J. J. Dujmović and H. L. Larsen, "Generalized conjunction/disjunction," *International Journal of Approximate Reasoning*, vol. 46, no. 3, pp. 423–446, 2007.

[44] F. Ecer and D. Pamucar, "A novel LOPCOW-DOBI multi-criteria sustainability performance assessment methodology: An application in developing country banking sector," *Omega*, vol. 112, p. 102690, 2022. doi: 10.1016/j.omega.2022.102690.

[45] M. Ehrgott, "A discussion of scalarization techniques for multiple objective integer programming," *Annals of Operations Research*, vol. 147, no. 1, pp. 343–360, 2006.

[46] M. Fakhfakh, Y. Cooren, A. Sallem, M. Loulou, and P. Siarry, "Analog circuit design optimization through the particle swarm optimization technique," *Analog Integrated Circuits and Signal Processing*, vol. 63, pp. 71-82, 2010.

[47] J. G. Falcón-Cardona, M. T. M. Emmerich, and C. A. Coello, "On the construction of pareto-compliant combined indicators," *Evolutionary Computation*, vol. 30, no. 3, pp. 381–408, 2022, MIT Press.

[48] Z. Fan, B. Ghaddar, X. Wang, L. Xing, Y. Zhang, and Z.-H. Zhou, "Artificial Intelligence for Operations Research: Revolutionizing the Operations Research Process," *arXiv preprint* arXiv:2401.03244, 2024.

[49] A. Fialho, L. DaCosta, M. Schoenauer, and M. Sebag, "Analyzing Bandit-Based Adaptive Operator Selection Mechanisms," *Annals of Mathematics and Artificial Intelligence*, 2010.

[50] E. Filatovas, O. Kurasova, and K. Sindhya, "Synchronous R-NSGA-II: an extended preference-based evolutionary algorithm for multi-objective optimization," *Informatica*, vol. 26, no. 1, pp. 33–50, 2015.

[51] P. C. Fishburn, "Letter to the Editor—Additive Utilities with Incomplete Product Sets: Application to Priorities and Assignments," *Operations Research*, vol. 15, no. 3, pp. 537-542, 1967. doi: 10.1287/opre.15.3.537.

[52] J. Figueira, S. Greco, and M. Ehrgott, *Multiple criteria decision analysis: state of the art surveys*. Springer Science & Business Media, 2005.

[53] M. Gagliolo and J. Schmidhuber, "Learning dynamic algorithm portfolios," *Annals of Mathematics and Artificial Intelligence*, vol. 47, no. 3-4, pp. 295–328, 2006.

[54] M. Gagliolo and J. Schmidhuber, "Algorithm portfolio selection as a bandit problem with unbounded losses," *Annals of Mathematics and Artificial Intelligence*, vol. 61, pp. 49–86, 2011.

[55] A. Garbajosa, T. de la Rosa, and R. Fuentetaja, "Planning with ensembles of classifiers," in *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pp. 1007–1008, 2014.

[56] A. M. Geoffrion, J. S. Dyer, and A. Feinberg, "An interactive approach for multi-criterion optimization, with an application to the operation of an academic department," *Management Science*, vol. 19, no. 4, pp. 357–368, 1972.

[57] H. Guo, Y. Ma, Z. Ma, J. Chen, X. Zhang, Z. Cao, J. Zhang, and Y.-J. Gong, "Deep Reinforcement Learning for Dynamic Algorithm Selection: A Proof-of-Principle Study on Differential Evolution," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024, IEEE.

[58] D. Hadka, "MOEA Framework: A Free and Open Source Java Framework for Multiobjective Optimization (Version 4.4) [Computer software]," 2024. Available: https://github.com/MOEAFramework/MOEAFramework.

[59] S. H. Razavi Hajiagha, S. S. Hashemi, and E. K. Zavadskas, "A complex proportional assessment method for group decision making in an interval-valued intuitionistic fuzzy environment," *Technological and Economic Development of Economy*, vol. 19, no. 1, pp. 22–37, 2013. doi: 10.3846/20294913.2012.762953.

