# 多目的最適化のためのハイブリッド適応進化アルゴリズムに関する研究

Doctoral Dissertation

# A Study of Hybrid Adaptive Evolutionary Algorithm for Multi-objective Optimization


Computational Intelligence Laboratory

Muroran Institute of Technology

Division of Information and Electronic Engineering

Han Jiayi

February 2024

# Contents

# Chapter 1 Introduction

## 1.1 Background of our study

The mathematical definition of a multi-objective optimization problem (MOP) [1,2] can be expressed as follows:

Assuming that there is a decision variable vector $\mathrm{x} = (x_1,\ x_2,\ x_3, \dots, x_n)$, where $x_i$ represents the $i$th decision variable. The objective function vector is denoted as $F(\mathrm{x}) = \big((f_1(\mathrm{x}),\ f_2(\mathrm{x}),\ f_3(\mathrm{x}), \dots, f_m(\mathrm{x})\big)$, where $f_j(\mathrm{x})$ represents the $j$th objective function.

The mathematical formulation of a multi-objective optimization problem is given by:

$$\text{Minimize (or Maximize) } F(\mathrm{x}) = \big((f_1(\mathrm{x}),\ f_2(\mathrm{x}),\ f_3(\mathrm{x}), \dots, f_m(\mathrm{x})\big) \qquad (1)$$

$$\text{Subject to: } \mathrm{x} \in X$$

Here, $X$ is the feasible solution space of the decision variable vector $\mathrm{x}$.

Starting from the mathematical definition of a multi-objective optimization problem, solving such problems requires considering multiple objectives simultaneously rather than focusing on a single objective. These types of problems are prevalent in the real world, and the multiple objectives involved often exhibit inherent conflicts and contradictions.

In the context of multi-objective optimization, Pareto dominance and Pareto optimal solutions are crucial concepts for assessing the relationships and qualities of solutions within a solution set. Pareto dominance [3,4] is a method of comparing different solutions in a solution set. Given two solutions, $A$ and $B$, solution $A$ Pareto dominates $B$ if $A$ is superior in at least one objective and is not inferior to $B$ in any other objective. Mathematically, the notation $A \preceq B$ is used to represent Pareto dominance. This implies that for all objective functions $f_j(A) \leq f_j(B)$, and there exists.

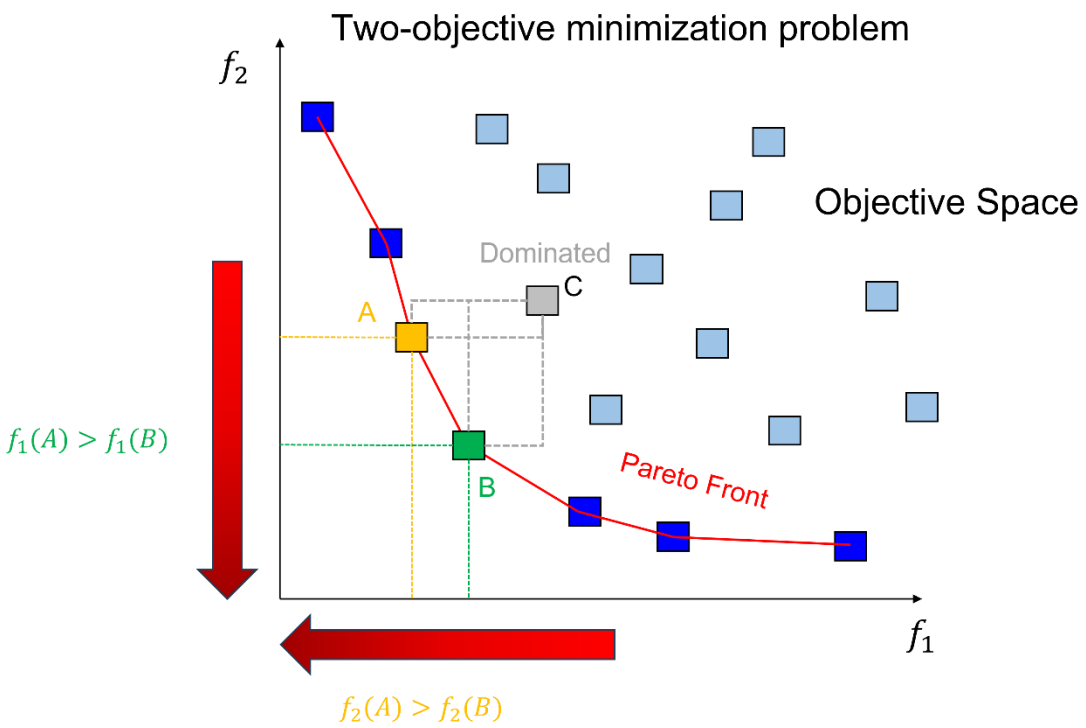


Figure 1: The basic concept of Pareto dominance

at least one $j$ such that $f_j(A) < f_j(B)$. The Pareto dominance diagram is illustrated in the following Figure 1.

Pareto optimal solutions are solutions within the solution space that cannot be dominated by any other solution in the set. In other words, a Pareto optimal solution cannot be improved in one objective without degrading performance in another. In a two-dimensional space, for instance, Pareto optimal solutions might lie on the boundary of the non-dominated solution set. In general, the goal of a multi-objective optimization problem is to identify the Pareto optimal solution set, where no other solution in the set outperforms all objectives simultaneously. This reflects the trade-offs and compromises that may exist among different objectives in a multi-objective context.

The study of multi-objective optimization problems holds significant practical importance [3-6]:

- Engineering Design: In the field of engineering, multi-objective optimization is widely applied for design optimization, aiming to balance multiple objectives such as cost, performance, and reliability.

- Financial Investment: Portfolio optimization represents a multi-objective problem, considering the balance between risk and return. Multi-objective optimization methods can identify the optimal investment portfolio.
- Supply Chain Management: In supply chain management, multi-objective optimization helps balance inventory costs, production efficiency, and customer service levels, enhancing the overall efficiency of the supply chain.
- Energy System Optimization: Multi-objective optimization in energy systems considers factors such as energy efficiency, environmental impact, and cost, providing effective solutions for sustainable development.
- Traffic Flow Optimization: In traffic management, multi-objective optimization is employed to minimize traffic congestion, reduce travel time, and mitigate environmental pollution, thereby improving the efficiency of the transportation system.

With the rise of evolutionary algorithms, such as genetic algorithms and particle swarm optimization, there has been remarkable success in addressing multi-objective optimization problems, propelling the research in this field.

The emergence of evolutionary algorithms has significantly advanced the field of multi-objective optimization. Algorithms like genetic algorithms [2,7,8] and particle swarm optimization have demonstrated exceptional performance in handling the complexities associated with optimizing multiple conflicting objectives simultaneously. This success has, in turn, stimulated and accelerated the research efforts dedicated to tackling multi-objective optimization problems.

Evolutionary algorithms [9,10], inspired by natural selection and population dynamics, offer effective solutions for exploring and exploiting the solution space, particularly in scenarios where objectives may be interrelated or in conflict. The adaptability and robustness of these algorithms make them well-suited for navigating the challenges posed by real-world problems characterized by diverse and competing goals.

In essence, the ascendancy of evolutionary algorithms has played a pivotal role in driving advancements in the study of multi-objective optimization, providing powerful tools to find optimal solutions in complex decision-making environments.

## 1.2 Purpose of our study

Evolutionary algorithms draw inspiration from the natural evolution of living organisms, encompassing fundamental operations such as genetic encoding, population initialization, crossover and mutation operators, and selection mechanisms. Compared to traditional calculus-based methods and exhaustive approaches, evolutionary computation stands out as a mature, highly robust, and widely applicable global optimization technique. It exhibits self-organization, self-adaptation, and self-learning characteristics, allowing it to effectively handle complex problems that traditional optimization algorithms struggle to address, irrespective of the nature of the problem. With the increasing participation of researchers, solving both single-objective and multi-objective optimization problems through evolutionary computation has become mainstream in recent years. Our research focus is on employing evolutionary computation methods to efficiently tackle multi-objective optimization problems.

Since evolutionary algorithms simulate the evolutionary process of organisms in nature, the evolution of individuals carrying information is crucial in evolutionary computation. The strategy for generating new individuals determines the algorithm's performance. In our research, we emphasize the study of individual generation operators based on CX (crossover-based) [11-14] and ED (Estimation of Distribution-based) [15-17] strategies.

On the other hand, as mentioned earlier, multi-objective optimization problems are more complex compared to single-objective optimization problems, requiring a systematic approach to addressing them. Currently, more advanced frameworks for solving multi-objective optimization problems include dominance-based (NSGA-II,

e.g.) [3], indicator-based (Inverted Generational Distance, e.g.) [18,19], and decomposition-based (MOEA/D, e.g.) [20] approaches. We specifically focus on the MOEA/D framework.

However, in our attempts to enhance the overall efficiency of the algorithm by designing new operators or modifying classical operators, we have observed that a single individual operator cannot handle all search situations. The search capability of operators is often described in terms of exploration and exploitation [21]. Exploration is the process of visiting entirely new regions of a search space, while exploitation is visiting those regions within the neighborhood of previously visited points. according to the ''No Free Lunch'' theory [22,23], it is difficult for one operator to exhibit both exploitation and exploration capabilities. Therefore, to overcome the aforementioned issue and enhance the search efficiency of the algorithm, it is essential to combine multiple operators with distinct search characteristics into a hybrid algorithm. Through an adaptive operator-switching mechanism, operators are selected based on varying search conditions.

Additionally, the adaptability of operators within the framework is also crucial. For most evolutionary computation operators, their design is originally intended to mimic the evolution of organisms in the natural world. In other words, many classical evolutionary operators did not initially consider addressing more complex multi-objective optimization problems. We focus on the MOEA/D framework because one of its most significant features is the ability to transform a multi-objective optimization problem into multiple subproblems through scalar function decomposition, with each subproblem treated as a single-objective optimization problem. Based on this characteristic, with necessary modifications, most classical evolutionary operators can be introduced into the MOEA/D framework. Furthermore, the MOEA/D framework introduces the concept of subproblem neighborhoods, which is another important feature. Individuals within the neighborhood are closely connected, and during the evolution process, information among subproblems (individuals) within the neighborhood is shared. Therefore, considering the distinctive

features of the MOEA/D framework, our research goal is to extend classical evolutionary operators for addressing multi-objective optimization problems.

## 1.3 Structure of this doctor thesis

This doctoral thesis will comprise six main chapters, and the outline for each chapter is as follows:

- Chapter 1 describes the background and purpose of the study.
- Chapter 2 serves as an introduction and review of fundamental concepts relevant to this research, encompassing detailed insights into the MOEA/D framework and foundational concepts of classical evolutionary operators.
- Chapter 3 discusses the adaptation of evolutionary operators within the MOEA/D framework. It covers the details of necessary modifications to evolutionary operators to align them with the unique characteristics of the MOEA/D framework.
- Chapter 4 introduces a hybrid model named Ensemble Framework based on MOEA/D (MOEA/D-EF). Its primary feature involves dividing the iterative process into distinct phases and adaptively selecting suitable evolutionary operators through a priori means.
- Chapter 5 introduces a Hyper-Heuristic multi-objective optimization approach based on MOEA/D framework (MOEA/D-HH) [24]. Its main characteristic is the integration of different types of evolutionary operators into the MOEA/D framework, adaptively selecting evolutionary operators that align with the current search state through an efficiency-based operator selection mechanism.
- Chapter 6 presents the conclusions of this doctor thesis and discusses future issues.

# Chapter 2 Related works

## 2.1 Introduction

To enhance the clarity and comprehensibility of our research, we can broadly divide the application of evolutionary algorithms to solve multi-objective optimization problems into two main components: the framework and the evolutionary operators.

As mentioned in the first chapter, our research is centered around the MOEA/D framework. Therefore, in this chapter, we will delve into an in-depth review of MOEA/D [20]. This review aims to provide a comprehensive understanding of MOEA/D's structure, key components, and unique features that distinguish it as a powerful tool for addressing multi-objective optimization challenges.

In our research, we employed two distinct types of evolutionary operators: crossover-based and distribution estimation-based operators. These two categories of operators are typical and exhibit markedly different search characteristics.

Crossover operators are fundamental evolutionary components that mimic genetic recombination processes in nature. They involve the combination of genetic information from two or more parent solutions to generate new offspring solutions. Crossover-based operators promote exploration by blending genetic material and creating diverse solutions that inherit traits from multiple parents. These operators are known for their ability to explore the solution space broadly.

Distribution estimation-based operators, on the other hand, focus on modeling the probability distribution of promising solutions in the population. These operators typically involve statistical techniques to estimate the underlying distribution of high-quality solutions. By emphasizing exploitation, distribution estimation-based operators guide the search toward regions of the solution space where better solutions are likely to be found. They leverage probabilistic models to exploit promising areas efficiently.

Additionally, incorporating specific performance metrics to quantify the disparity

and distribution of the solution set generated by an algorithm compared to the true Pareto front is a common practice in the study of multi-objective optimization problems. Performance metrics serve as valuable tools for not only explicitly assessing the algorithm's performance but also providing insights into the evolution direction of the population during the algorithm's iterative process. In this chapter, we will review two widely used performance metrics.

## 2.2 Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D)

Before the introduction of the MOEA/D algorithm, the majority of multi-objective genetic optimization methods did not rely on a decomposition strategy. They did not decompose Multi-Objective Optimization Problems (MOP) [1,2], meaning that each individual solution did not correspond to a single-objective optimization problem. In contrast, MOEA/D decomposes one MOP into $N$ subproblems using scalarization function [20,25-27], where each subproblem can be treated as a single-objective optimization problem. Through the evolution of the population, MOEA/D concurrently optimizes these subproblems. In each iteration, the algorithm retains the current optimal solution for each subproblem in the population.

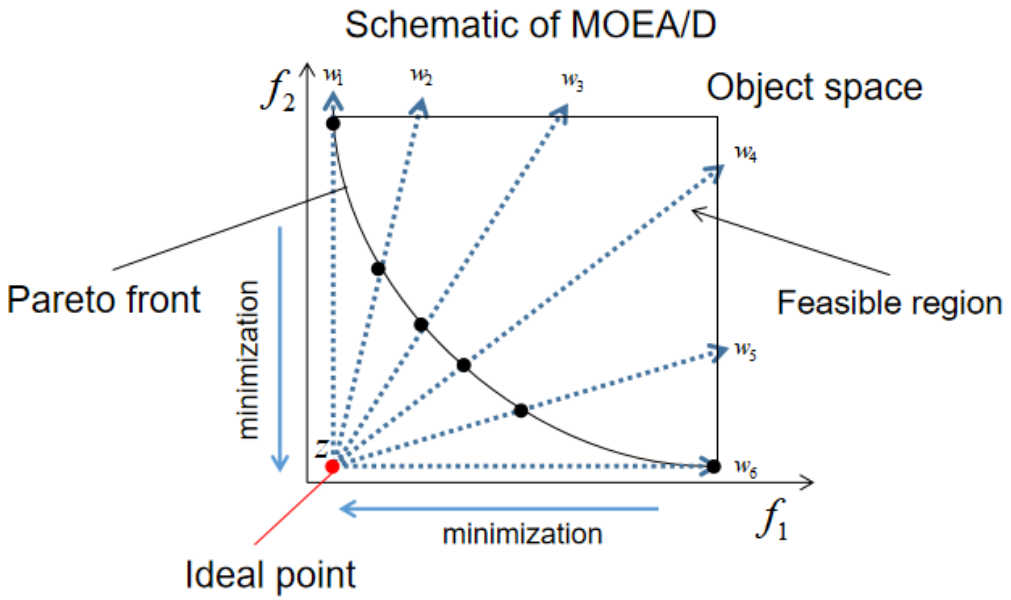


Figure 2 Schematic of MOEA/D. Uniformly generated weight vectors w decompose one MOP into several subproblems.

In the MOEA/D framework, there exists a neighborhood relationship among the subproblems, determined by their aggregation coefficients. The optimal solutions of two neighboring subproblems are guaranteed to be similar. Therefore, each subproblem only needs to optimize based on the information from its neighboring subproblems.

Figure 2 is used to demonstrate the basic concepts of MOEA/D. It is built upon the following idea: in the objective space, a set of weight vectors originating from reference point will inevitably intersect with the true Pareto front. If this set of weight vectors is uniformly distributed in the objective space, the intersections between the weight vectors and the true Pareto front will also be uniform. Therefore, if the population explores new individuals around the directions given by these weight vectors, the final result will be a good approximation of the true Pareto front.

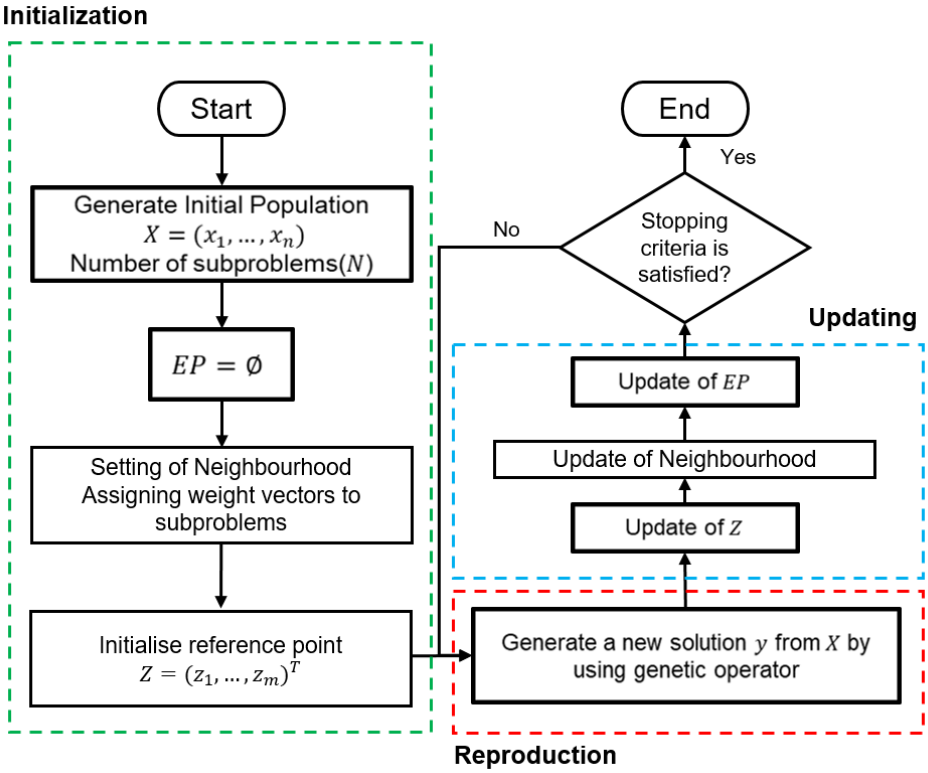


Figure 3 The Basic Flowchart of MOEA/D framework.

The scalarization function transforms the approximate optimization of the true Pareto front into a series of single-objective optimization problems, essentially achieving the decomposition of one MOP. The scalarization function establishes a connection between individuals and weight vectors. More specifically, through the computation of the scalarization function, we can evaluate the quality of individuals in a particular weight vector subproblem. For each weight vector subproblem, the optimal solution is retained in each iteration and carried over to the next iteration. This process enables individuals to search new solutions along the directions provided by the weight vectors.

The Basic Flowchart of MOEA/D framework is shown as Figure 3. Currently, there are various scalarization functions, but in our research, we primarily employed the penalty boundary (PBI) [20, 28] method, mathematically defined as follows:

$$minimize\ g^{bip}(x|\lambda, z^*) = d_1 + \theta d_2 \tag{2}$$

$$subject\ to\ x \in \Omega$$

where

$$d_1 = \frac{\left|\left|\left(z^* - F(x)\right)^T \lambda\right|\right|}{||\lambda||} \tag{3}$$

$$d_2 = \left||F(x) - (z^* - d_1\lambda)|\right| \tag{4}$$

$\lambda$ is weight vector, $z^* = (z_1^*, \dots, z_m^*)^T$ is reference point. For $i = 1, \dots, m$, usually definite $z_i^* = \max\{f_i(x)|x \in \Omega\}$. $\theta > 0$ is a preset penalty parameter. $d_1$ is the length of the projection of obtained solution $F(x)$ onto the current weight vector (current subproblem), $d_2$ is the distance from $F(x)$ to the current weight vector.

The flow of MOEA/D is in the following **Algorithm1**:

**Input:**

- MOP.
- A stopping criterion.
- $N$: the number of the subproblems considered in MOEA/D.
- A uniform spread of $N$ weight vectors: $\lambda^1, \dots, \lambda^N$ .
- $T$: the number of the weight vectors in the neighborhood of each weight vector.
- An external population (EP), which is used to store non-dominated solutions found during the search.

**Output:** EP

**Algorithm 1: Standard MOEA/D framework with PBI decomposition**

**Step 1) Initialization:**

**Step 1.1)** Set EP=∅.

**Step 1.2)** Compute the Euclidean distances between any two weight vectors and then work out the $T$ closest weight vectors to each weight vector. For each $i = 1, \dots, N$, set $B(i) = \{i_1, \dots, i_T\}$, where $\lambda^{i_1}, \dots, \lambda^{i_T}$ are the $T$ closest weight vectors to $\lambda^i$.

**Step 1.3)** Generate an initial population $x^1, \dots, x^N$ randomly or by a problem-specific method.

**Step 1.4)** Initialize reference point $\mathrm{z} = (z_1, \dots, z_m)^T$.

**Step 2) Update:**

For $i = 1, \dots, N$, do

**Step 2.1) Reproduction:** Randomly select two indexes $k, l$ form $B(i)$, and then generate a new solution $y$ from $x^k$ and $x^l$ by using genetic operators.