[60] M. P. Hansen and A. Jaszkiewicz, "Evaluating the quality of approximations to the non-dominated set," *Institute of Mathematical Modeling, Technical University of Denmark*, Denmark, 1998.

[61] K. Hatch, S. Dragićević, and J. Dujmović, "Logic scoring of preference and spatial multicriteria evaluation for urban residential land use analysis," in *Proceedings of the International Conference on Geographic Information Science*, Springer, 2014, pp. 64–80.

[62] D. Horn, K. Schork, and T. Wagner, *Multi-objective Selection of Algorithm Portfolios: Experimental Validation*, Technische Universitat Dortmund, 2016.

[63] R. Hrbacek, V. Mrazek, and Z. Vasicek, "Automatic design of approximate circuits by means of multi-objective evolutionary algorithms," in *2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS)*, IEEE, 2016, pp. 1-6.

[64] E. J. Hughes, "Multiple single objective Pareto sampling," in *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.*, vol. 4, IEEE, 2003, pp. 2678-2684.

[65] F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stützle, "ParamILS: An automatic algorithm configuration framework," *Journal of Artificial Intelligence Research*, vol. 36, pp. 267–306, 2009.

[66] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers*, vol. 5, pp. 507–523. Springer Berlin Heidelberg, 2011.

[67] C.-L. Hwang and A. S. M. Masud, *Multiple objective decision making, methods and applications: a state-of-the-art survey*, Springer-Verlag, 1979. ISBN 978-0-387-09111-2.

[68] C.-L. Hwang and K. Yoon, "Methods for multiple attribute decision making," in *Multiple attribute decision making: methods and applications a state-of-the-art survey*, Springer, 1981, pp. 58–191.

[69] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, T. Stützle, and M. Birattari, "The irace package: Iterated Racing for Automatic Algorithm Configuration," *Operations Research Perspectives*, 2016.

[70] A. D. B. Jesus, *Algorithm Selection for Multi-Objective Optimization*. PhD thesis, University of Coimbra; University of Lille, 2022.

[71] D. Jones and M. Tamiz, *Practical Goal Programming*. Springer, 2010.

[72] P. Kerschke, M. Preuss, S. Wessing, H. Trautmann, and C. Grimme, "Towards analyzing multi-modality of continuous multi-objective landscapes," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, pp. 1–8.

[73] P. Kerschke, H. H. Hoos, F. Neumann, and H. Trautmann, "Automated algorithm selection: Survey and perspectives," *Evolutionary Computation*, vol. 27, no. 1, pp. 3–45, 2019. MIT Press.

[74] E. B. Khalil, B. Dilkina, G. L. Nemhauser, S. Ahmed, and Y. Shao, "Learning to Run Heuristics in Tree Search," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 659–666, 2017.

[75] R. Kemcha, N. Nedjah, A. R. Maouche, and M. Bougherara, "Evolutionary design of approximate sequential circuits at RTL using particle swarm optimization," in *Computational Science and Its Applications–ICCSA 2019: 19th International Conference, Saint Petersburg, Russia, July 1–4, 2019, Proceedings, Part II*, vol. 19, Springer International Publishing, 2019, pp. 671-684.

[76] R. U. Kiran, M. Kitsuregawa, and M. Venu, "Selecting an Appropriate Data Mining Algorithm to Model the Compressive Strength of High Performance Concrete," 2012.

[77] A. Kostovska, A. Jankovic, D. Vermetten, J. de Nobel, H. Wang, T. Eftimov, and C. Doerr, "Per-run algorithm selection with warm-starting using trajectory-based features," in *Proceedings of the International Conference on Parallel Problem Solving from Nature*, pp. 46–60, Springer International Publishing, Cham, August 2022.

[78] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 826–830, 2016.

[79] G. Kou, Y. Lu, Y. Peng, and Y. Shi, "Evaluation of classification algorithms using MCDM and rank correlation," *International Journal of Information Technology & Decision Making*, vol. 11, no. 1, pp. 197-225, 2012.

[80] C. Kroer and Y. Malitsky, "Feature filtering for instance-specific algorithm configuration," in *Proceedings of the 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, pp. 849–855, 2011, IEEE.

[81] S. Kukkonen and J. Lampinen, "GDE3: The third evolution step of generalized differential evolution," in *2005 IEEE Congress on Evolutionary Computation*, vol. 1, pp. 443-450, Sept. 2005.

[82] S. Kukkonen and J. Lampinen, "Performance assessment of generalized differential evolution 3 (GDE3) with a given set of problems," in *2007 IEEE Congress on Evolutionary Computation*, IEEE, 2007, pp. 3593-3600.

[83] R. R. Kurada, K. K. Pavan, and A. V. Rao, "A preliminary survey on optimized multiobjective metaheuristic methods for data clustering using evolutionary approaches," *arXiv preprint* arXiv:1312.2366, 2013.

[84] B. Leite, A. O. S. da Costa, and E. F. da Costa Junior, "Multi-objective optimization of adiabatic styrene reactors using Generalized Differential Evolution 3 (GDE3)," *Chemical Engineering Science*, vol. 265, p. 118196, 2023.

[85] A. Liefooghe, S. Verel, B. Lacroix, A.-C. Zavoianu, and J. McCall, *Landscape features and automated algorithm selection for multi-objective interpolated continuous optimisation problems.* In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 421-429, 2021.

[86] M. Lindauer, H. Hoos, F. Hutter, and T. Schaub, "AutoFolio: An Automatically Configured Algorithm Selector," *Journal of Artificial Intelligence Research*, 2015.

[87] M. Lindauer, F. Hutter, H. H. Hoos, and T. Schaub, *AutoFolio: An Automatically Configured Algorithm Selector.* University of Freiburg, 2015.

[88] "LSP-NT," Available: https://systemevaluationandselection.com/LSPNT/login.php.

[89] K. Malan and A. P. Engelbrecht, "A survey of techniques for characterising fitness landscapes and some possible ways forward," *Information Sciences*, vol. 241, pp. 148–163, 2013.

[90] R. G. Mantovani, A. L. D. Rossi, E. Alcobaça, J. Vanschoren, and A. C. P. L. F. de Carvalho, "A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers," *Information Sciences*, vol. 501, pp. 193–221, 2019, Elsevier.

[91] "MCDA Calculator," Available: https://mcda-calculator.psi.ch/.

[92] I. Abi-Zeid and team, "MCDA-ULaval: a software tool for multicriteria decision aiding," Université Laval. Available: https://mcda.fsa.ulaval.ca/.

[93] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, and G. Rudolph, "Exploratory landscape analysis," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, 2011, pp. 829–836.

[94] K. D. Messer and W. L. Allen III, "Mathematical Foundations of the Logic Scoring of Preference Method," in *The Science of Strategic Conservation: Protecting More with Less*, Cambridge University Press, 2018, pp. 273–300. doi: 10.1017/9781108123778.014.

[95] K. Miettinen, *Nonlinear Multiobjective Optimization*. Springer, 1999.

[96] M. Mısır, S. D. Handoko, and H. C. Lau, "OSCAR: Online selection of algorithm portfolios with case study on memetic algorithms," in *Learning and Intelligent Optimization: 9th International Conference, LION 9, Lille, France, January 12-15, 2015. Revised Selected Papers*, vol. 9, pp. 59-73, Springer International Publishing, 2015.

[97] M. Mısır and M. Sebag, "Alors: An algorithm recommender system," *Artificial Intelligence*, vol. 244, pp. 291-314, 2017.

[98] M. Mısır, A. Gunawan, and P. Vansteenwegen, "Algorithm selection for the team orienteering problem," in *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*, pp. 33-45, Springer International Publishing, 2022.

[99] M. Mısır and X. Cai, "Algorithm Selection for Large-Scale Multi-objective Optimization," in *International Conference on Optimization and Learning*, pp. 36-47, Cham: Springer Nature Switzerland, 2023.

[100] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimisation," in *The 2003 Congress on Evolutionary Computation, 2003. CEC'03*, vol. 2, IEEE, 2003, pp. 878–885.

[101] S. Opricovic, "Programski paket VIKOR za visekriterijumsko kompromisno rangiranje," in *17th International symposium on operational research SYM-OP-IS*, 1990.