**Step 2.2) Improvement:** Apply a problem-specific repair/improvement heuristic on $y$ to produce $y'$.

**Step 2.3) Update of reference point:** For each $j = 1, \dots, m$, if $z_j < f_j(y')$, then set $z_j = f_j(y`)$.

**Step 2.4 Update of Neighboring Solutions:** For each index $j \in B(i)$, if $g^{bip}(y'|\lambda^j, \mathrm{z}) \leq g^{bip}(x^j|\lambda^j, \mathrm{z})$, then set $x^j = y'$.

**Step 2.5) Update of EP:** Remove form EP all the vectors dominated by $F(y')$. Add $F(y')$ to EP if no vectors in EP dominate $F(y')$.

**Step 3) Stopping Criteria:** If stopping criteria is satisfied, then stop and output EP. Otherwise, go to Step 2.

MOEA/D provides a simple yet efficient way of introducing decomposition approaches into multi-objective evolutionary computation. A decomposition approach, often developed in the community of mathematical programming, can be readily incorporated into evolutionary algorithms in the framework MOEA/D for solving MOPs. Another point worth noting is that, within the framework of MOEA/D, all evolution revolves around the concept of a neighborhood. This aspect is crucial for our subsequent research.

## 2.3 Crossover-based evolution operator

Crossover-based evolutionary operators refer to operators in evolutionary algorithms that primarily rely on the concept of crossover. Crossover is a genetic operator inspired by the natural process of genetic recombination. It involves combining genetic information from two or more parent solutions to generate new offspring solutions.

In the context of evolutionary algorithms, especially genetic algorithms, crossover is a fundamental mechanism for exploring the solution space by creating diverse solutions. The crossover-based evolutionary operators contribute to the exploration phase of the algorithm by blending genetic material and generating novel solutions that inherit traits from multiple parents.

These operators are contrasted with other evolutionary operators, such as mutation or selection, which serve different purposes in guiding the evolution of the population. The combination of various evolutionary operators, including crossover-based operators, aims to strike a balance between exploration and exploitation, enhancing the algorithm's ability to discover high-quality solutions in the search space.

The Differential Evolution (DE) [11,29] algorithm is one of the most representative and widely used crossover-based evolutionary operators. It primarily consists of three main strategies: mutation, crossover, and selection. The pseudo code of basic DE algorithm is shown in **Algorithm2**.

**Algorithm2: Basic DE algorithm (with DE/rand/1/bin)**

Generate a uniformly distributed random initial population including $NP$ solutions that contain $D$ variables according to $\mathrm{X}_{i,j}^{0} = \mathrm{X}_{j}^{min} + rand(0,1) \cdot \left(\mathrm{X}_{j}^{max} - \mathrm{X}_{j}^{min}\right)$ ($i \in [1, NP], j \in [1, D]$)
**while** termination condition is not satisfied
  **for** $i = 1$ to $NP$
    Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$ //**mutation**
    $\mathrm{V}_{i}^{G} = \mathrm{X}_{r1}^{G} + F \cdot (\mathrm{X}_{r2}^{G} - \mathrm{X}_{r3}^{G})$ //**end mutation**
    $j_{rand} = randind(1, D)$ //**crossover**
    **for** $j = 1$ to $D$
      **if** $rand(0,1) \leq CR$ or $j == j_{rand}$
        $\mathrm{U}_{i,j}^{G} = \mathrm{V}_{i,j}^{G}$
      **else**
        $\mathrm{U}_{i,j}^{G} = \mathrm{X}_{i,j}^{G}$
      **end if**
    **end for** //**end crossover**
    **if** $f(\mathrm{U}_{i}^{G}) \leq f(\mathrm{X}_{i}^{G})$ //**selection**
      $\mathrm{X}_{i}^{G+1} = \mathrm{U}_{i}^{G}$
    **else**
      $\mathrm{X}_{i}^{G+1} = \mathrm{X}_{i}^{G}$ //**end selection**
  **end for**
**end while**

DE adopts the mutation operator to generate the mutant vector (also called donor vector) $\mathrm{V}_i^G = (v_{i,1}^G, v_{i,2}^G, \ldots, v_{i,D}^G)$ with respect to each individual $\mathrm{X}_i^G$ (called target vector), where $D$ is the number of decision variables, $G$ is the generation time.

In order to distinguish among these schemes, the notation $\mathrm{DE}/x/y/z$ [30] is used, where $x$ represents the vector (also called basic vector) to be perturbed, $y$ denotes the number of difference vectors considered for perturbation and $z$ is the crossover scheme. Some well-known mutation strategies are listed as follows [31]:

$$\mathrm{DE1\ DE/rand/1}: \mathrm{V}_i^G = \mathrm{X}_{r1}^G + F \cdot \left(\mathrm{X}_{r2}^G - \mathrm{X}_{r3}^G\right) \tag{5}$$

$$\mathrm{DE2\ \ DE/rand/2}: \mathrm{V}_i^G = \mathrm{X}_{r1}^G + F \cdot \left(\mathrm{X}_{r2}^G - \mathrm{X}_{r3}^G\right) + F \cdot \left(\mathrm{X}_{r4}^G - \mathrm{X}_{r5}^G\right) \tag{6}$$

$$\mathrm{DE3\ \ DE/best/1}: \mathrm{V}_i^G = \mathrm{X}_{best}^G + F \cdot \left(\mathrm{X}_{r1}^G - \mathrm{X}_{r2}^G\right) \tag{7}$$

$$\mathrm{DE4\ \ DE/rand-to-best/1}: \mathrm{V}_i^G = \mathrm{X}_{r1}^G + F \cdot \left(\mathrm{X}_{best}^G - \mathrm{X}_{r2}^G\right) \tag{8}$$

$$\mathrm{DE5\ \ DE/current-to-rand/1}: \mathrm{V}_i^G = \mathrm{X}_i^G + F \cdot \left(\mathrm{X}_{r1}^G - \mathrm{X}_i^G\right) + F \cdot \left(\mathrm{X}_{r2}^G - \mathrm{X}_{r3}^G\right) \tag{9}$$

$$\mathrm{DE6\ \ DE/current-to-best/1}: \mathrm{V}_i^G = \mathrm{X}_i^G + F \cdot \left(\mathrm{X}_{best}^G - \mathrm{X}_i^G\right) + F \cdot \left(\mathrm{X}_{r1}^G - \mathrm{X}_{r2}^G\right) \tag{10}$$

where $F$ is the mutation scaling factor which is generally restricted in the range (0, 1], $\mathrm{X}_{best}^G$ is the best individual in current generation $G$, $r1$, $r2$, $r3$, $r4$ and $r5$ are randomly selected from the current population, and $r1 \neq r2 \neq r3 \neq r4 \neq r5 \neq i$. Generally, different mutation strategies have distinct characteristics, which have the advantages for solving certain kinds of problems.

After mutation, a trial vector $\mathrm{U}_i^G = (u_{i,1}^G, u_{i,2}^G, \ldots, u_{i,D}^G)$ is obtained according to a binomial crossover strategy on $\mathrm{X}_i^G$ and $\mathrm{V}_i^G$ as follow:

$$\mathrm{U}_{i,j}^G = \begin{cases} \mathrm{V}_{i,j}^G & \text{if } \left(rand_{i,j}(0,1) \leq CR \text{ or } j == j_{rand}\right) \\ \mathrm{X}_{i,j}^G & \text{otherwise} \end{cases} \tag{11}$$

where $rand_{i,j}(0,1)$ is a uniformly distributed random real number in $[0,1]$ and $j_{rand}$ is a uniformly distributed random integer in $[1, D]$. $CR$ is the crossover rate. If the $j$th variable $\mathrm{U}_{i,j}^G$ of the trial vector $\mathrm{U}_i^G$ violates the boundary constraints, it will be reset as follows:

$$\mathrm{U}_{i,j}^G = \begin{cases} \min\left\{\mathrm{X}_j^{\max}, 2\mathrm{X}_j^{\min} - \mathrm{U}_{i,j}^G\right\} & \text{if } \mathrm{U}_{i,j}^G < \mathrm{X}_j^{\mathrm{mix}} \\ \max\left\{\mathrm{X}_j^{\min}, 2\mathrm{X}_j^{\max} - \mathrm{U}_{i,j}^G\right\} & \text{if } \mathrm{U}_{i,j}^G > \mathrm{X}_j^{\max} \end{cases} \tag{12}$$

By simulating the rule of natural evolution, selection operator will keep the better one from the target vector $\mathrm{X}_i^G$ and the trial vector $\mathrm{U}_i^G$. This chosen vector will survive and enter the next evolutionary generation. Without loss of generality, for a minimization problem, the vector with better fitness value is selected as follows:

$$\mathrm{X}_i^{G+1} = \begin{cases} \mathrm{U}_i^G & \text{if } f\left(\mathrm{U}_i^G\right) \le f\left(\mathrm{X}_i^G\right) \\ \mathrm{X}_i^G & \text{otherwise} \end{cases} \tag{13}$$

It is noted that when the trial vector $\mathrm{U}_i^G$ is better than or equal to the parent $\mathrm{X}_i^G$, it is called a successful update (or successful replacement). Otherwise, it is marked as an unsuccessful update.

## 2.4 Estimation of distribution-based evolution operator

The concept of distribution estimation algorithms was initially proposed by *Mühlenbein* in 1996 [32]. The main idea is to integrate natural evolutionary algorithms with constructive mathematical analysis methods to guide an effective search in the problem space.

Distribution estimation algorithms are fundamentally a novel type of evolutionary algorithm based on probability models, combining genetic algorithms with statistical learning. It represents another typical implementation pattern in natural computation. These algorithms guide the search space for the next step by establishing a probability model based on the current set of relatively optimal individuals. New individuals are then sampled from the probability distribution function obtained from the distribution of relatively optimal solutions.

By describing the distribution of candidate solutions in the space through a probability model, distribution estimation algorithms use statistical learning methods to establish a probabilistic model describing the distribution of solutions from a macroscopic perspective. The algorithm then stochastically samples from this probability model to generate a new population. This process is repeated iteratively, facilitating the evolution of the population until termination conditions are met (Modeling-Sampling-Modeling-Sampling loop). The concept of comparison between genetic algorithm and distribution estimation algorithm is shown in Figure 4.

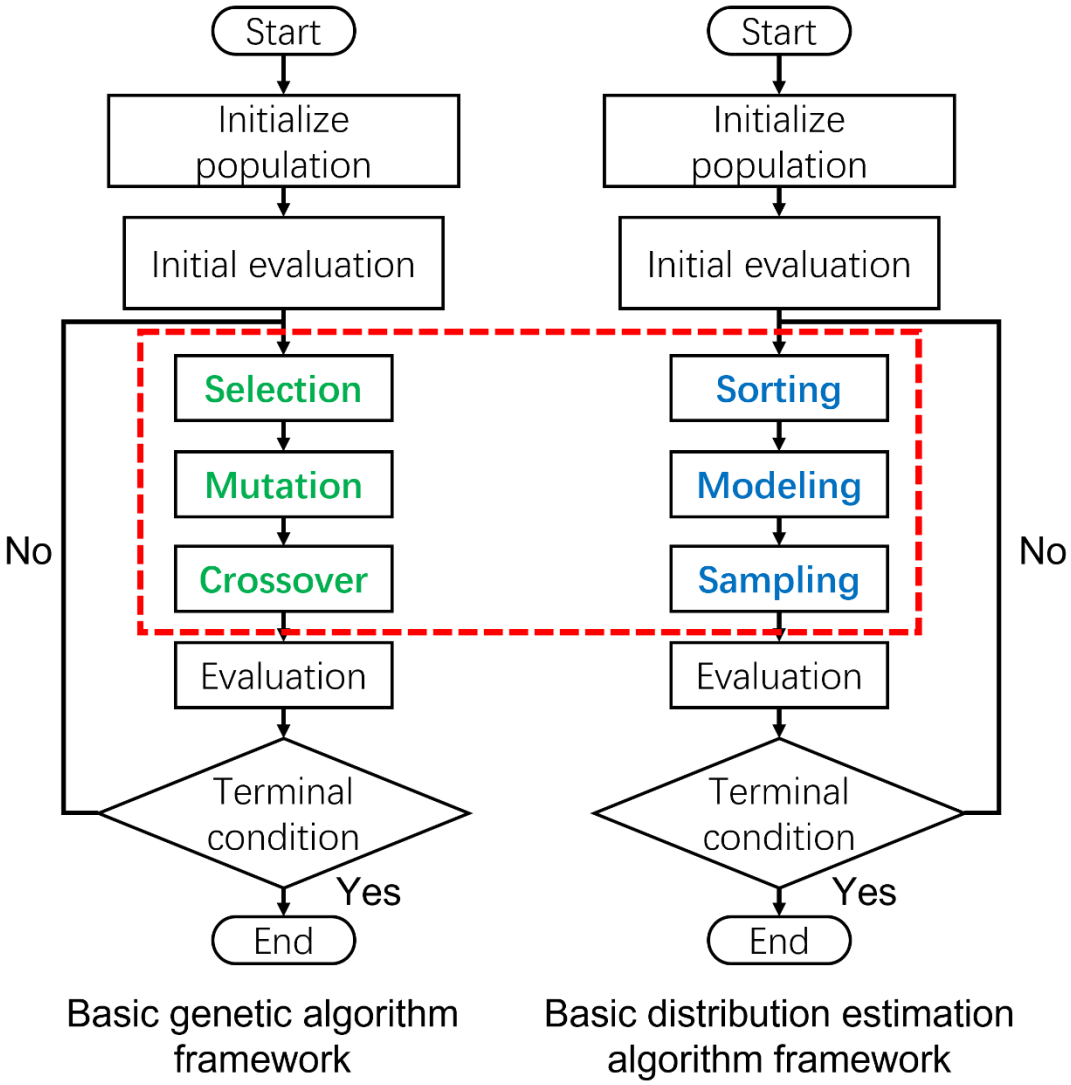


Figure 4 Comparison of flow between genetic algorithm and distribution estimation algorithm.

The general process of distribution estimation algorithms is illustrated in **Algorithm3** as follows:

**Input:**

- population size $NP$
- the number of selected individuals $k$ (top k individuals)

**Output:** the best individual and its fitness

**Algorithm3: Basic estimation of distribution algorithms**

**Step 1) Initialization:** Randomly initialize the population.

**while** the termination criteria are not satisfied

**Step 2) Ranking:** Sort the population in descending order based on fitness and select the $k$ ($k < NP$) best individuals as the optimal ones.

**Step 3) Modeling:** Estimate the probability distribution of the population according to the selected individuals.

**Step 4) Sampling:** Sample new individuals according to the estimated distribution.

**Step 5) Update:** Combine the sampled individuals and the old population to create a new population with $NP$ individuals.

**end while**

## 2.5 Inverted Generational Distance indicator

IGD (Inverted Generational Distance) [18] is a metric used to evaluate the performance of multi-objective optimization algorithms, providing a simultaneous assessment of both convergence and diversity. IGD involves uniformly sampling points from the true Pareto front. For each point on the real front, the algorithm identifies the nearest point on the obtained Pareto set, computes the distances between these pairs, and calculates the average. The mathematical definition is as follows:

$$IGD = \frac{1}{|P|}\sum_{p\in P} \min_{z\in Z} d(p,z) \tag{14}$$

where $|P|$ is the cardinality of the true Pareto front, $P$ is the set of points on the true Pareto front, $Z$ is the obtained Pareto set from the algorithm, $d(p,z)$ is the distance metric (e.g., Euclidean distance) between a point $p$ on the true Pareto front and its nearest neighbor $z$ on the obtained Pareto set.

In the context of IGD (Inverted Generational Distance), a smaller value indicates better performance of the algorithm in multi-objective optimization. The IGD metric measures the average distance between the points on the true Pareto front and their nearest counterparts in the obtained Pareto set generated by the algorithm. Therefore, a lower IGD value implies that the algorithm's solutions are closer to the true Pareto front, indicating better convergence and accuracy in approximating the Pareto front for multi-objective optimization problems.

## 2.6 Hypervolume indicator

Hypervolume (HV) is one of the commonly used performance evaluation metrics in multi-objective optimization problems, initially proposed by *Zitzler* et al [33]. It represents the volume of the hypercube formed by the individuals in the solution set and a reference point in the objective space. Hypervolume is utilized to measure the coverage and distribution of the Pareto front. The evaluation method of the Hypervolume metric is Pareto-compliant, meaning that if one solution set $S$ is superior to another solution set $S'$, then the Hypervolume metric of S will be greater than that of $S'$.

The accuracy of calculating the Hypervolume metric depends on the choice of the reference point. Different choices of reference points for evaluating the same solution set can lead to different computation results. The computation of Hypervolume is based on the points on the Pareto front, and these points should ideally have objective values that are as close to or better than a chosen reference point. Mathematically, the calculation of Hypervolume is expressed as:

$$HV = \int_{-\infty}^{f_1^*} \int_{-\infty}^{f_2^*} \dots \int_{-\infty}^{f_m^*} V(z)dz \tag{15}$$

where m is the number of objective functions, $f_i^*$ is the reference point for the $i$th objective function, $V(z)$ represents the hypervolume of a point $z$ on the Pareto front.

# Chapter 3 Evolution operators in MOEA/D framework

## 3.1 Introduction

As introduced in Chapter1, the purpose of our research is to develop efficient algorithms for solving multi-objective optimization problems by integrating various evolutionary operators with different characteristics into the MOEA/D framework. Therefore, in addition to the fundamental operators discussed in Chapter2, we explore several advanced operators. However, many evolutionary operators were initially designed for single-objective optimization problems. Some advanced operators cannot be directly applied to multi-objective optimization problems, and others, while applicable, may exhibit reduced efficiency compared to solving single-objective optimization problems. Given these challenges, our research focuses extensively on the MOEA/D framework.

As described in Chapter2, a significant characteristic of MOEA/D is its ability to decompose one MOP into multiple subproblems, each of which can be treated as a single-objective optimization problem. Therefore, introducing operators designed for single-objective optimization problems into the MOEA/D framework is both appropriate and natural. Another feature of the MOEA/D framework is that all evolution is conducted within the neighborhoods of subproblems. Hence, when incorporating operators into the MOEA/D framework, careful consideration must be given to this characteristic, necessary adaptive modifications to operators.

In this chapter, we will discuss several evolutionary operators that are prominently utilized in our research: Improved Differential Evolution (IDE) [34], Adaptive Differential Evolution with Optional External Archive (JADE) [35], DE-IDEAL, and Covariance matrix adaptation evolution strategy (CMA-ES) [36-37]. Among them, IDE and JADE were originally designed for solving single-objective optimization problems. Through adaptive modifications, we have extended and introduced them into the MOEA/D framework for addressing multi-objective optimization problems. DE-IDEAL is an original operator developed in our research, characterized by the use of the "vector pool" concept, which retains, and leverages vectors generated during the evolution process, containing information relevant to the correct evolution direction to some extent. CMA-ES is a distribution estimation-based operator, and its logic for generating new individuals differs from the other three operators, resulting in distinct search characteristics. We will delve into the details of these operators in the following.

## 3.2 Improved Differential Evolution (IDE) in MOEA/D

In the original DE, every individual generates new solutions in the same way by crossover and mutation. Since the way of generating a new solution is fixed and not-flexible, individuals tend to be gathered and are difficult to promote their search in the stagnation situation. To overcome this problem, original IDE uses two different approaches for generating new solutions according to the quality of individuals. The cores of IDE's approach are individual-dependent parameter (IDP) setting and individual-dependent mutation (IDM) strategy. To balance the possibility of each individual in the current generation for generating new individual, the IDP setting firstly ranks individuals in ascending order according to their fitness values.
For example, assuming that individual $x_i$ is the $i$th most superior one. The mutation factor (scale factor) $F_i$ and crossover probability (crossover rate) $CR_i$ can be set as:

$$F_i = CR_i = \frac{i}{N} \tag{16}$$

Obviously, individuals who rank poorly are given relatively large parameter values. More reasonable, a random algorithm based on normal distribution is used to improve IDP further. Denote a random number is created from a normal distribution by $randn(\text{mean}, \text{std})$ [35], then the IDP setting can be modified as follows:

$$F_i' = rand(F_i, 0.1) \tag{17}$$

$$CR_i' = rand(CR_i, 0.1) \tag{18}$$

It has been wildly known that different evolutionary operators have different characteristics. For example, mutation strategies with two difference vectors can increase the diversity in comparison with the case of using a single vector. Also, mutation strategies using the best individual as the base vector can strengthen exploitive search and achieve high-speed convergence. In IDM, all individuals are divided into two groups: Superior group ($S$) and Inferior group ($I$). These groups use different parameter values of the mutation and the methods of crossover operations for generating new individuals. Different group has different purpose: The superior group takes the role of exploiting their neighborhoods. The inferior group tries to explore the further area. Parameter $ps$ is used as the size of ratio between superior and inferior groups. This parameter is calculated by the following equation:

$$ps = 0.1 + 0.9 \times 10^{5(g/g_{\max}-1)} \tag{19}$$

where $g$ is the number of the current iteration and $g_{\max}$ is the number of the terminal iteration.