[102] S. Opricovic and G.-H. Tzeng, "Compromise solution by MCDM methods: A comparative analysis of VIKOR and TOPSIS," *European Journal of Operational Research*, vol. 156, no. 2, 2004, pp. 445–455.

[103] L. M. Pang, H. Ishibuchi, and K. Shang, "Counterintuitive experimental results in evolutionary large-scale multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 6, pp. 1609-1616, 2022.

[104] A. Panichella, "An adaptive evolutionary algorithm based on non-euclidean geometry for many-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019, pp. 595-603.

[105] A. Panwar, K. K. Tripathi, and K. N. Jha, "A qualitative framework for selection of optimization algorithm for multi-objective trade-off problem in construction projects," *Engineering, Construction and Architectural Management*, vol. 26, no. 9, pp. 1924-1945, 2019.

[106] L. M. Pavelski, M. Delgado, M.-É. Kessaci, and A. A. Freitas, "Stochastic local search and parameters recommendation: A case study on flowshop problems," *International Transactions in Operational Research*, vol. 30, no. 2, pp. 774–799, 2023.

[107] Y. Peng, G. Wang, and H. Wang, "User preferences based software defect detection algorithms selection using MCDM," *Information Sciences*, vol. 191, pp. 3-13, 2012.

[108] S. Petchrompo, D. W. Coit, A. Brintrup, A. Wannakrairot, and A. K. Parlikad, "A review of Pareto pruning methods for multi-objective optimization," *Computers & Industrial Engineering*, vol. 167, 2022, article 108022.

[109] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[110] S. Siraj, "PriEsT: Priority Estimation Tool (AHP)," Available: https://sourceforge.net/projects/priority/.

[111] "pymcdm": A Python package for multi-criteria decision making," Available: https://pypi.org/project/pymcdm/.

[112] J. R. Rice, "The Algorithm Selection Problem," in *Advances in Computers*, vol. 15, pp. 65–118, Elsevier, 1976. doi: 10.1016/S0065-2458(08)60520-3.

[113] B. Roy, "The outranking approach and the foundations of ELECTRE methods," *Theory and Decision*, vol. 31, pp. 49–73, 1991.

[114] T. L. Saaty, "What is the analytic hierarchy process?," in *Mathematical models for decision support*, Springer, 1988, pp. 109–121.

[115] T. L. Saaty, et al., *Decision making with dependence and feedback: The analytic network process*, vol. 4922, no. 2, RWS Publications, Pittsburgh, 1996.

[116] J. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, vol. 1, Jan. 1984.

[117] J. R. Schott, *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization*. PhD diss., Massachusetts Institute of Technology, 1995.

[118] H. Seada and K. Deb, "U-NSGA-III: A unified evolutionary algorithm for single, multiple, and many-objective optimization," *COIN Report 2014022*, 2014.

[119] K. Smith-Miles, "Cross-disciplinary perspectives on meta-learning for algorithm selection," *ACM Computing Surveys*, vol. 41, no. 1, pp. 1–25, 2009.

[120] A. Stoica, D. Keymeulen, R. S. Zebulum, A. Thakoor, T. Daud, G. Klimech, Y. Jin, R. Tawel, and V. Duong, "Evolution of Analog Circuits on Field Programmable Transistor Arrays," in *Proc. of 2nd NASA/DoD Workshop on Evolvable Hardware (EH2000)*, Palo Alto, CA, 13–15 July 2000, pp. 99–108.

[121] H. Taherdoost and M. Madanchian, "Multi-criteria decision making (MCDM) methods and concepts," *Encyclopedia*, vol. 3, no. 1, pp. 77–87, 2023.

[122] E.-G. Talbi, *Metaheuristics: From Design to Implementation*. John Wiley & Sons, 2009.

[123] Y. Tian, S. Peng, T. Rodemann, X. Zhang, and Y. Jin, "Automated selection of evolutionary multi-objective optimization algorithms," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 3225–3232, IEEE.

[124] A. Thompson, P. Layzell, and R. S. Zebulum, "Explorations in design space: unconventional electronics design through artificial evolution," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 3, pp. 167–196, 1999.