Further, *Linxin Tang and etc.* [38] also analyzed the Fitness Error, Diversity Indicator and Success Rate (SR) of the original DE algorithm. Generally, when iteration index reaches threshold $g_t$, the SR decline toward to 0, and the diversity of individuals also decline dramatically. This means that the other mutation operator

Table 1 Mutation strategy in IDE

| **Generation Index** | **The group of target individual** | **Mutation Operator** |
|---|---|---|
| $g \leq g_t$ | superior | $\mathrm{DE/current-to-rand/1(p)}$ |
| $g > g_t$ | inferior | $\mathrm{DE/current-to-better/1(p)}$ |
| $g \leq g_t$ | superior | $\mathrm{DE/rand/2(p)}$ |
| $g > g_t$ | inferior | $\mathrm{DE/rand-to-better/1(p)}$ |

should be replaced by the current operation, because it is difficult to generate better solutions now. Based on the above idea, IDM divides the entire iterations into two stages: earlier stage and later stage. IDM assigns different mutation operators for individuals of two groups in different stages. As same as original DE algorithm, IDM strategy can be denoted by a four-parameter notation: $\mathrm{DE/base/differ/cross}$. The following Table 1 shous detail settings of mutation operators in IDE

After understanding the basic logic of IDE, we attempted to modify and introduce it into the MOEA/D framework. However, a challenge we needed to confront was the requirement for a sufficient number of individuals to participate in the IDE process. When IDE is used to solve single-objective optimization problems, all individuals in the population are divided into Superior group and Inferior group. However, in the MOEA/D framework, the individuals available for optimizing a subproblem are only those within the neighborhood $B$ of that current subproblem [31,44]. We typically consider the neighborhood size $T$ not to be too large (less than or equal to 10% of the number of subproblem $N$) because subproblems within the neighborhood should have sufficiently high correlation. It is evident that when dealing with a small population size for a multi-objective optimization problem, relying solely on the existing individuals within the neighborhood cannot guarantee that both the Superior group and Inferior group have a sufficient number of individuals.

To address this issue, a straightforward and effective modification strategy is to increase the number of individuals in the population. In the original MOEA/D, the number of subproblems is usually equal to the population size. When introducing the IDE operator, we allocate more individuals for each subproblem. IDE operator in MOEA/D framework works as follows:

**Input:**

- A randomly generated population.
- Termination conditions.
- $N$: The number of subproblems.
- $T$: The size of neighborhood.
- $M$: The size of solution magnification.

**Output:** EP

**Algorithm4: IDE operator in MOEA/D framework**

**Step 1) Initialization:**

**Step 1.1) Initialize solution:** Generate $M \times N$ individuals randomly as initial solutions $IS$ .

**Step 1.2)** Initialize the reference point $\mathrm{z} = (z_1, \dots, z_m)^T$.

**Step 1.3) Distributing:**

**set** $i = N$

**while** $i > 0$ do:

Randomly select index $k \in i$, calculate $Y = g^{bip}(y|\lambda^k, \mathrm{z})$ $(y \in IS)$, sort $Y$ from largest to smallest, assign $y_1, \dots, y_M$ to subproblem $k$;

$i - 1$;

**Step 2) Update:**

**Step 2.1) Parameter calculation:** Calculate the value of $ps$, determine the size of superior(S) group and inferior (I) group in current generation $g$.

**for** $i = 1$ to $N$:

**Step 2.2) Recombination:** Extract $T \times M$ individuals in $B(i)$ as current sub-solution $TM$, calculate $Y' = g^{bip}(y'|\lambda^i, \mathrm{z})$ $(y' \in TM)$, sort $Y'$ from largest to smallest, divide $Y'$ into $S$ and $I$ according to $ps$ value, calculate parameter $F'_{y'}$ and $CR'_{y'}$.

**for** $k$ in $TM$:

**Step 2.3) Mutation:**

$$\mathrm{V}_k^g = \mathrm{X}_k^g + \begin{cases} F_k \cdot (\mathrm{X}_{r1}^g - \mathrm{X}_k^g) + F_k \cdot (\mathrm{X}_{r2}^g - \mathrm{DI}^g) & k \in S \\ F_k \cdot (\mathrm{X}_S^g - \mathrm{X}_k^g) + F_k \cdot (\mathrm{X}_{r2}^g - \mathrm{DI}^g) & k \in I \end{cases} \qquad g \le g_t$$

$$\mathrm{V}_k^g = \mathrm{X}_k^g + \begin{cases} F_k \cdot (\mathrm{X}_{r1}^g - \mathrm{X}_{r2}^g) + F_k \cdot (\mathrm{X}_{r3}^g - \mathrm{X}_{r4}^g) + F_k \cdot (\mathrm{X}_{r5}^g - \mathrm{DI}^g) & k \in S \\ F_k \cdot (\mathrm{X}_S^g - \mathrm{X}_{r1}^g) + F_k \cdot (\mathrm{X}_{r2}^g - \mathrm{DI}^g) & k \in I \end{cases} \qquad g \le g_t$$

$k \neq r1 \neq r2 \neq r3 \neq r4 \neq r5 \neq S$

**Step 2.4) Crossover:** As same as original DE in **Algorithm2.**

**Step 2.5) Update of reference point:** As same as Step 2.3 in **Algorithm1.**

**Step 2.6) Update of $TM$:** As same as Step 2.4 in **Algorithm1.**

**Step 2.7) Update of EP:** As same as Step 2.5 in **Algorithm1.**

**Step 3) Stopping Criteria:** As same as Step 3 in **Algorithm1.**

In Step 2.2, $\mathrm{X}_S^g$ represents a randomly selected individual from group $S$. $\mathrm{DI}^g$ is a disturbed individual derived from the current population with possible elements randomly taken from the entire solution space which determined in the following:

$$\mathrm{DI}_j = \begin{cases} \mathrm{L}_j + rand(0,1) \cdot \left(\mathrm{U}_j - \mathrm{L}_j\right) & \text{if } (0,1) < 0.1 \times ps \\ \mathrm{X}_r^j & \text{otherwise} \end{cases} \tag{20}$$

where $j \in (1, D)$ represents decision variables, $\mathrm{U}_j$ and $\mathrm{L}_j$ are the lower and the upper bound values of the $j$th decision variables. $\mathrm{X}_r^j$ is as same as $\mathrm{X}_{r1}^g$ or $\mathrm{X}_{r2}^g$, represents a randomly selected individual in $TM$.

The results of numerical experiments indicate that the aforementioned extension of IDE in the MOEA/D framework is effective, especially when the population size is small, and the neighborhood size is also small ($N \leq 100, T \leq 10$). However, MOEA/D-IDE still exhibits noticeable limitations. It is evident that, compared to the original DE operator, to ensure the stable operation of IDE in the MOEA/D framework, $M$ times more individuals need to be simultaneously maintained (updated). This directly results in IDE's evaluation cost being $M$ times higher than the original DE operator. When the population size is small enough, individuals assigned to the inferior group $I$ have a probability of generating new solutions that could be superior for other subproblems. Some losses in evaluation costs are acceptable. However, when the population size is large (e.g., $N > 300, T > 20$), the likelihood of this scenario occurring becomes negligible. In other words, a significant amount of evaluation cost is wasted on maintaining inferior individuals without hope. This significantly reduces the overall search efficiency of MOEA/D-IDE.

To address the aforementioned issue, when incorporating the IDE operator in the hybrid algorithm, there is no longer a deliberate generation of $M$ times more individuals. Instead, additional populations generated by other operators in the hybrid algorithm (such as sampling points generated by the CMA-ES operator) are utilized to ensure a sufficient number of individuals for the IDE to function properly. Simultaneously, when using the IDE operator, not all individuals involved in IDE calculations are maintained. Only individuals with the best fitness for a specific vector

are retained.

Another important insight brought by the IDE operator is the concept of grouping. In the original DE operator, $\mathrm{X}_{r1}^{G}$, $\mathrm{X}_{r2}^{G}$ or $\mathrm{X}_{r3}^{G}$ are randomly selected. This means that the direction of the difference vector is also random, implying that the difference vector could represent either the correct evolutionary direction or its opposite. If we follow the IDE approach of sorting and grouping the populations involved in the calculation by fitness, then the $\mathrm{X}_{S}^{G}$ selected from the $S$ group and the $\mathrm{X}_{I}^{G}$ selected from the $I$ group will definitely form a difference vector with information pointing towards the evolutionary direction, not its opposite. Therefore, in the hybrid algorithm, we retained the grouping aspect inspired by IDE.

## 3.3 Adaptive Differential Evolution with Optional External Archive (JADE) in MOEA/D

Adaptive Differential Evolution with Optional External Archive which named as JADE for short, initially developed by *Zhang and Sanderson* [35], is a simple yet efficient DE variants. $\mathrm{DE/rand/1}$ is the first mutation strategy developed for DE and is said to be the most successful and widely used scheme in the literature [29, 39, 40]. However, [41] indicates that $\mathrm{DE/best/2}$ may have some advantages over $\mathrm{DE/rand/1}$, and [42] favors $\mathrm{DE/rand/1}$ for most technical problems investigated. Also, the authors of [43] argue that the incorporation of best solution information is beneficial and use $\mathrm{DE/current-to-best/1}$ in their algorithm. Compared to $\mathrm{DE/rand/}k$, greedy strategies such as $\mathrm{DE/current-to-best/}k$ and $\mathrm{DE/best/}k$ benefit from their fast convergence by incorporating best solution information in the evolutionary search. However, the best solution information may also cause problems such as premature convergence due to the resultant reduced population diversity. In view of the fast but less reliable convergence performance of existing greedy strategies $\mathrm{DE/current-to-best/1}$ and $\mathrm{DE/best/1}$, a new mutation strategy, named $\mathrm{DE/current-to-}p\mathrm{best/1}$ (without archive) with optional archive, is proposed to serve as the basis

mutation strategy in of the adaptive JADE shown as follows:

$$\mathrm{V}_{i,g} = \mathrm{X}_{i,g} + F_i \cdot \left(\mathrm{X}^{p}_{\mathrm{best},g} - \mathrm{X}_{i,g}\right) + F_i \cdot \left(\mathrm{X}_{r1,g} - \mathrm{X}_{r2,g}\right) \tag{21}$$

where $\mathrm{X}^{p}_{\mathrm{best},g}$ is randomly chosen as one of the top $100p\%$ individuals in the current population with $p \in (0,1]$, and $F_i$ is the mutation factor that is associated with $\mathrm{X}_i$ and is re-generated at each generation by the adaptation process (details of $F_i$ will be introduced later). $\mathrm{DE/current-to-}p\mathrm{best/1}$ is indeed a generalization of $\mathrm{DE/current-to-best/1}$. Any of the top $100p\%$ solutions can be randomly chosen to play the role of the single best solution in $\mathrm{DE/current-to-best/1}$.

Recently explored inferior solutions, when compared to the current population, provide additional information about the promising progress direction. Denote $\boldsymbol{A}$ as the set of archived inferior solutions and $\boldsymbol{P}$ as the current population. In $\mathrm{DE/current-to-}p\mathrm{best/1}$ with archive, a mutation vector is generated as follows:

$$\mathrm{V}_{i,g} = \mathrm{X}_{i,g} + F_i \cdot \left(\mathrm{X}^{p}_{\mathrm{best},g} - \mathrm{X}_{i,g}\right) + F_i \cdot \left(\mathrm{X}_{r1,g} - \widetilde{\mathrm{X}}_{r2,g}\right) \tag{22}$$

where $\mathrm{X}_{i,g}$, $\mathrm{X}_{r1,g}$ and $\mathrm{X}^{p}_{\mathrm{best},g}$ are selected from $\boldsymbol{P}$ in the same way as in $\mathrm{DE/current-to-}p\mathrm{best/1}$ (without archive), while $\widetilde{\mathrm{X}}_{r2,g}$ is randomly chosen from the union, $\boldsymbol{P} \cup \boldsymbol{A}$, of the current population and the archive.

The archive operation is made very simple to avoid significant computation overhead. The archive is initiated to be empty. Then, after each generation, the parent solutions that fail in the selection process are added to the archive. If the archive size exceeds a certain threshold ($|\boldsymbol{A}| \leq NP, e.g.$), then some solutions are randomly removed from the archive to keep the archive size.

At each generation $g$, the crossover probability $CR_i$ of each individual $\mathrm{X}_i$ is independently generated according to a normal distribution of mean $\mu_{CR}$ and standard deviation 0.1 and then truncated to $[0, 1]$, as shown as follows:

$$CR_i = \mathrm{randn}_i(\mu_{CR}, 0.1) \tag{23}$$

Denote $S_{CR}$ as the set of all successful crossover probabilities $CR_i$'s at generation $g$. The mean $\mu_{CR}$ is initialized to be 0.5 and then updated at the end of each generation as:

$$\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot \text{mean}_A(S_{CR}) \tag{24}$$

where $c$ is a positive constant between 0 and 1 and $\text{mean}_A(*)$ is the usual arithmetic mean.

Similarly, at each generation $g$, the mutation factor $F_i$ of each individual $\text{X}_i$ is independently generated according to a Cauchy distribution with location parameter $\mu_F$ and scale parameter 0.1 and then truncated to be 1 if $F_i \geq 1$ or regenerated if $F_i \leq 0$, as shown as follows:

$$F_i = \text{randc}_i(\mu_F, 0.1) \tag{25}$$

Denote $S_F$ as the set of all successful mutation factors in generation $g$. The location parameter $\mu_F$ of the Cauchy distribution is initialized to be 0.5 and then updated at the end of each generation as:

$$\mu_F = (1 - c) \cdot \mu_F + c \cdot \text{mean}_L(S_F) \tag{26}$$

where $\text{mean}_L(*)$ is Lehmer mean:

$$\text{mean}_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \tag{27}$$

The pseudo code of JADE is presented as follows:

**Algorithm5: Procedure of JADE with Archive**

**Begin**

Set $\mu_{CR} = 0.5$; $\mu_F = 0.5$; $\boldsymbol{A} = \emptyset$.

Create a random initial population $\{\mathrm{X}_{i,0} | i = 1, 2, \ldots, NP\}$.

**for** $g = 1$ to $G$:

$S_F = \emptyset$; $S_{CR} = \emptyset$.

**for** $i = 1$ to $NP$:

Generate $CR_i = \mathrm{randn}_i(\mu_{CR}, 0.1)$; $F_i = \mathrm{randc}_i(\mu_F, 0.1)$.

Randomly choose $\mathrm{X}^p_{\mathrm{best},g}$ as one of the $100p\%$ best vectors.

Randomly choose $\mathrm{X}_{r1,g} \neq \mathrm{X}_{i,g}$ from current population $\boldsymbol{P}$.

Randomly choose $\widetilde{\mathrm{X}}_{r2,g} \neq \mathrm{X}_{r1,g} \neq \mathrm{X}_{i,g}$ from $\boldsymbol{P} \cup \boldsymbol{A}$.

$\mathrm{V}_{i,g} = \mathrm{X}_{i,g} + F_i \cdot \left(\mathrm{X}^p_{\mathrm{best},g} - \mathrm{X}_{i,g}\right) + F_i \cdot (\mathrm{X}_{r1,g} - \widetilde{\mathrm{X}}_{r2,g})$.

Generate $j_{\mathrm{rand}} = \mathrm{randint}(1, D)$

**for** $j = 1$ to $D$:

**if** $j = j_{\mathrm{rand}}$ or $\mathrm{rand}(0,1) < CR_i$:

$\mathrm{U}^j_{i,g} = \mathrm{V}^j_{i,g}$.

**else:**

$\mathrm{U}^j_{i,g} = \mathrm{X}^j_{i,g}$.

**end if**

**end for**

**if** $f(\mathrm{X}_{i,g}) \leq f(\mathrm{U}_{i,g})$: $\mathrm{X}_{i,g+1} = \mathrm{X}_{i,g}$.

**else:** $\mathrm{X}_{i,g+1} = \mathrm{U}_{i,g}$; $\mathrm{X}_{i,g} \rightarrow \boldsymbol{A}$; $CR_i \rightarrow S_{CR}$; $F_i \rightarrow S_F$.

**end if**

Randomly remove solutions from $\boldsymbol{A}$ so that $|\boldsymbol{A}| \leq NP$.

$\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot \mathrm{mean}_A(S_{CR})$

$\mu_F = (1 - c) \cdot \mu_F + c \cdot \mathrm{mean}_L(S_F)$

**end for**

**end for**

**End**

The original JADE is a parameter-adaptive algorithm designed for single-objective optimization problems. However, when we incorporate JADE into the MOEA/D framework, a fatal issue arises with the original JADE. As mentioned earlier, in the MOEA/D framework, all evolution is conducted around the concept of neighborhoods. individuals involved in the calculation should be selected from within the neighborhood, that means $\mathrm{X}_{i,g}$, $\mathrm{X}_{\mathrm{best},g}^{p}$ and $\mathrm{X}_{r1,g}$ should be picked out from the neighborhood $B(i)$. Based on the logic of subproblem updating in MOEA/D, in the vast majority of cases, the fitness of $\mathrm{X}_{i,g}$ is optimal for the current subproblem $\lambda^{i}$, that is, vector $(\mathrm{X}_{\mathrm{best},g}^{p} - \mathrm{X}_{i,g})$ almost always do negative work on the evolution of $\mathrm{V}_{i,g}$.

To retain the advantages of JADE as much as possible and make it applicable to MOEA/D, some necessary modifications of $\mathrm{DE/current-to-}p\mathrm{best/1}$ are as follows:

$$\mathrm{V}_{i,g} = \mathrm{X}_{i,g} + F_i \cdot \left(\mathrm{X}_{i,g} - \widetilde{\mathrm{X}}_{i,g}\right) + F_i \cdot \left(\mathrm{X}_{r1,g} - \widetilde{\mathrm{X}}_{r2,g}\right) \tag{28}$$

where $\widetilde{\mathrm{X}}_{i,g}$ is randomly selected from set $\boldsymbol{E}_i$, which is an external archive store those individuals belonging to the $i$th subproblem that have survived in one of the previous generations but have been replaced until the current generation.

## 3.4 DE-IDEAL

From DE1-6 [29,30] and the mutation strategy of the IDE operator, it is evident that most operators focus on individuals from the current generation. JADE's mutation strategy involves recently explored inferior solutions, but a significant feature of JADE is the adaptive adjustment of parameters using $S_{CR}$ and $S_F$. A simple archive is used to store no more than $NP$ replaced individuals, and the methods for using and removing these replaced individuals are randomly selected. It is reasonable to believe that the utilization of these replaced individuals is limited.

After analyzing and studying several variants of the DE operator, we realize that from the object space perspective, individual evolution can be abstracted as step size and direction. Differential vectors and $F$ influence the evolution step size, while

differential vectors and $CR$ influence the evolution direction. When $CR$ is large enough (e.g., CR=0.9 in DE1), the key factor determining the evolution direction is only the differential vector itself. Compared to step size, a correct evolution direction is more important. However, a differential vector that can provide a suitable evolution direction may not necessarily appear in the current population. In other words, differential vectors that have appeared in the evolutionary history have an equally high probability of generating differential vectors suitable for the current search situation.

Therefore, based on the above logic, we propose a DE variant operator designed specifically for the MOEA/D framework called DE-IDEAL. In DE-IDEAL, we introduce the concept of the Historical Information Pool ($\boldsymbol{HIP}$) to store differential vectors that have appeared in the evolutionary process. The name DE-IDEAL is chosen because, when using vectors from $\boldsymbol{HIP}$, we rely on basic Euclidean geometry, aiming to refine the current evolutionary direction towards an ideal state.

Additionally, we aspire for DE-IDEAL to exhibit search characteristics that are entirely different from IDE and JADE. Consequently, we base the design of the DE-IDEAL operator on the mutation strategy of DE6. The mutation strategy of DE6 is defined as:

$$\text{DE6}\ \ \text{DE/current}-\text{to}\ -\text{best/1: V}_i^G = \text{X}_i^G + F\cdot(\text{X}_{best}^G - \text{X}_i^G) + F\cdot(\text{X}_{r1}^G - \text{X}_{r2}^G)$$

where $\text{X}_{best}^G$ is the best solution in $G$ generation, and $\text{X}_{r1}^G$, $\text{X}_{r2}^G$ are randomly selected from the current population $\boldsymbol{P}$. As same as mentioned in Chapter 3.2, in MOEA/D framework, $\text{X}_i^G$ always has a minimum fitness-value to $\lambda_i$ in neighborhood $B(i)$, and $\text{X}_{r1}^G$, $\text{X}_{r2}^G$ should be randomly selected from $B(i)$, the mutation strategy of DE-IDEAL should be modified as:

$$\text{V}_i^G = \text{X}_i^G + F\cdot\left(\text{X}_i^G - \text{X}_{r1}^G\right) + F'\cdot H_i \tag{29}$$

where $H_i$ is a historical vector with the form as same as $\text{X}_i^G$ or $\text{X}_{r1}^G$ in design space, and randomly selected from set $\widetilde{H}_i$ ($\widetilde{H}_i \in \boldsymbol{HIP}$).

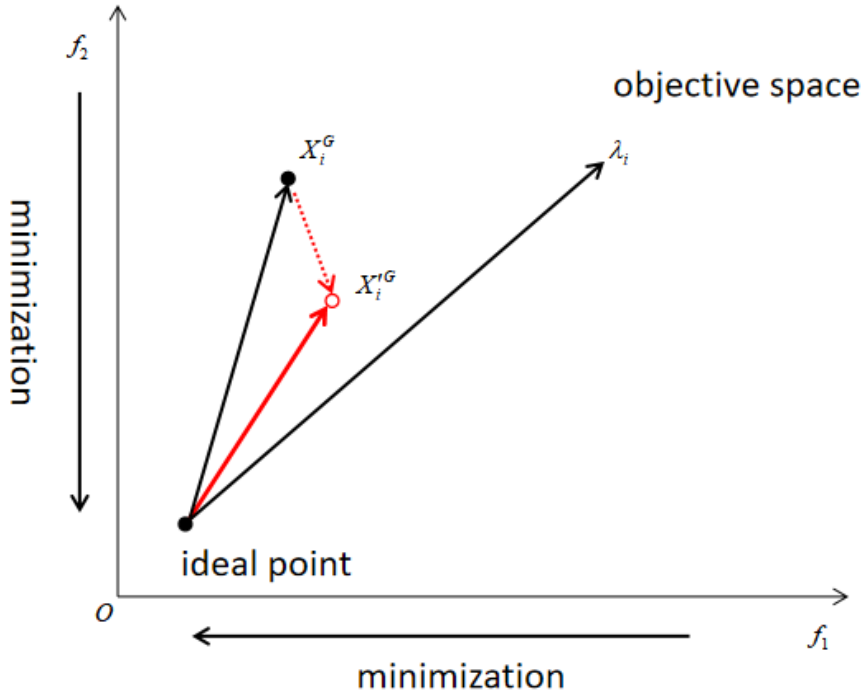


Figure 5: $\mathrm{X}_i^G$ and $\mathrm{X}'^G_i$ represent a positional relationship in the objective space, and the red dotted arrow represents the vector $\widetilde{\mathrm{V}}'$. It is important to note that vector $\mathrm{X}_i^G$, $\mathrm{X}'^G_i$ and $\boldsymbol{\lambda_i}$ here all start from ideal point.

Historical information pool (***HIP***) is used to store historical details. Specifically, when the parent individual $\mathrm{X}_i^G$, is successfully replaced by its child $\mathrm{X}_i^{G+1}$, $\widetilde{\mathrm{V}} = \mathrm{X}_i^{G+1} - \mathrm{X}_i^G$ will be append to ***HIP***. To be clear, $\widetilde{\mathrm{V}}$ is a vector with the same form as $\mathrm{X}_{r1}^G$ and $\mathrm{X}_{r2}^G$ in the design space, but it is also related to another vector $\widetilde{\mathrm{V}}' = F\left(\mathrm{X}_i^{G+1}\right) - F(\mathrm{X}_i^G)$ in the objective space. ***HIP*** can hold as many items as the population size $NP$. If the number of items in ***HIP*** reaches $NP$, the items that were first added to ***HIP*** will be removed, and the new $\widetilde{\mathrm{V}}$ will be added in. $\widetilde{H}_i$ is a subset of ***HIP***, and the corresponding $\widetilde{\mathrm{V}}'$ of $\widetilde{\mathrm{V}}$ in $\widetilde{H}_i$ should satisfy certain Euclidean geometric relations in the objective space.

Assume we generate an individual $\mathrm{X}_i^G$, and its position in the objective space is shown in Fig. Ideally, it would be alluring for individual $\mathrm{X}_i^G$ to approach the ideal point strictly in the opposite direction of the weight vector $\lambda_i$. Still, the reality is that there is usually some distance between individual $\mathrm{X}_i^G$ and he weight vector $\lambda_i$. We expect that the individual $\mathrm{X}_i^G$ to be in the position of $\mathrm{X}'^G_i$, because $\mathrm{X}'^G_i$ is closer to the weight

vector $\lambda_i$ and closer to the ideal point than $\mathrm{X}_i^G$, simultaneously. The key to getting from $\mathrm{X}_i^G$ to $\mathrm{X}'^G_i$ is to find a vector represents $\mathrm{X}'^G_i - \mathrm{X}_i^G$. Let $\widetilde{\mathrm{V}}' = \mathrm{X}'^G_i - \mathrm{X}_i^G$, $\widetilde{\mathrm{V}}'$ must satisfy two conditions:

- Based on Euclidean geometry, $\angle VIW$ (the angle from $-\widetilde{\mathrm{V}}'$ to $\lambda_i$) must bigger than $\angle XIW$ (the angle from $\mathrm{X}_i^G$ to $\lambda_i$).
- $\angle VIX$ (the angle from $-\widetilde{\mathrm{V}}'$ to $\mathrm{X}_i^G$) must be less than a threshold $\theta'$. Because an oversize $\angle VIX$ may still lead $\mathrm{X}'^G_i$ closer to the weight vector $\lambda_i$ but take $\mathrm{X}'^G_i$ further away from the ideal point.

We subjectively relate $\theta'$ to $\theta$ as $\theta' = \arctan\frac{1}{\theta}$. In addition, according to the geometric relationship shown in Figure 5, the scaling factor $F'$ should not be too big. Because an oversize $F'$ also lead $\mathrm{X}'^G_i$ further away from both weight vector $\lambda_i$ and the ideal point. In general, $F'$ should be smaller then $F$ ($F' = 0.2F$, e.g.).

The pseudo code of DE-IDEAL is presented as follows:

**Algorithm6: Procedure of DE-IDEAL**

**Begin**

set $\boldsymbol{HIP} = \emptyset$; Calculate $\theta' = \arctan\frac{1}{\theta}$.

Create a random initial population $\{\mathrm{X}_{i,0} | i = 1, 2, \ldots, NP\}$.

**for** $g = 1$ to $G$:

  **for** $i = 1$ to $NP$:

    Randomly choose $\mathrm{X}_{r1}^G \neq \mathrm{X}_i^G$ from $B(i)$.

    **if** $\boldsymbol{HIP} = \emptyset$:

      $\mathrm{V}_i^G = \mathrm{X}_i^G + F \cdot (\mathrm{X}_i^G - \mathrm{X}_{r1}^G)$.

    **else:**

      set $\widetilde{H}_i = \emptyset$; get current weight vector $\lambda_i$.

      Calculate $\angle XIW$.

      **for** $k$ in $\boldsymbol{HIP}$:

        Calculate $\angle VIX$; Calculate $\angle VIW$.

**If** $\angle VIX < \theta'$ && $\angle VIW \geq \angle XIW$:

$k \rightarrow \widetilde{H}_i$.

**end if**

**end for**

**if** $\widetilde{H}_i == \emptyset$:

$\mathrm{V}_i^G = \mathrm{X}_i^G + F \cdot (\mathrm{X}_i^G - \mathrm{X}_{r1}^G)$.

**else:**

Randomly choose $H_i$ from $\widetilde{H}_i$.

$\mathrm{V}_i^G = \mathrm{X}_i^G + F \cdot (\mathrm{X}_i^G - \mathrm{X}_{r1}^G) + F' \cdot H_i$

**end if**

**end if**

Generate $j_{\mathrm{rand}} = \mathrm{randint}(1, D)$

**for** $j = 1$ to $D$:

**if** $j = j_{\mathrm{rand}}$ or $\mathrm{rand}(0,1) < CR_i$:

$\mathrm{U}_{i,g}^j = \mathrm{V}_{i,g}^j$.

**else:**

$\mathrm{U}_{i,g}^j = \mathrm{X}_{i,g}^j$.

**end if**

**end for**

**if** $f(\mathrm{X}_{i,g}) \leq f(\mathrm{U}_{i,g})$:

$\mathrm{X}_{i,g+1} = \mathrm{X}_{i,g}$.

**else:**

$\mathrm{X}_{i,g+1} = \mathrm{U}_{i,g}$; $(\mathrm{X}_{i,g+1} - \mathrm{X}_{i,g}) \rightarrow \boldsymbol{HIP}$.

**end if**

remove earliest item appended into $\boldsymbol{HIP}$ so that $|\boldsymbol{HIP}| \leq NP$.

**end for**

**end for**

**End**

## 3.4 Covariance matrix adaptation evolution strategy (CMA-ES) in MOEA/D

Differing from IDE, JADE, and DE-IDEAL, CMA-ES is an operator based on estimation of distribution and has a process similar to **Algorithm3.** CMA-ES incorporates several key innovations: It utilizes a covariance matrix C to track dependencies between two samples. This covariance matrix is a diagonal matrix and can be entirely defined by a set of standard orthogonal bases and a set of eigenvalues $\lambda$. The update step is controlled by $\sigma$. The direction of mean update aims to maximize the probability values of the last successful candidates, resembling a natural gradient descent. Two paths are recorded: evolution path $p_c$ primarily stores the correlation between consecutive step size for computing a well-defined correlation matrix, while conjugate evolution path $p_\sigma$ is used to control the step size.

CMA-ES addresses the dependencies and scaling between variables in the normal distribution $\mathcal{N}\left(m^{(g)}, \left(\sigma^{(g)}\right)^2 \mathrm{C}^{(g)}\right)$ by adjusting the covariance matrix C. Its core lies in how to fine-tune these parameters, especially the step size parameter and covariance matrix, to achieve the best possible search performance. The fundamental approach of parameter adjustment in CMA-ES is to gradually increase the probability of generating good solutions (increasing the probability of searching along favorable directions).

In CMA-ES, a population of new search points is generated by sampling a multivariate normal distribution. The basic equation for sampling the search points in generation g+1 is expressed as (Hansen, 2006) [37]:

$$\mathrm{x}_k^{(g+1)} \sim \mathcal{N}\left(m^{(g)}, \left(\sigma^{(g)}\right)^2 \mathrm{C}^{(g)}\right) \quad (30)$$

$$\text{for } k = 1, \ldots, \lambda$$

where $\sim$ denotes the same distribution on the left and right side, $\mathrm{x}_k^{(g+1)} \in \mathbb{R}^n$ denotes the $k$th offspring (search point) in generation $g+1$, $m^{(g)} \in \mathbb{R}^n$ denotes the mean value of the search distribution at generation $g$, $\mathrm{C}^{(g)} \in \mathbb{R}^{n \times n}$ denotes the covariance

matrix at generation $g$. $\lambda = 4 + 3\lfloor \ln D \rfloor$ for the population size, sample size, and number of offspring.

$N\left(m^{(g)}, \left(\sigma^{(g)}\right)^2 \mathrm{C}^{(g)}\right) \sim m^{(g)} + \sigma^{(g)} N\left(0, C^{(g)}\right) \sim m^{(g)} + \sigma^{(g)} B^{(g)} D^{(g)} N(0, I)$ is the multi-variate normal search distribution.

In the MOEA/D framework, all evolution centers around the neighborhood $B(i)$. Therefore, when employing the CMA-ES algorithm to generate new individuals in MOEA/D, the initial step involves sorting the individuals within $B(i)$ based on their fitness-value to the current subproblem as:

$$g^{bip}\left(x'_1 \middle| \lambda^i, \mathrm{z}\right) \leq g^{bip}\left(x'_2 \middle| \lambda^i, \mathrm{z}\right), \ldots, g^{bip}\left(x'_T \middle| \lambda^i, \mathrm{z}\right) \tag{31}$$

It is essential to emphasize that in this context, the $\lambda^i$ represents the current subproblem in MOEA/D, and it differs from the $\lambda$ in CMA-ES, which represents the number of sampling solutions. $\mathbf{X} = (x'_1, x'_2, \ldots, x'_T)$ represents the new set formed by sorting individuals in $B(i)$, from which the top $\mu = \left\lfloor \frac{\lambda}{2} \right\rfloor$ solutions are selected to update the distribution parameters of CMA-ES.

The mean value of the distribution is the weighted maximum likelihood estimate of the selected $\mu$ solutions. Its update formula is as follows:

$$m^{(g+1)} = \sum_{i=1}^{u} \omega_i \, x^g_{i:|\mathbf{X}|} = m^{(g)} + \sum_{i=1}^{u} \omega_i \left( x^g_{i:|\mathbf{X}|} - m^{(g)} \right) \tag{32}$$

The practice of obtaining the next generation's mean by combining the selected partial solutions here is referred to as multi-recombination, analogous to the crossover strategy DE operator, which involves exchanging information between different solutions. If the weight coefficients ($\omega_1$, $\omega_2$,..., $\omega_\mu$) are all set to the same value $\frac{1}{\mu}$, then this formula represents the maximum likelihood estimate using the selected solutions. Alternatively, choosing $\omega_1 \geq \omega_2 \geq, \ldots, \omega_\mu > 0$ emphasize the solutions ranked at the front [45].

In CMA-ES, the evolution path $p_c$ constructed using historical search information is crucial for the update of the covariance matrix $C$. The method for constructing $p_c$ is as follows:

$$p_c^{(g+1)} = (1 - c_c)p_c^{(g)} + \sqrt{c_c(2 - c_c)}\sqrt{\mu_\omega}\frac{m^{(g+1)} - m^{(g)}}{\sigma^{(g)}} \tag{33}$$

The above formula describes the movement of the distribution mean value, and it takes a weighted average of the movement direction $\frac{m^{(g+1)}-m^{(g)}}{\sigma^{(g)}}$ in each iteration. This weighting ensures the cancellation of opposing directional components and the accumulation of similar components. This process is akin to the commonly used Momentum in neural network optimization. Therefore, the evolution path represents one of the best search directions. The design of term $\mu_\omega = (\sum_{i=0}^{\mu}(\omega_i{}^2))^{-1}$ is based on $\sqrt{\mu_\omega}\frac{m^{(g+1)}-m^{(g)}}{\sigma^{(g)}} \sim N(0, \mathrm{C}^{(g)})$ [45], which can be considered as a random vector sampled from the distribution described above. The design of term $\sqrt{c_c(2 - c_c)}$ is based on $(1 - c_c)^2 + (\sqrt{c_c(2 - c_c)})^2 = 1$ [45], where learning rate $c_c = \frac{4}{n+4}$ is inversely proportional to the adjusted degrees of freedom (number of parameters). $\mu_\omega$ and $\sqrt{c_c(2 - c_c)}$ are referred to as the stationarity condition.

The principle of updating the covariance matrix C is as follows:

$$argmax\ p_c^{(g)}\left(p_c^{(g+1)}\middle|m, \mathrm{C}\right) \tag{34}$$

$$argmax\ \prod_{i=1}^{\mu} p_c^{(g)}\left(\frac{x_{i:|\mathbf{X}|}^{g} - m^{(g)}}{\sigma^{(g)}}\middle|m, \mathrm{C}\right) \tag{35}$$

Based on this, the updating method for the covariance is as follows:

$$\mathrm{C}^{(g+1)} = \left(1 - c_1 - c_\mu + c_c\right)\mathrm{C}^{(g)} + c_1 p_c^{(g)}\left(p_c^{(g)}\right)^T + c_\mu \sum_{i=1}^{\mu} \omega_i \frac{x_{i:|\mathbf{X}|}^{g} - m^{(g)}}{\sigma^{(g)}}\left(\frac{x_{i:|\mathbf{X}|}^{g} - m^{(g)}}{\sigma^{(g)}}\right)^T \tag{36}$$

where learning rate $c_1 = \frac{2+\min(1,\lambda/6)}{(D+1.3)^2\mu_\omega}$, $c_\mu = \min\left(1 - c_1, \sqrt{\frac{2(\mu_\omega - 2 + \mu_\omega{}^{-1})}{(D+2)^2+\mu_\omega}}\right)$.

CMA-ES defaults to using Cumulative Step Size Adaptation (CSA), which is currently the most successful and widely used step size adjustment method. The core principle of CSA can be understood as the successive search directions should be conjugate. That is, if the successive search directions are positively correlated (angle less than

$\pi/2$), it indicates that the step size is too small and should be increased. If the successive search directions are negatively correlated, it indicates that the step size is too large and should be decreased. Similar to the previous evolution path $p_c$, the method for constructing the conjugate evolution path is as follows:

$$p_\sigma^{(g+1)} = (1 - c_\sigma)p_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_\omega}\mathrm{C}^{(g)^{-\frac{1}{2}}}\sum_{i=1}^{u}\omega_i\frac{x_{i:|\mathbf{X}|}^{g} - m^{(g)}}{\sigma^{(g)}} \tag{37}$$

The update formula based on $p_\sigma$ and step size $\sigma$ should be:

$$\sigma^{(g+1)} = \sigma^{(g)} \times \exp\left(\min\left(1, \frac{c_\sigma}{d_\sigma}\left(\frac{\left\|p_\sigma^{(g+1)}\right\|}{E\left\|\mathcal{N}(0,\mathrm{I})\right\|} - 1\right)\right)\right) \tag{38}$$

where $c_\sigma = \frac{\mu_\omega + 2}{D + \mu_\omega + 3}$ and $d_\sigma = 1 + c_\sigma + 2\max\left(0, \sqrt{\frac{\mu_\omega - 1}{D+1}} - 1\right)$are control parameters that adjust the magnitude of step size variations.

# Chapter 4 Ensemble Framework based on MOEA/D

## 4.1 Introduction

As one of the current state-of-the-art multi-objective evolutionary algorithms, Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [20] provides a framework for solving MOPs. In MOEA/D, a set of evenly weight vectors $\lambda^1, \lambda^2, \ldots, \lambda^{NP}$ in the objective space is used to decompose one MOP into multiple subproblems. For each of the subproblems, scalarization function (such as Weighted Sum Approach, Tchebycheff Approach [46] or PBI [20,28]) and weight vectors are used to coordinate each objective function's relationship, and each subproblem can be considered as a scalar optimization problem. Since that, it is natural to use genetic evolution operators those were originally designed to solve single-objective optimization problem such as the Differential Evolution (DE) and its variants to generate new solutions. The details regarding the MOEA/D framework and the DE operator have been thoroughly described in Chapter 2.

Due to different mutation strategies, the DE algorithm and its variants exhibit different search capabilities, which can be described as exploitation and exploration. Exploration is the process of visiting entirely new regions of a search space, while exploitation is visiting those regions within the neighborhood of previously visited points [21]. However, according to the ‘‘No Free Lunch’’ theory, it is difficult for one new solution generating operator to exhibit both exploitation and exploration capabilities. More specifically, different iteration stages have obvious tendency to the search characteristics. It is easy to imagine that at the beginning of the iteration, new solution generating operator with exploration capability will be more likely to achieve good performance, because most of the objective space is still untouched; Instead, at the ending of the iteration, new solution generating operator with exploitation capability will be more likely to achieve good performance.

In order to efficiently solve the optimization problem through evolutionary computation, it is necessary to include new solution generating operators with

different search capabilities in the framework and adjust them in time for different problems and iteration stages. Therefore, in this chapter, we propose Ensemble Framework based on MOEA/D which named as MOEA/D-EF for short. MOEA/D-EF can contain a variety of new solution generating operators with different search capabilities and has a mechanism to switch them by the current search situation. Compared with other algorithms with fixed search capability, MOEA/D-EF can adapt to various problems and has better universality.

Some other researchers inspired this study. For example, in HMJCDE [47], the author constructed a hybrid framework based on modified CoDE [48] and JADE [35] to achieve a similar purpose to our study; Also, in the MVC framework [49], the author systematically divides the iterative process into learning generation and executing generation. The new solutions generated by three different candidate operators are compared in the learning generation. The algorithm that develops superior solutions is selected in the following executing generation. The essence of learning generation in the MVC framework is to evaluate the performance of three different candidate operators.

We inherit part of the idea in MVC framework and provide fairer environment for evaluating multiple new solution generating operators in the Evaluation Generation ($EG$). However, most similar attempts, including the above model, are designed for single-objective optimization problems. For single-objective optimization problems, when evaluating the advantages and disadvantages of a single-objective optimization algorithm, a simple comparison of improved difference is enough; On the contrary, one of the critical problems of multi-objective optimization is how to balance the relationship between multiple objective functions. Even though the MOEA/D framework can import modified single-objective optimization algorithms, evaluation of a multi-objective optimization algorithm needs to use a comprehensive indicator. Thus, the Hypervolume [] indicator, widely used in solving MOPs, is introduced for switching candidate algorithms in our approach. The candidate operator that achieves a higher cumulative Hypervolume value in one $EG$ will be considered

superior and will be executed in the following Implementation Generation ($IG$).

## 4.2 Algorithm

Exploration and exploitation are two basic search capabilities for problem solving. Exploration is the process of visiting entirely new regions of a search space, while exploitation is the process of visiting those regions of a search space within the neighborhood of previously visited points [21]. Different DE mutation strategies possess different characteristics, and each of them shows distinct tradeoff between exploration and exploitation (TEE) [31,44,48]. The diagram illustrates the relative relationships of some mutation strategies on TEE as shown as in Figure 6:

In order to make MOEA/D-EF as versatile as possible for various MOPs, it is essential that the new solution generation operators in MOEA/D-EF possess distinct TEE characteristics. Therefore, we select DE1, JADE, and DE-IDEAL as candidate operators. DE1 tends to emphasize exploration characteristics, DE-IDEAL leans more towards exploitation characteristics, while JADE serves as a balanced intermediary between the two.

In MOEA/D-EF, we divide all generations into two parts, the Evaluation generation ($EG$) and the Implementation generation ($IG$). $EG$ and $IG$ match each other, and multiple $EG - IG$ pairs will be presented in the whole iteration process. $EG$ aims to evaluate the performance of different new solution generating operators (candidate operator) fairly by providing the same initial environment. However, it leads $EG$ to consume several times the evaluation cost of its corresponding $IG$. Therefore, $EG$ is usually set to have a small capacity. In this study, all $EG$s including the initial $EG_1$ were set to contain 5 generations, and the initial $IG_1$ was set to contain 10 generations. The capacity of the following $IG_k|_{k>1}$ has the capacity as $|IG_k| = 2|IG_{k-1}|_{k>1}|$ until $|IG_k| \geq 40$.

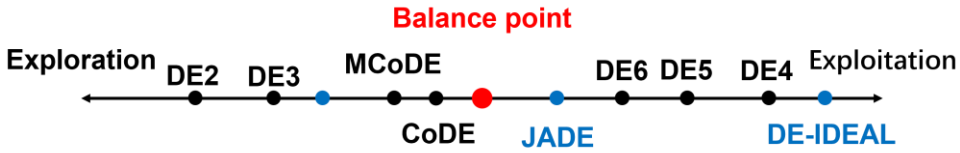


Figure 6 TEE of DE mutation strategies and related algorithms.

In $EG$, we first copy the population at current generation $g$ into $q$ copies（in this paper $q = 3$）and assign the identical population as an initial population to each candidate operator. The candidate operators evolve independently by using copies until the last generation of the current $EG$. At the end of the current $EG$, we evaluate the performance of each candidate operator based on the Hypervolume value $HV_j^g|_{1\le j\le q}$. The best-performing operator will be executed in the subsequent $IG$. Also, at the end of the current $EG$, each subproblem has $q$ solutions generated by candidate operators. For each subproblem, the optimal individuals are selected based on fitness value to form the initial population for the upcoming $IG$. MOEA/D-EF works as follows:

**Input:**

- MOP.
- A stopping criterion.
- $NP$: the number of the subproblems considered in MOEA/D.
- A uniform spread of $N$ weight vectors: $\lambda^1, \ldots, \lambda^{NP}$ .
- $T$: the number of the weight vectors in the neighborhood of each weight vector.
- $q$ candidate operators: DE1, JADE, DE-IDEAL.
- An external population (EP), which is used to store non-dominated solutions found during the search.