[125] L. Tian, "Test Problems for Large-Scale Multiobjective and Many-Objective Optimization," in *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 12, pp. 4108–4121, 2023.

[126] D. A. Van Veldhuizen, G. B. Lamont, and others, "Evolutionary computation and convergence to a Pareto front," in *Late Breaking Papers at the Genetic Programming 1998 Conference*, Citeseer, 1998, pp. 221–228.

[127] D. A. Van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. Air Force Institute of Technology, 1999.

[128] N. Veerapen, J. Maturana, and F. Saubion, "An Exploration-Exploitation Compromise-Based adaptive operator selection for local search," in *Proceedings of the 14th International Conference on Genetic and Evolutionary Computation (GECCO '12)*, ACM, 2012, pp. 1277–1284.

[129] W. Wei, C. M. Li, and H. Zhang, "Switching among Non-Weighting, clause weighting, and variable weighting in local search for SAT," in *Proceedings of the 14th International Conference on Principles and Practice of Constraint Programming*, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 313–326.

[130] W. Wu, Z. Xu, G. Kou, and Y. Shi, "Decision-making support for the evaluation of clustering algorithms based on MCDM," *Complexity*, 2020:1-17, 2020.

[131] X. Wu, Y. Zhong, J. Wu, and K. C. Tan, "AS-LLM: When Algorithm Selection Meets Large Language Model," *arXiv preprint arXiv:2311.13184*, 2023.

[132] X. Wu, Y. Zhong, J. Wu, and K. C. Tan, "Large Language Model-Enhanced Algorithm Selection: Towards Comprehensive Algorithm Representation," *arXiv preprint arXiv:2311.13184*, 2024.

[133] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "SATzilla: Portfolio-based algorithm selection for SAT," *Journal of Artificial Intelligence Research*, vol. 32, pp. 565-606, 2008.

[134] Y. Xu, D. Stern, and H. Samulowitz, "Learning adaptation to solve constraint satisfaction problems," in *Learning and Intelligent Optimization*, 2009.

[135] L. Xu, H. H. Hoos, and K. Leyton-Brown, "Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010.

[136] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Hydra-MIP: Automated Algorithm Configuration and Selection for Mixed Integer Programming," in *RCRA workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion*, 2011.

[137] N. Yang, Z. Tang, X. Cai, L. Chen, and Q. Hu, "Cooperative multi-population Harris Hawks optimization for many-objective optimization," *Complex & Intelligent Systems*, vol. 8, no. 4, pp. 3299-3332, 2022.

[138] S. Y. Yuen, C. K. Chow, and X. Zhang, "Which algorithm should I choose at any point of the search: An evolutionary portfolio approach," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, pp. 567–574, 2013.

[139] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[140] Q. Zhang *et al.*, "Multiobjective Optimization Test Instances for the CEC 2009 Special Session and Competition," Technical Report CES-487, The School of Computer Science and Electronic Engineering, University of Essex, 2009.

[141] X. Zhang, Z. Lv, Y. Liu, X. Xiao, and D. Xu, "Novel Multi-Criteria Group Decision Making Method for Production Scheduling Based on Group AHP and Cloud Model Enhanced TOPSIS," *Processes*, vol. 12, no. 2, p. 305, 2024.

[142] H. Zille and S. Mostaghim, "Linear search mechanism for multi-and many-objective optimisation," in *Evolutionary Multi-Criterion Optimization: 10th International Conference, EMO 2019, East Lansing, MI, USA, March 10-13, 2019, Proceedings*, vol. 10, Springer International Publishing, 2019, pp. 399-410.

[143] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000. MIT Press.

[144] E. Zitzler, M. Laumanns, and L. Thiele, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. Tik-report 103, ETH Zurich, 2001.

[145] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[146] E. Zitzler and S. Künzli, "Indicator-Based Selection in Multiobjective Search," in *Parallel Problem Solving from Nature - PPSN VIII*, 2004, pp. 832-842.

[147] F. Zou, D. Chen, H. Liu, S. Cao, X. Ji, and Y. Zhang, "A survey of fitness landscape analysis for optimization," *Neurocomputing*, vol. 503, pp. 129–139, 2022.