**Output:** $EP$

**Algorithm7: MOEA/D-EF with candidate operators: DE1, JADE, DE-IDEAL**

**Step 1) Initialization:**

**Step 1.1)** Set $EP$=∅.

**Step 1.2)** Compute the Euclidean distances between any two weight vectors and then work out the $T$ closest weight vectors to each weight vector. For each $i = 1, \dots, NP$, set $B(i) = \{i_1, \dots, i_T\}$, where $\lambda^{i_1}, \dots, \lambda^{i_T}$ are the $T$ closest weight vectors to $\lambda^i$.

**Step 1.3)** Generate an initial population $x^1, \dots, x^{NP}$ randomly or by a problem-specific method.

**Step 1.4)** Initialize reference point $\mathrm{z} = (z_1, \dots, z_m)^T$.

**Step 1.5)** Initialize the capacity of $EG$ and $IG$. Set the first generation $g_1$ as the beginning of $EG_1$

**Step 2) Switch:**

**if** the current generation $g$ is the beginning of an $EG$:

Copy current population $\boldsymbol{P}$, set $\boldsymbol{P}_1 = \boldsymbol{P}_1 = \boldsymbol{P}_1 = \boldsymbol{P}$, assign $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, and $\boldsymbol{P}_3$ to DE1, JADE and DE-IDEAL operator, respectively.

then **go Step 3)**.

**else if** the current generation $g$ is in EG:

**go Step 3)**.

**else if** the current generation $g$ is the end of an $EG$:

**go Step 4)**.

**else:**

**go Step 3)**.

**Step 3) Update:**

**for** $i = 1, \dots, NP$:

**Step 3.1) Reproduction:**

**if** the current generation $g$ is not the end of an $EG$:

generate $\mathrm{X}_{i,1}^{g+1}$, $\mathrm{X}_{i,2}^{g+1}$ and $\mathrm{X}_{i,3}^{g+1}$ through the mutation strategies of DE1, JADE and DE-IDEAL, respectively.

**else if** he current generation $g$ is in an $IG$:

generate $\mathrm{X}_{i,R}^{g+1}$, which represents the individual generated by the regnant

candidate operator determined in the end of last $EG$.

**Step 3.2) Update of reference point** $\mathrm{z} = (z_1, z_2, \dots, z_j, \dots, z_m)$:

**if** $\mathrm{z}_j > \min\left(f_j(\mathrm{X}_{i,1}^{g+1}), f_j(\mathrm{X}_{i,2}^{g+1}), f_j(\mathrm{X}_{i,3}^{g+1})\right)$:

$\mathrm{z}_j = \min\left(f_j(\mathrm{X}_{i,1}^{g+1}), f_j(\mathrm{X}_{i,2}^{g+1}), f_j(\mathrm{X}_{i,3}^{g+1})\right)$.

$\mathrm{X}_{i,R}^{g+1}$ in $IG$ has the similar way to update the reference point.

**Step 3.3) Update of Neighboring Solutions:**

Depends on the operator itself

**Step 3.4) Update of $\boldsymbol{EP}$:** As same as Step 2.5 in **Algorithm1**.

**Step 4) End of $\boldsymbol{EG}$:**

**Step 4.1) Reproduction:**

generate $\mathrm{X}_{i,1}^{g+1}$, $\mathrm{X}_{i,2}^{g+1}$ and $\mathrm{X}_{i,3}^{g+1}$ as the same way as Step 3.1).

**Step 4.2) Calculation of Hypervolume value:**

Calculate $HV_1^{g+1}$, $HV_2^{g+1}$ and $HV_3^{g+1}$ represent the Hypervolume values of DE1, JADE and DE-IDEAL, respectively.

**Step 4.3) Determine candidate operators:**

Regnant operator $R = \text{index of} \max\left(HV_1^{g+1}, HV_2^{g+1}, HV_3^{g+1}\right)$.

**Step 4.4) Population merging:**

for $i = 1, \dots, NP$:

$\boldsymbol{P}_i^{g+1} = \min\left(\mathrm{X}_{i,1}^{g+1}, \mathrm{X}_{i,2}^{g+1}, \mathrm{X}_{i,3}^{g+1}\right)$

**Step 4.5) Update of $\boldsymbol{EP}$:** As same as Step 2.5 in **Algorithm1**.

**Step 5) Stopping Criteria:** As same as Step 3 in **Algorithm1**.

It should be added that even if the regnant algorithm being executed in IG is not DE-IDEAL or JADE, the historical information pool of DE-IDEAL and JADE's adaptive parameters should be continuously updated in their respective ways.

## 4.3 Numerical experiments

### 4.3.1 Experimental conditions

As described above, the proposed framework MOEA/D-EF contains three new solutions generation operators: DE1, JADE and DE-IDEAL. In this section, comprehensive experiments are conducted to evaluate our new framework and compare the effectiveness of MOEA/D-EF with the algorithms that only use pure DE1, JADE and DE-IDEAL operator as the new solutions generation operator. Since Hypervolume values were used as the switching indicator in the iteration of MOEA/D-EF, the IGD value was selected as the evaluation indicator in the comprehensive experiment. IGD value is widely used in the MOPs. field because it is easy to be calculated and can directly show the convergence of the checked approach. Three-objective WFG series problems [55] with 30,50 and 100 design variables are taken as the test instances. Our motivation is to make the MOEA/D-EF framework perform as well as possible in the face of different problems. Therefore, in addition to the conventional numerical analysis, we discuss the universality of the new algorithm utilizing horizontal comparison.

For each approach, the subproblem number $NP$ is set to 300, and the neighborhood size $T$ is set as 21. As introduced in Chapter 2, the scalarization function PBI is imported to decompose a MOP. The preset penalty parameter $\theta = 5$. For DE1 and DE-IDEAL, the scaling factor $F$ crossover rate $CR$ are set as 0.5 and 0.9, respectively. The unique scaling factor $F'$ of DE-IDEAL must be less than $F$. Therefore, $F'$ is set as 0.1. As the definition given in Chapter 3, the $\angle VIX$ in DE-IDEAL must be less than a threshold $\theta'$, we subjectively relate $\theta'$ to $\theta$ as $\theta = \arctan\frac{1}{\theta}$.

For JADE, the process of generating adaptive parameters $F_i$ in $CR_i$ is the same as that of the original JADE, with the only difference being that the initial $F_m$ and $CR_m$ are randomly generated in the interval $[0.3,0.7]$, whereas the initial $F_m$ and $CR_m$ in the original JADE are both set to $0.5$. The switching indicator Hypervolume in MOEA/D-EF

is set with the Reference point [5,5,5].

Numerical experiments with 30,50 and 100 design vectors were repeated 21 times, respectively. The result of IGD values is shown in Table x to x, where MOEA/D-EF is called EF and DE-IDEAL is called IDEAL for short, respectively; S300V30 represents each approach contains 300 sub-problems and 30 design variables e.g. The **AV** column represents the average IGD value, and the **R** column represents the ranking of the approach in the current problem. The **R** column in the last row represents the average score of the ranking, and obviously, the smaller the average score, the better the approach performs.

### 4.3.2 Experimental results

As shown from Table 2 to 4, DE-IDEAL always takes the lead in WFG4,7,8. JADE takes the lead in WFG5,6 but always performs worst in other problems. MOEA/D-EF never took the lead on any problems but never had a worst-case performance. DE-IDEAL had the best overall performance in IGD value and ranking score. With the number of design variables increases to 100, the difference between those two approaches became very limited. To some extent, this result supports our conjecture about the new solution generating operator using historical information.

Table 2: Average IGD value and Ranking Score for WFG problems with 30 design variables.

| **S300 V30** | **EF** | | **DE1** | | **IDEAL** | | **JADE** | |
|---|---|---|---|---|---|---|---|---|
| | **AV** | **R** | **AV** | **R** | **AV** | **R** | **AV** | **R** |
| **WFG1** | 1.4676 | 3 | **1.4607** | **1** | 1.4613 | 2 | 1.4814 | 4 |
| **WFG3** | 0.1679 | 2 | **0.1566** | **1** | 0.1723 | 3 | 0.6122 | 4 |
| **WFG4** | 0.3100 | 2 | 0.3239 | 3 | **0.2899** | **1** | 0.4765 | 4 |
| **WFG5** | 0.1821 | 2 | 0.2253 | 4 | 0.1877 | 3 | **0.1760** | **1** |
| **WFG6** | 0.1940 | 2 | 0.2518 | 4 | 0.2118 | 3 | **0.1885** | **1** |
| **WFG7** | 0.3562 | 3 | 0.3553 | 2 | **0.3495** | **1** | 0.5983 | 4 |
| **WFG8** | 0.4414 | 3 | 0.4440 | 2 | **0.4201** | **1** | 0.6483 | 4 |
| **WFG9** | 0.1991 | 2 | 0.2489 | 4 | 0.2202 | 3 | **0.1916** | **1** |
| | | 2.4 | | 2.8 | | **2.1** | | 2.9 |

Table 3： Average IGD value and Ranking Score for WFG problems with 50 design variables.

| **S300 V50** | **EF** | | **DE1** | | **IDEAL** | | **JADE** | |
|---|---|---|---|---|---|---|---|---|
| | **AV** | **R** | **AV** | **R** | **AV** | **R** | **AV** | **R** |
| WFG1 | 1.4676 | 3 | 1.4663 | 2 | **1.4635** | **1** | 1.4925 | 4 |
| WFG3 | 0.2628 | 2 | 0.2880 | 3 | **0.2617** | **1** | 0.7330 | 4 |
| WFG4 | 0.3375 | 2 | 0.3627 | 3 | **0.3177** | **1** | 0.5045 | 4 |
| WFG5 | 0.1896 | 2 | 0.2531 | 4 | 0.1903 | 3 | **0.1814** | **1** |
| WFG6 | 0.1730 | 2 | 0.2827 | 4 | 0.2090 | 3 | **0.1680** | **1** |
| WFG7 | 0.4240 | 2 | 0.4410 | 3 | **0.4060** | **1** | 0.7042 | 4 |
| WFG8 | 0.4794 | 2 | 0.4982 | 3 | **0.4702** | **1** | 0.7507 | 4 |
| WFG9 | 0.1842 | 2 | 0.2906 | 4 | 0.2052 | 3 | **0.1763** | **1** |
| | | 2.1 | | 3.3 | | **1.8** | | 2.9 |

Table 4：Average IGD value and Ranking Score for WFG problems with 100 design variables.

| **S300 V100** | **EF** | | **DE1** | | **IDEAL** | | **JADE** | |
|---|---|---|---|---|---|---|---|---|
| | **AV** | **R** | **AV** | **R** | **AV** | **R** | **AV** | **R** |
| WFG1 | 1.4760 | 3 | **1.4684** | **1** | 1.4688 | 2 | 1.4965 | 4 |
| WFG3 | 0.3995 | 2 | 0.4166 | 3 | **0.3917** | **1** | 0.8600 | 4 |
| WFG4 | 0.3622 | 2 | 0.4094 | 3 | **0.3447** | **1** | 0.5995 | 4 |
| WFG5 | 0.1922 | 2 | 0.1961 | 3 | 0.1984 | 4 | **0.1865** | **1** |
| WFG6 | 0.1811 | 2 | 0.3307 | 4 | 0.2051 | 3 | **0.1652** | **1** |
| WFG7 | 0.4796 | 2 | 0.5117 | 3 | **0.4560** | **1** | 0.7669 | 4 |
| WFG8 | 0.5199 | 2 | 0.5470 | 3 | **0.4956** | **1** | 0.8012 | 4 |
| WFG9 | 0.1776 | 2 | 0.3238 | 4 | 0.2034 | 3 | **0.1687** | **1** |
| | | 2.1 | | 3 | | **2** | | 2.9 |

Although the average IGD value and ranking score intuitively reflect the performance of each approach on specific problems, they cannot intuitively reflect the universality of a certain approach. Universality represents a spirit with equilibrium that does not require the approach to be dominant in a particular problem but requires the approach should not perform poorly in any situation. Based on the above thinking, we made a horizontal comparison of the four different approaches` IGD value. The specific evaluation methods are as follows:

$$E_j|_{1\le j\le 4} = \sum_{i=1}^{9}\left(\frac{IGD_j^i}{IGD_{\text{mean}}^i} - IGD_{\text{mean}}^i\right) \tag{39}$$

$$IGD_{\text{mean}}^i = \frac{\sum_{i=1}^{9} IGD_j^i}{4} \tag{40}$$

where $j$ is the index represents MOEA/D-EF, DE1, JADE and DE-IDEAL, respectively. $i$ is the index of WFG series problems. $\frac{IGD_j^i}{IGD_{\text{mean}}^i}$ scales the average IGD values of each approach in the same problem to the same scale, The result of the summation $E_j$, is the final universality degree of algorithms. The results are listed in Table x, according to the definition given above, a negative $E_j$ indicates that this algorithm performs

better than the average of all. The smaller the value of $E_j$, the more obvious the advantage of this algorithm. From what has been discussed above, it can be seen that MOEA/D-EF has obvious advantages.

## 4.4 Summary

In this study, we propose a new hybrid model based on the MOEA/D framework for addressing multi-objective optimization problems, abbreviated as MOEA/D-EF. In comparison to existing frameworks with similar motivations, our novel research introduces the following innovations:

- We extend the algorithmic framework and evolutionary operators originally designed for solving single-objective optimization problems to address multi-objective optimization problems, with adaptive modifications specifically tailored to the MOEA/D framework.
- We designed an evolution operator adaptive switching mechanism based on prior knowledge. In contrast to traditional methods such as mathematical induction, we utilize performance metrics to assess the current search status, enabling the adaptive selection of evolution operators.
- Unlike a simple comparison of algorithm performance on a specific problem, we propose a more comprehensive approach to evaluate algorithm adaptability. Assessing the overall performance of different algorithms across an entire test suite provides a better indication of the algorithm's general applicability.

The results indicate that the new MOEA/D-EF algorithm, while not consistently outperforming all other algorithms on any individual test problem within the WFG test suite, generally exhibits outstanding performance, often ranking second across various scenarios. In contrast, the comparison algorithms may excel on certain problems but demonstrate poor performance on others. This outcome reaffirms the "No Free Lunch" theorem and underscores the importance of algorithm generality.

# Chapter 5 Hyper-Heuristic Multi-Objective Optimization Approach Based on MOEA/D Framework

## 5.1 Introduction

The multi-objective evolutionary algorithm based on decomposition (MOEA/D, Zhang and Li, 2007) [20] is a state-of-the-art algorithm that provides a framework for solving multi-objective optimization problems (MOPs, Coello and Lamont, 2004; Deb, K., 2011) [2,4]. Each subproblem is treated as a scalar optimization problem, where a scalarization function (e.g., the weighted-sum or Tchebycheff approach proposed by Miettinen (1999)) [50] and weight vectors are utilized to coordinate the relationships among the objective functions. Therefore, it is reasonable to utilize genetic evolution algorithms originally designed to solve single-objective optimization problems (SOPs), especially crossover (CX)-based algorithms such as the differential evolution (DE) and its variants [11,12,14], to generate new solutions.

Hansen et al. (2003) acknowledged that the evolution strategy with covariance matrix adaptation (CMA-ES) [36] was an effective algorithm based on the estimation of distribution (ED) (Springer Science & Business Media, 2001) [51] that has been widely applied in various domains. CMA-ES enhances the search process by estimating a more promising region utilizing the distribution information of the current population. Although CMA-ES was originally utilized for single-objective optimization, it can be applied to solve MOPs, as demonstrated by Igel et al. (2007) and Zapotecas-Mart'ınez et al. (2015) [52, 53].

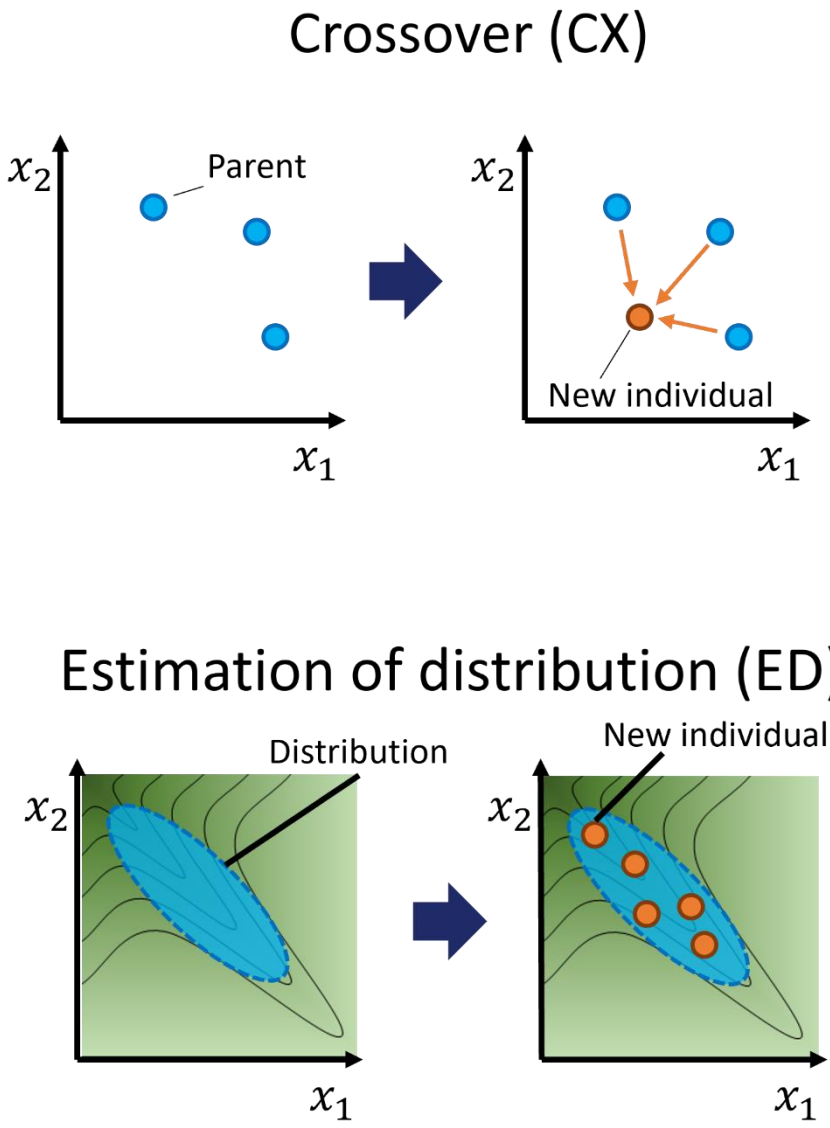


Figure 7 The Conceptual comparison of generating new individuals based on CX strategy and ED strategy.

As a further comparison between the CX and ED strategies, it was observed that CX was efficient and effective for a global search but tended to become stagnant and was not well suited for problems with strong variable dependence (non-separable problems). Conversely, ED was effective for problems with dependent variables but incurred a huge evaluation cost. he concepts of these two different strategies are illustrated in Figure 7.

DE and its variants use crossover (CX) and mutation strategies to generate new solutions. In particular, CX is recognized as an efficient approach for creating promising individuals by combining information from two or more existing

individuals (parents).

Generally, new individuals (children) are compared to one of their parents, and the children replace their parents if they demonstrate better evaluation values. This is referred to as a successful replacement scenario. Similar to but different from the feasibility ratio (FR) [54], the overall successful replacement rate (SRR) is expressed as follows:

$$\mathrm{SRR} = \frac{\text{successful replacement times}}{\max \text{evaluation times}} \tag{41}$$

Table 5 presents the average SRRs in two-, three-, and five-objective WFG [57] problems. The numerical experiment was initialized with 300 subproblems and 30 variables (1000 sub-problems and 32 variables in WFG_5D problems). The iterations were repeated 21 times. The results of numerical experiments revealed that the SRR in pure MOEA/D-DE exhibited a low level of efficiency for the WFG test suite. That is, a significant portion of the evaluation cost did not contribute to the final solution. Also, there is an inherent problem when using ED strategies especially CMA-ES to solve MOPs that significantly increases the evaluation cost. This is because many sampling points are required to obtain the probability distribution in the objective space. Based on the above analysis, a hybrid approach was proposed in this study, called MOEA/D-HH for short, which aiming to dynamically switch between different generating operators based on the search situation within the MOEA/D framework, with the core concept of 'recycling- redistribute'.

Table 5: Average SRRs in two-, three-, and five-objective WFG problems (where # indicates the number of subsequent hyperparameter). The numerical experiment was initialized with 300 subproblems and 30 variables (1000 subproblems and 32 variables in WFG_5D problems). The iterations were repeated 21 times.

| **# Objectives ($M$)** / **Problems** | **Two** | **Three** | **Five** |
|---|---|---|---|
| WFG1 | 0.1833 | 0.0963 | 0.2120 |
| WFG2 | 0.2579 | 0.1152 | 0.2452 |
| WFG3 | 0.2667 | 0.1333 | 0.2177 |
| WFG4 | 0.1273 | 0.0541 | 0.1494 |
| WFG5 | 0.1370 | 0.0632 | 0.2006 |
| WFG6 | 0.1752 | 0.0687 | 0.1556 |
| WFG7 | 0.2457 | 0.0726 | 0.1957 |
| WFG8 | 0.2488 | 0.0759 | 0.2019 |
| WFG9 | 0.1122 | 0.0594 | 0.1383 |

Specifically, when a CX operator in a subproblem fails to generate new nondominated individuals for several generations, this indicates that CX cannot be expected to dis-cover a new nondominated individual within a finite number of generations. At this time, MOEA/D-HH switches from the CX to the CMA-ES operator for recycling inefficient evaluation costs.

According to its characteristics, the CMA-ES operator has a high probability of estimating the correct search direction for a sub-problem if there are sufficient sampling points. The evaluation cost of evaluating these sampling points comes from the redistribution of the evaluation cost occupied by CX. After switching to CMA-ES, CMA-ES is continued if a new individual can be nondominated. However, CMA-ES switches back to CX if it fails to update the individual to save evaluation costs. The sampling points generated by the CMA-ES are based on the distribution of the current subproblem and its neighborhood. Even if the sampling points are not nondominated, they still contain useful information regarding the evolutionary process.

To balance the evaluation cost of CMA-ES, when a sub-problem begins using the CMA-ES operator to generate new individuals, the individuals in the vicinity of this sub-problem (sub-neighborhood) switch to using the IDE operator for generating new individuals. The IDE is recognized as a powerful algorithm for solving SOPs, but it also requires a higher evaluation cost to solve MOPs. The sampling points generated

by the CMA-ES are sufficient for the IDE operator, so there is no need for additional evaluation costs. The IDE reuses the sampling points generated by CMA-ES, which dilutes the high evaluation cost of CMA-ES, thus achieving the purpose of mitigate the evaluation cost.

## 5.2 Algorithm

The framework is divided into three main parts: initialization, reproduction, and updating. Classical algorithms based on this framework utilize genetic operators to generate an offspring population (new individuals) during the reproduction phase. Subsequently, the newly generated individuals are used to update the current population. The reproduction and update processes continue to iterate until the stopping criteria are satisfied. Most approaches, such as MOEA/D-DE, utilize a single fixed operator throughout the search process to generate new individuals. Conversely, MOEA/D-HH, which is built up-on the MOEA/D framework and shares the same hyper-parameter initialization method as MOEA/D-DE, introduces the integration of multiple distinct operators for generating off-spring. Therefore, an operator selector was necessary during the reproduction phase of MOEA/D-HH. This selector not only determined the mating population but also chose the appropriate generating operator based on the current search condition. An efficiency inspection was performed on the subproblem after the regular update phase to evaluate the current search condition. The results of this inspection served as the criteria for switching between CX- and ED-based operators. This switching mechanism involved two components: a selector operator and efficiency inspection. The core process of MOEA/D-HH is illustrated in Figure 8.

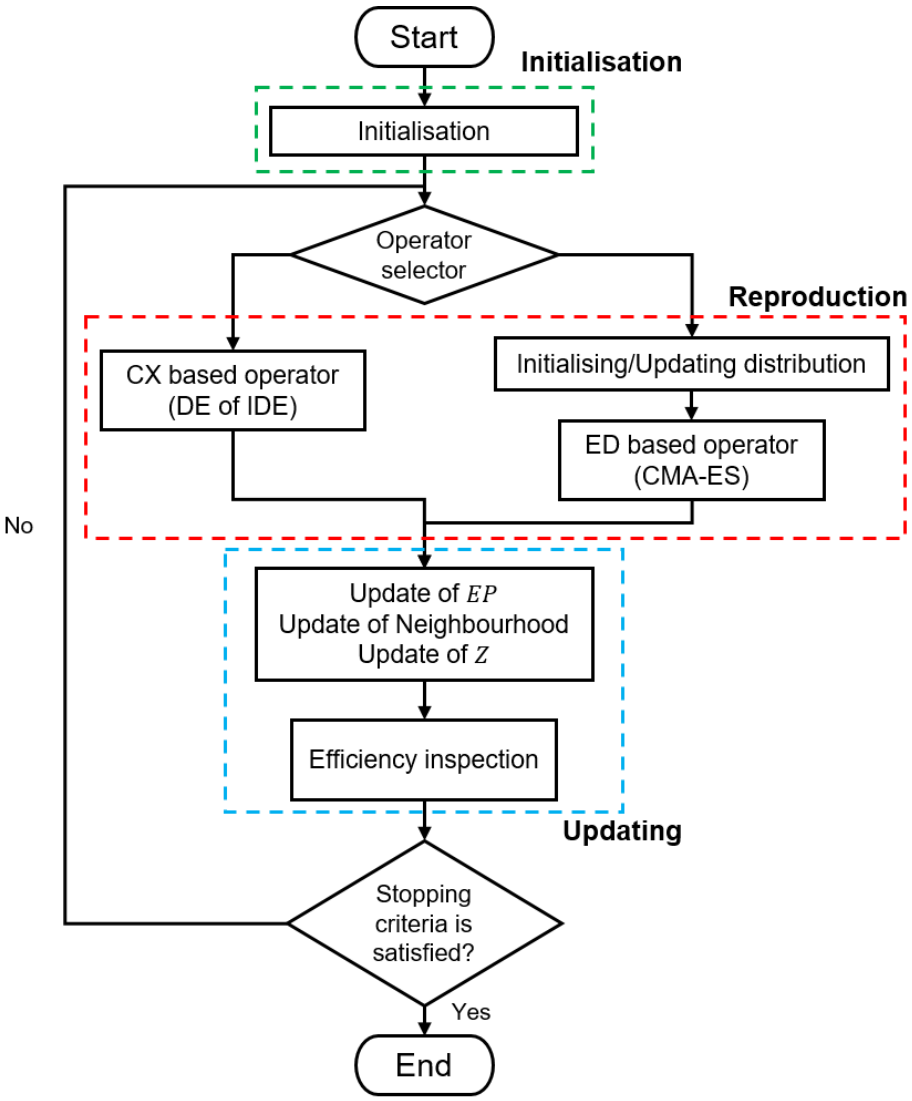


Figure 8: Core process of MOEA/D-HH.

In MOEA/D-HH, different sub-problems may utilize different operators to generate new solutions. Initially, all the subproblems employ the DE1 operator, as specified in Equation (). However, a decline in DE operator effectiveness is indicated if it encounters difficulties in identifying the appropriate evolutionary direction for several generations as the search progresses. In such cases, MOEA/D-HH switches the operator from DE1 to CMA-ES. The CMA-ES operator is known for its ability to identify correct search directions by utilizing distribution information. Therefore, is expected to be more effective in generating new non-dominated individuals. The parameters of CMA-ES will be initialized assuming that the operator of subproblem $i$ is switched to CMA-ES at generation $g$, as shown in Table 6.

Table 6: Initialized parameters of CMA-ES in MOEA/D-HH.

| Parameters | Values |
|---|---|
| $m$ | mean value of $B(i)^g$ |
| $p_\sigma$ | 0 |
| $\sigma$ | 0.5 |
| $p_c$ | 0 |
| C | $I$ |

At the same time, MOEA/D-HH will create a provisional archives population $\mathrm{A}(i)^g = \emptyset$ to store new individuals that are subsequently generated. After the initialisation, the procedure for CMA-ES in MOEA/D-HH is shown as follows:

**Input:** Setting solution $\mathrm{X} = B(i)^g \cup \mathrm{A}(i)^g$.

**Output:** $y_{best}$, which is the individual with best fitness value in set Y.

**Algorithm8: CMA-ES in MOEA/D-HH Framework**

**Step 1)** Sorting X using the fitness value of the scalarizing function.

**Step 2)** Updating distribution parameters using sorted X.

**Step 3)** Generating new solutions:

Initialize the set of new solutions $\mathrm{Y} = \emptyset$.

**for** $i = 0,1, \dots, U_{count} \times \lambda$:

$Y \leftarrow Y \cup y_i = m + \sigma \mathrm{C} \frac{z}{||z||}$, where $z \sim \mathcal{N}(0, \mathrm{I})$.

**Step 4) Repairing:** if an element of $y_i \in Y$ is out of the boundary, its element value is reset to the boundary.

**Step 5) Storing:** $\mathrm{A}(i)^{g+1} \leftarrow \mathrm{A}(i)^g \cup Y$.

The method in Step3) is equivalent to Equation (30). Specifically, $U_{count} = 1,2, \dots, U_{max} \in \mathbb{Z}$ represents the number of unsuccessful replacements. If $y_{best}$ cannot replace the current best ${x_i}^g$, then $U_{count} += 1$.

The CMA-ES strategy generates several new individuals based on the dominance distribution. Obviously, $y_{best}$ can be generated as long as enough individuals are generated that are not dominated by ${x_i}^g$. However, considering the practical evaluation cost, the volume of set Y cannot be expanded without limit. Therefore, in this study, $U_{max} = 3$.

Based on the characteristics of the MOEA/D framework, subproblems within the neighborhood have similar solutions. Moreover, set Y (similar to $A(i)$) is strongly dependent on $B(i)$, indicating that the individuals in $Y$ are influenced by the current subproblem $i$ and the entire neighbourhood. In addition, archived individuals improve the diversity of the population.

In MOEA/D-HH, the operator of other subproblems $i' \in [i - \frac{\lambda}{2}, i) \cup (i, i + \frac{\lambda}{2}]$ within the neighbourhood will switch to the IDE operator after the operator of subproblem i switches to CMA-ES, where $\lambda$ described in Equation () is the size of CMA-ES offspring. The IDE procedure in MOEA/D-HH is shown as follows:

**Input:** Setting solution $\mathrm{X} = B(i')^g \cup \mathrm{A}(i)^g$.

**Output:** new individual of the $i'$th subproblem $\mathrm{X}_{i'}^{g+1}$.

**Algorithm9: IDE in MOEA/D-HH Framework**

Step 1) Sorting and Partitioning:

Step 1.1) Sorting X using fitness value of scalarizing function.

Step 1.2) Superior group $S \leftarrow$ top 30% of X, group $I \leftarrow$ other individuals.

Step 2) Generation of mutation vector:

Randomly select $\mathrm{X}_S{}^g$ and $\mathrm{X}_I{}^g$ from groups $S$ and $I$, respectively.

Generate new mutation vector $\mathrm{V}_{i'}{}^g = \mathrm{X}_{i'}{}^g + F \cdot (\mathrm{X}_S{}^g - \mathrm{X}_I{}^g)$.

Step 3) Crossover:

$$u_{i',j}^g = \begin{cases} v_{i',j}^g, \text{if } (rand(0,1) \le CR, & \text{or } j = j_{rand}) \\ x_{i',j}^g & \end{cases}$$

where $u_{i',j}^g$ denotes the trial vector, $F$ denotes the mutation factor, and $CR$ denotes the crossover probability.

Step 4) Updating:

if $fit(\mathrm{U}_{i'}^g) \le fit(\mathrm{X}_{i'}^g)$:

$\mathrm{X}_{i'}^{g+1} = \mathrm{U}_{i'}^g$.

else:

$\mathrm{X}_{i'}^{g+1} = \mathrm{X}_{i'}^g$.

As described earlier in this Chapter, the operator switching mechanism in MOEA/D-HH consists of two components: selection and efficiency inspection. MOEA/D-HH initialises an empty list $L_{index} = \emptyset$ to store the indices of these subproblems, which facilitates the tracking of subproblems using CMA-ES. The subproblems execute the CMA-ES strategy if the current subproblem $i$ is present in $L_{index}$. Additionally, the current subproblem $i$ will use the IDE strategy for generating new individuals, if subproblem $i$ is not included in $L_{index}$ and there is a subproblem $i' \in [i - \frac{\lambda}{2}, i) \cup (i, i + \frac{\lambda}{2}]$ in $L_{index}$. When neither of the above conditions is satisfied, the subproblems execute the DE1 strategy. The selection part of the operator-switching mechanism is summarised as Figure 9:

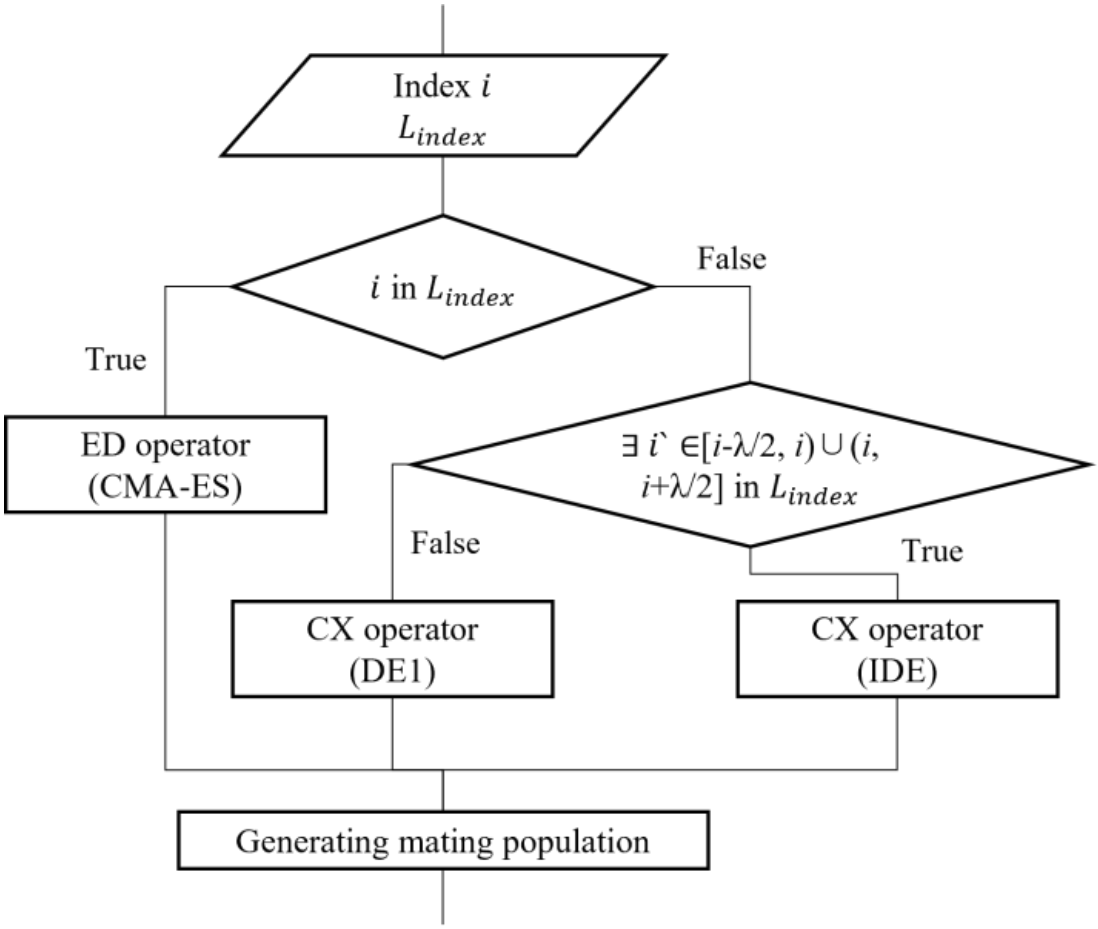


Figure 9: Operator selection strategy in MOEA/D-HH.

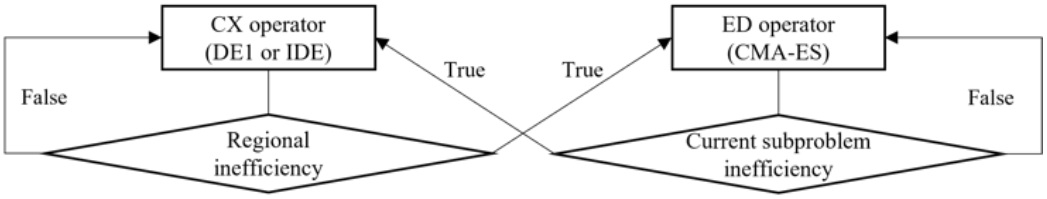


Figure 10: Efficiency inspection concept in MOEA/D-HH.

Efficiency Inspection is the other component of the operator-switching mechanism that is responsible for managing the members in $L_{index}$. The fundamental concept is illustrated in Figure 10. The criterion for identifying inefficiency is when a newly generated individual fails to outperform the current best solution, and this situation persists for a specified number of generations. This specified number of generations is considered a threshold. In our study, we established thresholds of five and three generations for the CX and ED strategies, respectively.

The threshold for the ED strategy was the same as the setting of $U_{count}$ that was proposed in **Algorithm8**. This was because the evaluation cost of the ED strategy was much higher than that of CX. The criterion during the efficiency inspection of a subproblem using CMA-ES is whether the threshold is exceeded. If the threshold is exceeded, the index of the subproblem is removed from $L_{index}$ and its provisional archive population is cleared simultaneously. Unlike the ED strategy, the CX strategy has a lower evaluation cost that allows for higher tolerance for inefficient situations.

In numerical experiments, it is common to observe stagnation in the search progress of a certain subproblem for several generations, whereas other subproblems in its neighbourhood continue to update non-dominated individuals with new solutions during the same time period. Therefore, the focus is on the subproblem itself and all other subproblems in its sub-neighbourhood that may exceed the inefficiency threshold when inspecting the efficiency of a subproblem using DE1 or IDE. In this study, the size of the sub-neighbourhood was set to parameter $\lambda$.

## 5.3 Numerical experiments

### 5.3.1 Experimental conditions

Experiments were conducted using the WFG test suite (Huband et al., 2005) [55] to assess the effectiveness of the proposed MOEA/D-HH and compare its performance with that of the original MOEA/D-DE. Furthermore, a reference version called MOEA/D-HHF was introduced that utilised fixed operators to examine the efficacy of the switching mechanism in MOEA/D-HH.

In MOEA/D-HHF, the index of a subproblem is represented by $i$ and the size of the neighbourhood is denoted by $T$. The ith subproblem employs the CMA-ES strategy to generate new individuals only when $i\%T == 0$. Subproblem $i' \in [i - \frac{\lambda}{2}, i) \cup (i, i + \frac{\lambda}{2}]$ uses the IDE strategy, while the remaining subproblems utilise the DE1 operator.

In this experiment, 21 trials were conducted for each approach and the average values were calculated. The penalty-based boundary intersection PBI scalarising function and WFG test suite were employed as the set of problem instances. The performances of these approaches were evaluated using the inverted generational distance plus (IGD+) metric (Ishibuchi et al., 2015) [56].

The WFG test suite was utilised as a problem instance in these experiments, which has been widely used and offers flexibility in adjusting the number of objectives and decision variables as needed. Test functions and a true Pareto front for two-, three-, and five-objective problems (referred to as WFG_2D, WFG_3D, and WFG_5D, respectively) were generated following the methodology outlined by Huband et al. (2006) [57].

The position and distance parameters were set to $k = 2$ and $l = n - k$, respectively, for the WFG_2D problems, where n represents the number of variables. The position parameter was set to $k = 4$ for WFG_3D and WFG_5D.

Table 7: Hyper-parameter compositions in the MOEA/D framework, where indicates the number of sub-sequent hyperparameter.

| **Parameters** | **Values** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| # Objectives ($M$) | 2 | | | | | 3 | | | 5 |
| Population size ($N$) | 150 | | 300 | | | 300 | | | 1000 |
| # Design variables ($n$) | 30 | 50 | 30 | 50 | 100 | 30 | 50 | 100 | 32 |
| Neighbourhood size ($T$) | 15 | | 21 | | | 21 | | | 51 |
| Terminal criteria (# evaluate) | 100,000 | | | | | | | | |

IGD+ was employed as a performance indicator to evaluate the performance of the approaches. The IGD+ metric measures the distance between the obtained solutions and true Pareto front. A lower IGD+ value indicates that the solutions are closer to the true Pareto front, implying better performance.

The parameters used for the MOEA/D framework are listed in Table 7. The neighbourhood size $T$ in the MOEA/D framework is generally set to be less than $10\%$ of the population size $N$. This is because having a large neighbourhood can result in a loss of necessary similarity between subproblems. However, the CMA-ES algorithm requires a certain number of individuals in the neighbourhood, and there is a hidden condition regarding the number of offspring, which is expressed as $T \geq \lambda$, where $\lambda = 4 + 3\lfloor \ln n \rfloor$.

Therefore, MOEA/D-HH cannot be applied when the parameters are set to $N = 150$, $T = 15$, and $n = 100$. The conditions for the CMA-ES were not met in this case.
For the WFG problems, the position parameter k must be a multiple of $M - 1$. Additionally, the distance parameter $l = n - k$ must be divisible by $k$. Consequently, the number of design variables was set as $n = 32$ for the WFG_5D problems, which was the closest number to $30$ and satisfied the aforementioned conditions.
The parameter for the PBI scalarising function was set to $\theta = 5$. The mutation factor $F$ and crossover probability $CR$ were set to $0.5$ and $0.9$, respectively, for the DE1 and IDE operators.

### 5.3.2 Experimental results of wfg_2D problem

Numerical experiments were conducted on the WFG_2D problem using five sets of hyper-parameters. The average IGD+ values obtained from these experiments are presented in Table 8, where the parameters $N$ and $n$ represent the number of subproblems and variables, respectively. The DE1 column represents the results of the original MOEA/D-DE, which served as a baseline control and was generated using the *jMetalpy library* (Benítez-Hidalgo et al., 2019) [58]. Columns HH and HHF represent the results for MOEA/D-HH and MOEA/D-HHF, respectively. The results indicated that the original MOEA/D-DE outperformed MOEA/D-HH and MOEA/D-HHF in most cases. However, MOEA/D-HH demonstrated better performance in WFG5 ($N = 300$, $n = 30, 50, 100$), WFG6 ($N = 150$, $n = 50$; $N = 300$, $n = 30, 50, 100$), and WFG9 ($N = 300$, $n = 100$). However, MOEA/D-HHF, which served as a control to evaluate the effectiveness of the switching mechanism, performed better than MOEA/D-HH for WFG2 ($N = 150$, $n = 30, 50$), WFG6 ($N = 150$, $n = 30$), and WFG9 ($N = 150$, $n = 30$; $N = 300$, $n = 30$). These results provided preliminary evidence of the effectiveness of the adaptive switching mechanism.

Table 8: Average value of IGD+ in WFG_2D problems.

| | | $N = 150$ | | | | $N = 300$ | | |
|---|---|---|---|---|---|---|---|---|
| | | **DE1** | **HH** | **HHF** | | **DE1** | **HH** | **HHF** |
| WFG1 | $n = 30$ | **1.1343** | 1.2215 | 1.2354 | $n = 30$ | **1.1682** | 1.2107 | 1.2359 |
| | $n = 50$ | **1.1912** | 1.2379 | 1.2489 | $n = 50$ | **1.2057** | 1.2322 | 1.2463 |
| | | | | | $n = 100$ | **1.2355** | 1.2456 | 1.2572 |
| WFG2 | | **0.0757** | 0.0989 | 0.0966 | | **0.0608** | 0.0766 | 0.0990 |
| | | **0.1264** | 0.1692 | 0.1686 | | **0.1050** | 0.1260 | 0.1764 |
| | | | | | | **0.1800** | 0.1964 | 0.2437 |
| WFG3 | | **0.1356** | 0.1616 | 0.1809 | | **0.1394** | 0.1474 | 0.1810 |
| | | **0.1640** | 0.1962 | 0.2535 | | **0.1691** | 0.1761 | 0.2489 |
| | | | | | | **0.2368** | 0.2379 | 0.3229 |
| WFG4 | | **0.0973** | 0.1068 | 0.1181 | | **0.1009** | 0.1051 | 0.1151 |
| | | **0.1156** | 0.1309 | 0.1446 | | **0.1207** | 0.1259 | 0.1442 |
| | | | | | | **0.1484** | 0.150 | 0.1655 |
| WFG5 | | **0.0672** | 0.0686 | 0.0729 | | 0.0678 | **0.0665** | 0.0724 |
| | | **0.0692** | 0.0704 | 0.0777 | | 0.0708 | **0.0683** | 0.0809 |
| | | | | | | 0.0752 | **0.0706** | 0.0990 |
| WFG6 | | 0.0893 | 0.0889 | **0.0887** | | 0.0924 | **0.0887** | 0.0949 |
| | | 0.0593 | **0.0570** | 0.0632 | | 0.0645 | **0.0572** | 0.0883 |
| | | | | | | 0.0513 | **0.0391** | 0.0633 |
| WFG7 | | **0.0183** | 0.0481 | 0.0633 | | **0.0259** | 0.0365 | 0.0598 |
| | | **0.0356** | 0.0728 | 0.1236 | | **0.0464** | 0.0567 | 0.1112 |
| | | | | | | **0.0948** | 0.1010 | 0.1884 |
| WFG8 | | **0.1058** | 0.1376 | 0.1799 | | **0.1046** | 0.1185 | 0.1620 |
| | | **0.1390** | 0.1804 | 0.2400 | | **0.1400** | 0.1499 | 0.2185 |
| | | | | | | **0.1959** | 0.2045 | 0.2757 |
| WFG9 | | 0.0795 | 0.0890 | **0.0758** | | **0.0816** | 0.0895 | 0.0715 |
| | | **0.0558** | 0.0655 | 0.0850 | | **0.0682** | 0.0700 | 0.1000 |
| | | | | | | 0.0637 | **0.0331** | 0.1227 |

The evolutionary trajectory was analysed, and selected results are presented in Figures 11-13. The $x$ coordinate denotes the progress of iterations, whereas the $y$ coordinate represents the average IGD+ values. These figures demonstrate that MOEA/D-HHF exhib-ited faster convergence during the initial stages of iteration, despite the significant differences between the results of the HHF and the other two algorithms.

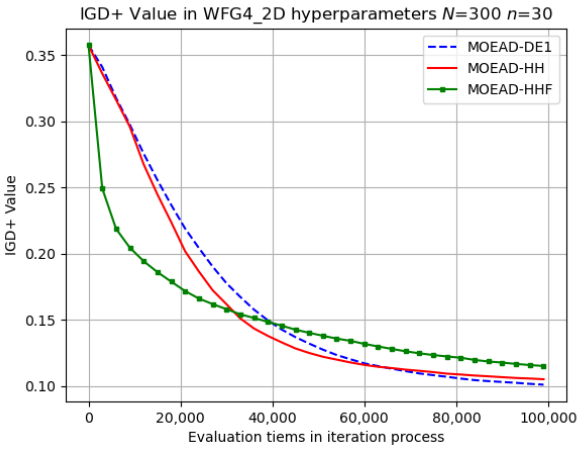


Figure 11: Visual evolutionary trajectory of two-objective WFG4 problem with 300 subproblems and 30 variables.

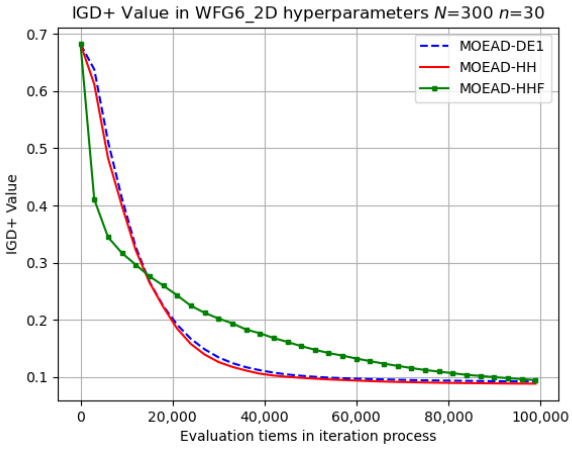


Figure 12: Visual evolutionary trajectory of two-objective WFG6 problem with 300 subproblems and 30 variables.

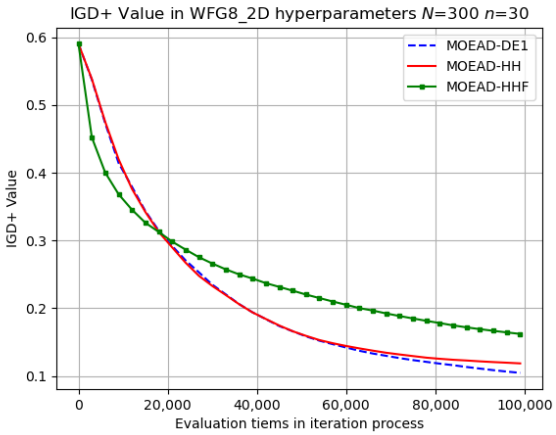


Figure 13: Visual evolutionary trajectory of two-objective WFG8 problem with 300 subproblems and 30 variables.

Table 9: SRR and PR of operators in WFG_2D problems (1).

| Subproblem Number $N = 150$ | | MOEA/D-DE SRR | MOEA/D-HH SRR | MOEA/D-HH | | |
|---|---|---|---|---|---|---|
| | | | | DE-SRR(PR) | IDE-SRR(PR) | CMA-SRR(PR) |
| WFG1 | $n = 30$ $n = 50$ | **0.1057** | 0.0637 | 0.2928 (0.3070) | 0.0513 (0.6605) | 0.0344 (0.0325) |
| | | **0.0921** | 0.0604 | 0.3003 (0.2984) | 0.0501 (0.6683) | 0.0328 (0.0332) |
| WFG2 | | **0.1482** | 0.1015 | 0.2830 (0.5539) | 0.0405 (0.4211) | 0.0541 (0.0249) |
| | | **0.1579** | 0.1054 | 0.2883 (0.5627) | 0.0455 (0.4133) | 0.0718 (0.0240) |
| WFG3 | | **0.1471** | 0.1117 | 0.2923 (0.5510) | 0.0489 (0.4278) | 0.0251 (0.0212) |
| | | **0.1580** | 0.1246 | 0.2828 (0.5692) | 0.0707 (0.4133) | 0.0723 (0.0175) |
| WFG4 | | **0.0641** | 0.0540 | 0.2269 (0.3278) | 0.0510 (0.6378) | 0.0437 (0.0344) |
| | | **0.0710** | 0.0589 | 0.2306 (0.3481) | 0.0563 (0.6204) | 0.0516 (0.0314) |
| WFG5 | | **0.0726** | 0.0562 | 0.2619 (0.3339) | 0.0424 (0.6321) | 0.0044 (0.0340) |
| | | **0.0785** | 0.0611 | 0.2587 (0.3400) | 0.0514 (0.6302) | 0.0071 (0.0299) |
| WFG6 | | **0.0936** | 0.0780 | 0.2895 (0.3937) | 0.0512 (0.5786) | 0.0028 (0.0277) |
| | | **0.1015** | 0.0838 | 0.2895 (0.4078) | 0.0567 (0.5680) | 0.0025 (0.0243) |
| WFG7 | | **0.1281** | 0.0990 | 0.2789 (0.4900) | 0.0581 (0.4869) | 0.0479 (0.0231) |
| | | **0.1378** | 0.1067 | 0.2686 (0.4978) | 0.0765 (0.4819) | 0.0830 (0.0203) |
| WFG8 | | **0.1305** | 0.1057 | 0.2869 (0.4794) | 0.0682 (0.4984) | 0.0685 (0.0222) |
| | | **0.1357** | 0.1073 | 0.2779 (0.4764) | 0.0790 (0.5029) | 0.0900 (0.0206) |
| WFG9 | | **0.0658** | 0.0495 | 0.2283 (0.3155) | 0.0432 (0.6498) | 0.0211 (0.0347) |
| | | **0.0737** | 0.0563 | 0.2316 (0.3690) | 0.0466 (0.5989) | 0.0246 (0.0321) |

The impacts of MOEA/D-HH on the SRR is presented in Tables 9 and 10. The bold entries in these tables compare the SRR values of MOEA/D-DE with those of MOEA/D-HH. The rightmost columns provide the specific SRR and picked rate (PR) of each operator in MOEA/D-HH. In these tables, it is a fundamental requirement that the sum of the PR in each row equals one.

Table 10: SRR and PR of operators in WFG_2D problems (2).

| Subproblem Number $N = 300$ | | MOEA/D-DE SRR | MOEA/D-HH SRR | MOEA/D-HH | | |
|---|---|---|---|---|---|---|
| | | | | DE-SRR(PR) | IDE-SRR(PR) | CMA-SRR(PR) |
| WFG1 | $n = 30$ | **0.1833** | 0.1472 | 0.2907 (0.5575) | 0.0937 (0.4295) | 0.0533 (0.0130) |
| | $n = 50$ | **0.1598** | 0.1332 | 0.2938 (0.5253) | 0.0893 (0.4595) | 0.0553 (0.0152) |
| | $n = 100$ | **0.1424** | 0.1276 | 0.2846 (0.5355) | 0.0772 (0.4513) | 0.0447 (0.0132) |
| WFG2 | $n = 30$ | **0.2579** | 0.2225 | 0.3082 (0.8461) | 0.0621 (0.1463) | 0.0843 (0.0076) |
| | $n = 50$ | **0.2664** | 0.2252 | 0.3107 (0.8544) | 0.0629 (0.1381) | 0.1185 (0.0075) |
| | $n = 100$ | **0.2650** | 0.2300 | 0.3002 (0.8761) | 0.0581 (0.1186) | 0.1295 (0.0053) |
| WFG3 | $n = 30$ | **0.2667** | 0.2480 | 0.3002 (0.9064) | 0.0622 (0.0896) | 0.0198 (0.0039) |
| | $n = 50$ | **0.2869** | 0.2757 | 0.3025 (0.9478) | 0.1109 (0.0503) | 0.1076 (0.0020) |
| | $n = 100$ | 0.2871 | **0.2888** | 0.2968 (0.9814) | 0.1671 (0.0180) | 0.1991 (0.0006) |
| WFG4 | $n = 30$ | **0.1273** | 0.1129 | 0.2276 (0.5780) | 0.0829 (0.4054) | 0.0749 (0.0166) |
| | $n = 50$ | **0.1351** | 0.1196 | 0.2227 (0.6084) | 0.0924 (0.3773) | 0.0863 (0.0143) |
| | $n = 100$ | **0.1445** | 0.1317 | 0.2181 (0.6522) | 0.0958 (0.3379) | 0.1076 (0.0099) |
| WFG5 | $n = 30$ | **0.1370** | 0.1148 | 0.2592 (0.5816) | 0.0617 (0.4010) | 0.0033 (0.0173) |
| | $n = 50$ | **0.1469** | 0.1294 | 0.2644 (0.6132) | 0.0718 (0.3727) | 0.0062 (0.0141) |
| | $n = 100$ | **0.1613** | 0.1420 | 0.2516 (0.6487) | 0.0783 (0.3422) | 0.0058 (0.0091) |
| WFG6 | $n = 30$ | **0.1752** | 0.1509 | 0.2936 (0.6324) | 0.0744 (0.3543) | 0.0025 (0.0133) |
| | $n = 50$ | **0.2032** | 0.1905 | 0.3005 (0.7172) | 0.0964 (0.2745) | 0.0054 (0.0083) |
| | $n = 100$ | **0.2268** | 0.2179 | 0.2947 (0.7798) | 0.1105 (0.2158) | 0.0122 (0.0044) |
| WFG7 | $n = 30$ | **0.2457** | 0.2265 | 0.2920 (0.8624) | 0.0771 (0.1323) | 0.0358 (0.0053) |
| | $n = 50$ | **0.2625** | 0.2473 | 0.2875 (0.9039) | 0.1216 (0.0928) | 0.1203 (0.0033) |
| | $n = 100$ | **0.2652** | 0.2567 | 0.2782 (0.9305) | 0.1625 (0.0678) | 0.2050 (0.0017) |
| WFG8 | $n = 30$ | **0.2488** | 0.2316 | 0.2981 (0.8415) | 0.1028 (0.1532) | 0.0870 (0.0054) |
| | $n = 50$ | **0.2539** | 0.2413 | 0.2930 (0.8660) | 0.1259 (0.1299) | 0.1262 (0.0041) |
| | $n = 100$ | **0.2531** | 0.2448 | 0.2838 (0.8845) | 0.1395 (0.1127) | 0.1663 (0.0028) |
| WFG9 | $n = 30$ | **0.1122** | 0.0930 | 0.2168 (0.5046) | 0.0739 (0.4760) | 0.0525 (0.0194) |
| | $n = 50$ | **0.1361** | 0.1062 | 0.2259 (0.5521) | 0.0817 (0.4312) | 0.0642 (0.0167) |
| | $n = 100$ | **0.1649** | 0.1315 | 0.2294 (0.6444) | 0.0844 (0.3459) | 0.0427 (0.0098) |

From these tables, particularly Table 10, it is clear that the HH-DE(PR) values were remarkably high, while the SRR values of MOEA/D-DE were superior to those of MOEA/D-HH. These high values indicate that MOEA/D-DE performed exceptionally well in the experimental environment.

Consequently, MOEA/D-HH adaptively selected a higher proportion of DE1 operators. This observation is consistent with the MOEA/D-DE and MOEA/D-HH curves shown in the figures.

### 5.3.3 Experimental results of wfg_3D problem

Numerical experiments were conducted on the WFG_3D problems using three different sets of hyper-parameters, and the average IGD+ values are listed in Table 11. In contrast to the results observed for the WFG2_D problems, the original MOEA/D-DE only outperformed MOEA/D-HH in certain instances, such as WFG1 ($n = 30, 50, 100$), WFG3 ($n = 30$), and WFG9 ($n = 30$).

The performance of MOEA/D-HHF was inferior to that of MOEA/D-HH, highlighting the effectiveness of the operator-switching mechanism. Selected results for the evolutionary trajectory are shown in Figures 14-18. MOEA/D-HH did not exhibit superior performance compared to that of MOEA/D-DE in WFG1, WFG3, and WFG9 with $n = 30$, as shown in Figures 14, 15, and 16, respectively. However, the differences in their performances were exceedingly small.

Table 11: Average value of IGD+ in WFG_3D problems.

| | | DE1 | HH | HHF |
|---|---|---|---|---|
| WFG1 | $n = 30$ | **1.4392** | 1.4551 | 1.4489 |
| | $n = 50$ | **1.4459** | 1.4610 | 1.4705 |
| | $n = 100$ | **1.4516** | 1.4632 | 1.4789 |
| WFG2 | | 0.3499 | **0.3114** | 0.3875 |
| | | 0.4195 | **0.3618** | 0.4525 |
| | | 0.5159 | **0.4161** | 0.5376 |
| WFG3 | | **0.1334** | 0.1429 | 0.2299 |
| | | 0.2580 | **0.2390** | 0.3126 |
| | | 0.3920 | **0.3491** | 0.3988 |
| WFG4 | | 0.3000 | **0.2566** | 0.3289 |
| | | 0.3426 | **0.2779** | 0.3143 |
| | | 0.3839 | **0.2904** | 0.3151 |
| WFG5 | | 0.1728 | **0.1521** | 0.1866 |
| | | 0.1979 | **0.1604** | 0.2326 |
| | | 0.2323 | **0.1701** | 0.2698 |
| WFG6 | | 0.1913 | **0.1792** | 0.2102 |
| | | 0.1996 | **0.1421** | 0.2429 |
| | | 0.2367 | **0.1228** | 0.2404 |
| WFG7 | | 0.3225 | **0.2580** | 0.3462 |
| | | 0.4050 | **0.3072** | 0.3983 |
| | | 0.4842 | **0.3503** | 0.4083 |
| WFG8 | | 0.4180 | **0.3846** | 0.4474 |
| | | 0.4714 | **0.4126** | 0.4801 |
| | | 0.5202 | **0.4227** | 0.4671 |
| WFG9 | | **0.1925** | 0.1968 | 0.2214 |
| | | 0.2212 | **0.1809** | 0.2567 |
| | | 0.2651 | **0.1567** | 0.2814 |

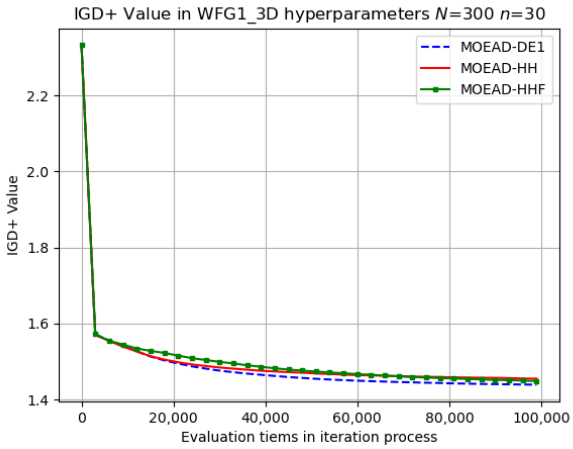


Figure 14: Visual evolutionary trajectory of three-objective WFG1 problem with 300 subproblems and 30 variables.

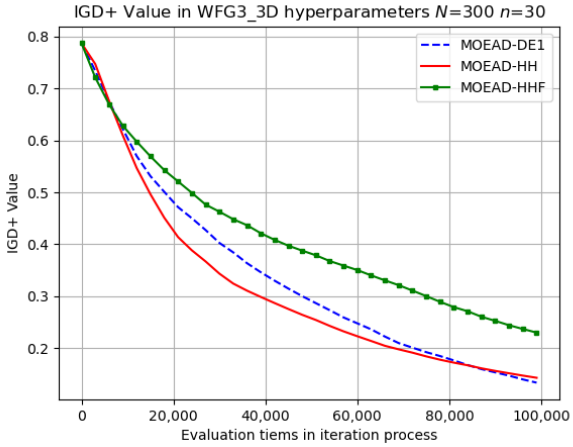


Figure 15: Visual evolutionary trajectory of three-objective WFG3 problem with 300 subproblems and 30 variables.

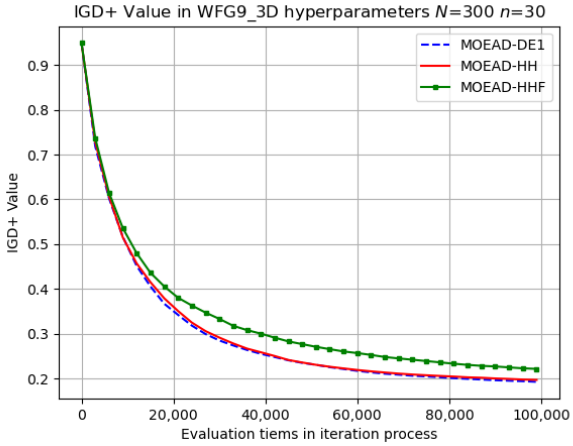


Figure 16: Visual evolutionary trajectory of three-objective WFG9 problem with 300 subproblems and 30 variables.

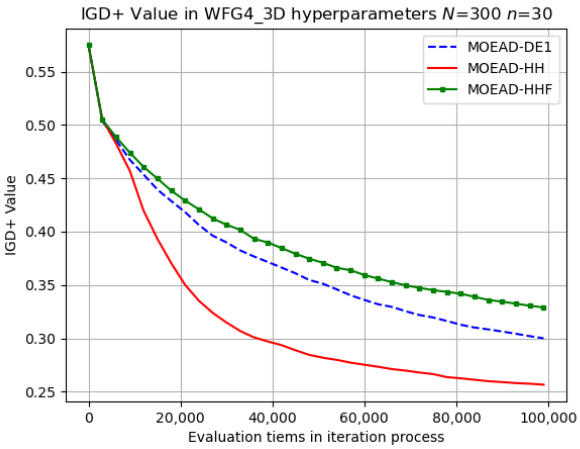


Figure 17: Visual evolutionary trajectory of three-objective WFG4 problem with 300 subproblems and 30 variables.

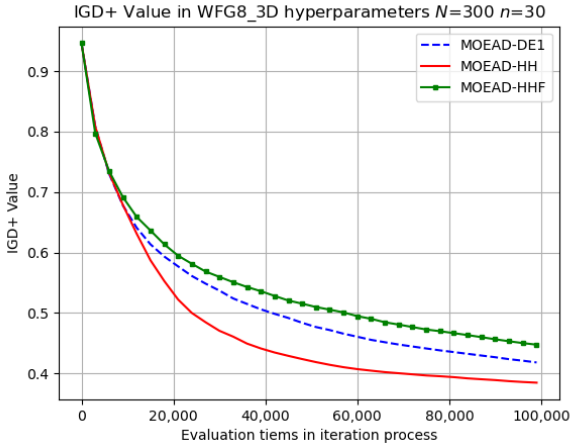


Figure 18: Visual evolutionary trajectory of three-objective WFG8 problem with 300 subproblems and 30 variables.

In contrast, MOEA/D-HH exhibited a significantly better performance than those of MOEA/D-DE and MOEA/D-HHF for WFG4, as shown in Figure 17. In addition, MOEA/D-HH demonstrated superior performance on WFG8, as shown in Figure 18. WFG8 can be considered as a challenging problem owing to the variations in the distance related parameter values among the different Pareto optimal solutions. Figure x displays the solutions obtained through optimization for the WFG8 problem. In this figure, the red points represent our proposed MOEA/D-HH algorithm, while the blue and green points correspond to the reference MOEA/D-DE1 and MOEA/D-HH, respectively. Upon careful observation, it becomes apparent that, regardless of

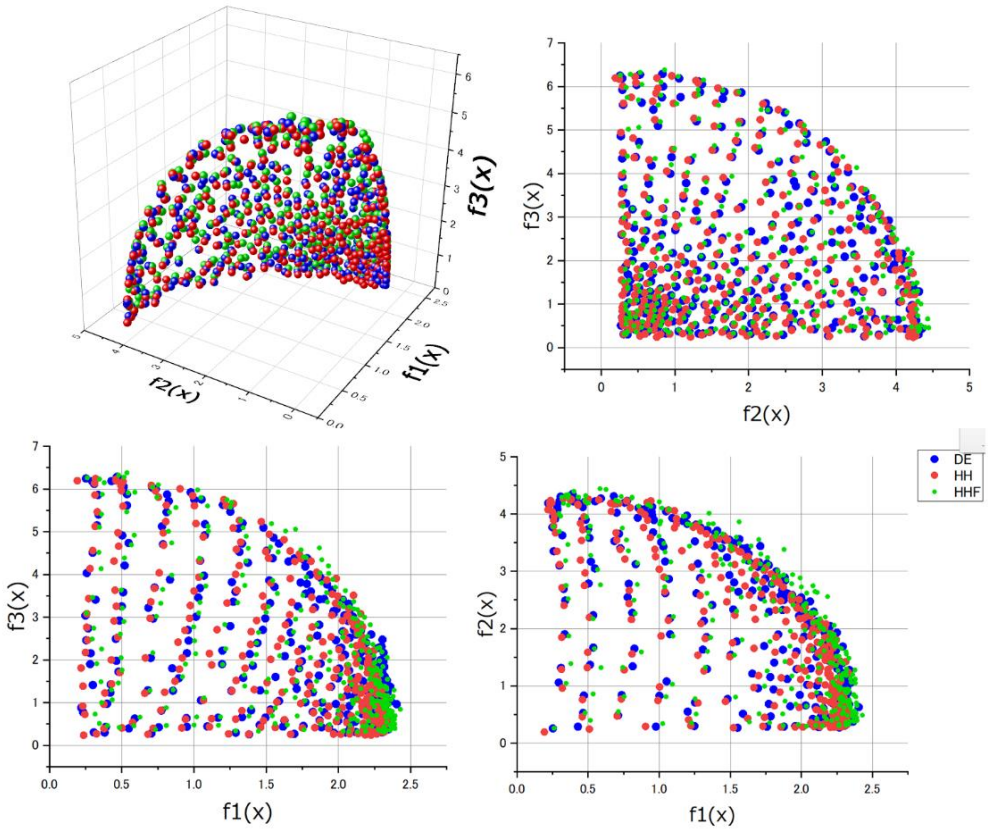


Figure 19: Comparison of evolutionary results for the WFG8 problem, including an overall 3D performance graph and projections on each dimension.

the projection on any dimension, the red points consistently cluster closer to the origin than the points in other colours. In Pareto optimization terms, this indicates that the evolutionary results of MOEA/D-HH have a greater dominance over the solutions obtained by other methods. This result aligns with the outcomes of IGD+.

The SRR and PR results are listed in Table 12. Similar to the WFG_2D problems, MOEA/D-HH generally exhibited lower SRR values than those of MOEA/D-DE. However, the HH-DE-SRR values were significantly higher in the MOEA/D-HH group. The dominance relationship between the solutions was less likely to be generated and the solutions were more likely to be non-dominated as the number of objectives increased. This indicates that searching is very difficult in multi-objective problems.

Table 12: SRR and PR of operators in WFG_3D problems.

| | | MOEA/D-DE SRR | MOEA/D-HH SRR | MOEA/D-HH | | |
|---|---|---|---|---|---|---|
| | | | | DE-SRR(PR) | IDE-SRR(PR) | CMA-SRR(PR) |
| WFG1 | $n = 30$ | **0.0963** | 0.0842 | 0.2664 (0.4828) | 0.0512 (0.4892) | 0.0283 (0.0281) |
| | $n = 50$ | **0.0983** | 0.0850 | 0.2667 (0.4955) | 0.0530 (0.4775) | 0.0297 (0.0270) |
| | $n = 100$ | **0.0978** | 0.0861 | 0.2587 (0.5298) | 0.0443 (0.4475) | 0.0247 (0.0227) |
| WFG2 | $n = 30$ | **0.1152** | 0.0998 | 0.2398 (0.5386) | 0.0752 (0.4384) | 0.0888 (0.0230) |
| | $n = 50$ | **0.1143** | 0.0977 | 0.2378 (0.5321) | 0.0786 (0.4453) | 0.0933 (0.0225) |
| | $n = 100$ | **0.1089** | 0.1000 | 0.2326 (0.5515) | 0.0744 (0.4302) | 0.0948 (0.0182) |
| WFG3 | $n = 30$ | **0.1333** | 0.1173 | 0.2357 (0.5140) | 0.1084 (0.4631) | 0.1527 (0.0229) |
| | $n = 50$ | **0.1267** | 0.1211 | 0.2574 (0.5221) | 0.1185 (0.4565) | 0.1764 (0.0214) |
| | $n = 100$ | 0.1190 | **0.1240** | 0.2503 (0.5368) | 0.1163 (0.4462) | 0.1806 (0.0170) |
| WFG4 | $n = 30$ | **0.0541** | 0.0497 | 0.2363 (0.3399) | 0.0455 (0.6184) | 0.0481 (0.0418) |
| | $n = 50$ | **0.0534** | 0.0506 | 0.2362 (0.3492) | 0.0484 (0.6108) | 0.0481 (0.0400) |
| | $n = 100$ | **0.0533** | 0.0522 | 0.2244 (0.3690) | 0.0480 (0.5980) | 0.0529 (0.0330) |
| WFG5 | $n = 30$ | **0.0632** | 0.0567 | 0.2698 (0.3327) | 0.0468 (0.6307) | 0.0104 (0.0367) |
| | $n = 50$ | **0.0639** | 0.0586 | 0.2679 (0.3441) | 0.0508 (0.6214) | 0.0104 (0.0345) |
| | $n = 100$ | **0.0635** | 0.0587 | 0.2510 (0.3524) | 0.0501 (0.6194) | 0.0112 (0.0282) |
| WFG6 | $n = 30$ | **0.0687** | 0.0670 | 0.2434 (0.3909) | 0.0630 (0.5752) | 0.0552 (0.0340) |
| | $n = 50$ | **0.0735** | 0.0731 | 0.2349 (0.4199) | 0.0736 (0.5501) | 0.0614 (0.0300) |
| | $n = 100$ | 0.0785 | **0.0791** | 0.2140 (0.4489) | 0.0812 (0.5286) | 0.0752 (0.0225) |
| WFG7 | $n = 30$ | **0.0726** | 0.0718 | 0.2502 (0.3962) | 0.0697 (0.5849) | 0.1573 (0.0389) |
| | $n = 50$ | 0.0713 | **0.0714** | 0.2501 (0.4100) | 0.0711 (0.5516) | 0.1592 (0.0384) |
| | $n = 100$ | 0.0666 | **0.0703** | 0.2342 (0.4292) | 0.0667 (0.5389) | 0.1564 (0.0319) |
| WFG8 | $n = 30$ | **0.0759** | 0.0755 | 0.2600 (0.4101) | 0.0693 (0.5517) | 0.1508 (0.0382) |
| | $n = 50$ | 0.0716 | **0.0720** | 0.2562 (0.4195) | 0.0676 (0.5419) | 0.1516 (0.0386) |
| | $n = 100$ | 0.0680 | **0.0717** | 0.2424 (0.4397) | 0.0639 (0.5281) | 0.1543 (0.0323) |
| WFG9 | $n = 30$ | **0.0594** | 0.0549 | 0.2044 (0.3834) | 0.0540 (0.5818) | 0.0380 (0.0348) |
| | $n = 50$ | **0.0650** | 0.0594 | 0.2051 (0.3815) | 0.0648 (0.5868) | 0.0536 (0.0317) |
| | $n = 100$ | **0.0719** | 0.0662 | 0.1927 (0.4206) | 0.0687 (0.5561) | 0.0587 (0.0233) |

The random-based CX strategy incurs a high evaluation cost to explore a vast search space in multi-objective problems. The distribution-based ED operator incurs a higher evaluation cost for generating new individuals than the CX operator. However, the individuals generated by the ED strategy have a higher likelihood of being in the correct search direction (promising regions). This is because the individuals generated by ED take advantage of the approximate gradient information.

### 5.3.4 Experimental results of wfg_3D problem

Numerical experiments were conducted on the WFG_5D problem using a single set of hyper-parameters, and the average IGD+ values are presented in Table 13. The experimental results exhibited interesting patterns as the number of objectives increased. Although MOEA/D-DE performed well in the WFG1 problem, MOEA/D-HHF demonstrated its superiority for the first time in the WFG2, WFG5, and WFG6 problems.

To provide further insight, selected results of the evolutionary trajectory are presented in Figures 20–22. MOEA/D-HHF achieved the best overall performance with a faster convergence rate during the early iterations, as shown in Figure 20. However, the convergence rate of MOEA/D-HH exhibited a significant improvement when the number of evaluations exceeded $40{,}000$, achieving a final result very close to that of MOEA/D-HHF.

Table 13: Average value of IGD+ in WFG_5D problems.

| | DE1 | HH | HHF |
|---|---|---|---|
| WFG1 | **2.0336** | 2.0346 | 2.0403 |
| WFG2 | 0.7278 | 0.6249 | **0.6214** |
| WFG3 | 0.9334 | **0.7364** | 0.9091 |
| WFG4 | 0.7183 | **0.6825** | 0.7058 |
| WFG5 | 0.6621 | 0.6579 | **0.6431** |
| WFG6 | 1.1639 | 1.1193 | **1.1156** |
| WFG7 | 0.8753 | **0.7354** | 0.8351 |
| WFG8 | 0.9514 | **0.8803** | 0.9610 |
| WFG9 | 0.3865 | **0.3614** | 0.4929 |

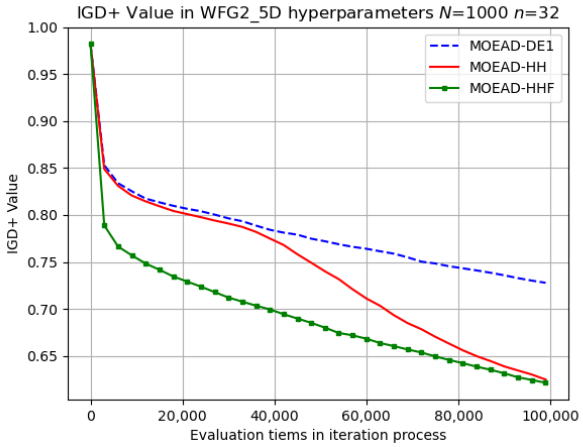


Figure 20: Visual evolutionary trajectory of five-objective WFG2 problem with 1000 subproblems and 32 variables.

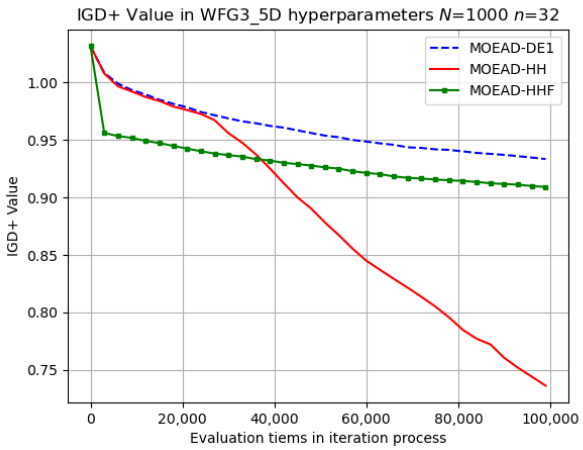


Figure 21: Visual evolutionary trajectory of five-objective WFG3 problem with 1000 subproblems and 32 variables.

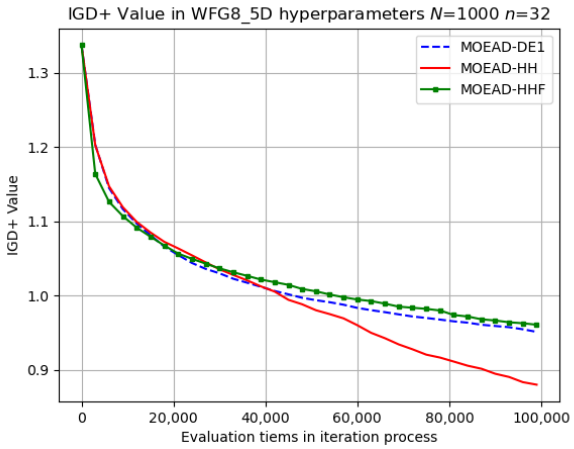


Figure 22: Visual evolutionary trajectory of five-objective WFG8 problem with 1000 subproblems and 32 variables.

MOEA/D-HH presented substantial advancements in the search process, particularly during the middle stage, as shown in Figure 21. Specifically, the evolutionary trajectory of MOEA/D-HH was significantly different from those of the other methods after the middle stage of the search. This behaviour can be attributed to the favourable compatibility between the switching mechanism and characteristics of this problem.

The improvements and evolutionary trajectory observed in MOEA/D-HH clearly indicate its superior performance compared to that of MOEA/D-DE. The same is true for the difficult WFG8 problem, as shown in Figure 22. The SRR and PR results are listed in Table 14. In contrast to previous results, the SRR values of MOEA/D-HH were generally higher than those of MOEA/D-DE for WFG_5D problems, and HH-DE(PR) achieved the highest values compared to those of the 2D and 3D problems.

HH-IDE(PR) and HH-CMA(PR) exhibited relatively lower values. Conversely, HH-IDE-SRR and HH-CMA-SRR were even higher than HH-DE-SRR in some cases, which was not observed in the 2D and 3D problems. These results are consistent with the expectations for MOEA/D-HH.

Table 14: SRR and PR of operators in WFG_5D problems.

| | MOEA/D-DE SRR | MOEA/D-HH SRR | MOEA/D-HH | | |
|---|---|---|---|---|---|
| | | | DE-SRR(PR) | IDE-SRR(PR) | CMA-SRR(PR) |
| WFG1 | **0.2122** | 0.1916 | 0.2535 (0.8602) | 0.0955 (0.1323) | 0.1257 (0.0074) |
| WFG2 | 0.2342 | **0.3019** | 0.3134 (0.9508) | 0.4769 (0.0465) | 0.3647 (0.0027) |
| WFG3 | 0.2178 | **0.2218** | 0.3580 (0.7831) | 0.1150 (0.2036) | 0.0996 (0.0133) |
| WFG4 | **0.1494** | 0.1402 | 0.2636 (0.7645) | 0.0506 (0.2185) | 0.0134 (0.0170) |
| WFG5 | **0.2006** | 0.1893 | 0.3727 (0.7586) | 0.0458 (0.2236) | 0.0011 (0.0178) |
| WFG6 | 0.1556 | **0.1616** | 0.2765 (0.7340) | 0.1306 (0.2490) | 0.0987 (0.0167) |
| WFG7 | 0.1957 | **0.2117** | 0.3690 (0.7769) | 0.1254 (0.1986) | 0.3421 (0.0245) |
| WFG8 | 0.2019 | **0.2043** | 0.3731 (0.7672) | 0.1024 (0.2075) | 0.3159 (0.0253) |
| WFG9 | 0.1383 | **0.1521** | 0.2142 (0.7350) | 0.1661 (0.2535) | 0.1211 (0.0115) |

## 5.4 Summary

Numerous studies have highlighted the limitations of relying on only one single offspring-generation strategy in optimization algorithms, which has led to the increasing popularity of hybrid evolutionary algorithms. In our paper, we introduce the MOEA/D-HH method, and we aim to elucidate the underlying factors contributing to the advantages of hybrid algorithms. Within this proposed algorithm, we have devised an adaptive operator switching mechanism rooted in the concept of operator efficiency inspection, specifically focusing on the successful replacement rate (SRR). This mechanism takes into consideration the specific characteristics of the MOEA/D framework and strives to balance the evaluation costs between the CX and ED strategy. Empirical support for the effectiveness of this switching mechanism is provided through experimental results.

Furthermore, the experimental results indicate that operators (DE1 and IDE) based on the CX strategy take a mainstream in the hybrid algorithm MOEA/D-HH (they are selected with a higher probability by the switching mechanism). Simultaneously, from the perspective of the SRR, the inclusion of non-mainstream operators (the ED operator) significantly enhances the search efficiency of DE1. The significant improvement in the mainstream strategy (or operator) at the SRR level can directly impact the overall performance of the algorithm, even when the overall SRR of the hybrid algorithm does not exhibit substantial fluctuations. This phenomenon is

particularly evident when MOEA/D-HH is applied to the 3-objective test suite. We hope that our research can provide fresh insights for related studies. The main contributions of this paper are follows:

- Concept of re-emphasising the necessity for a hybrid algorithm: Different from the previous hybrid algorithms that combine CX and ED strategies, we dynamically 'recycle' the inefficient evaluation cost occupied by CX and 'redistribute' it to ED from the point of view of successful replacement rate (SRR).
- Reuse of evaluation costs: Although various evolutionary algorithms based on CX or ED have been proposed in existing studies, most of them focus only on non-dominant individuals. Based on the underlying logic of the ED strategy, those dominated solutions are also considered containing information relevant to evolution. Therefore, in the proposed algorithm, dominated solutions generated by the ED strategy are reused as archive populations, which contributes to new ideas for maintaining diversity and balancing the high evaluation cost of ED strategies in hybrid algorithms.
- Framework adaptation: In this study, we propose an operator switching mechanism depend on the Efficiency Inspection. Different from the switching mechanism in other hybrid algorithms, our proposal is tailor-made, which fully considers the characteristics of the neighbourhood in the MOEA/D framework.

# Chapter 6 Conclusions

## 6.1 Summary of this research

In this doctoral dissertation, I encompassed my primary work over the past few years. During my doctoral studies, I dedicated myself to solving multi-objective optimization problems through evolutionary algorithms, designing and implementing efficient hybrid algorithms based on the MOEA/D framework. Among these, the MOEA/D-EF model integrates several evolution operators with different search characteristics based on the CX strategy into the MOEA/D framework. Meanwhile, the MOEA/D-HH model integrates evolution operators based on both the CX and ED strategies into the MOEA/D framework. To achieve this objective, we extensively researched and analyzed advanced evolution operators with different search characteristics and strategies, making necessary adaptations during the research process. Additionally, based on the distinct characteristics of these operators, adaptive operator switching mechanisms were designed to align with the MOEA/D framework.

In the second chapter of this paper, several important concepts relevant to this study are reviewed, including the MOEA/D framework, evolution operators based on the CX strategy, evolution operators based on the ED strategy, and two performance metrics widely used in multi-objective optimization problems. In the third chapter, the introduced evolution operators in this study are discussed in detail. Among them, IDE and JADE were originally designed to solve single-objective optimization problems, and we modified these two operators in conjunction with the relevant concepts of neighborhoods in the MOEA/D framework. DE-IDEAL is an original operator tailored for the MOEA/D framework. Its main feature is based on the geometric relationships in Euclidean space between ideal and current situations, selecting appropriate historical information to enhance the evolution efficiency of the operator. CMA-ES is an operator based on the ED strategy, similar to IDE and JADE, and is suitable for single-objective optimization problems. We achieved the objective of using CMA-ES within the MOEA/D framework to address multi-objective optimization problems by modifying its initialization and parameter update methods. In chapter four of this paper, we provided a detailed introduction to a hybrid model named MOEA/D-EF, characterized by the utilization of a priori-based operator switching mechanism. This mechanism aims for adaptive selection of suitable operators based on the current search situation. Additionally, we introduced a method for evaluating the overall performance of the algorithm across an entire test suite in the context of the MOEA/D-EF research. The results of numerical experiments supported the broader applicability of MOEA/D-EF compared to the contrastive model. In chapter five of this paper, we provided a detailed introduction to a hybrid model named MOEA/D-HH, which features an operator switching mechanism based on Efficiency Inspection. This mechanism aims to achieve adaptive switching between operators based on the CX strategy and operators based on the ED strategy,

depending on the current search situation. Considering the substantial evaluation cost associated with operators based on the ED strategy, the MOEA/D-HH model effectively reallocates the evaluation cost saved by operators based on the CX strategy, which would otherwise contribute minimally to the final evolutionary results. This saved computational cost is then allocated to operators based on the ED strategy. Simultaneously, offspring generated by operators based on the ED strategy can be reused by operators based on the CX strategy, further balancing their evaluation costs. The results of numerical experiments indicate that the operator switching mechanism based on Efficiency Inspection in MOEA/D-HH is highly effective, demonstrating significant advantages in certain test problems.

The structure of this doctoral dissertation closely parallels my research journey. It commenced by establishing a knowledge foundation related to the fundamental theory of multi-objective optimization problems. Subsequently, I engaged in the analysis, design, and modification of evolutionary operators. With a deeper understanding of the 'No Free Lunch' theory, the focus shifted towards implementing hybrid optimization algorithms within the MOEA/D framework. Furthermore, during the research on MOEA/D-HH, efforts were made to identify and explain the potential reasons for the superiority of hybrid algorithms. More than a mere conclusion for this paper, it serves as a summary of my doctoral research work. I sincerely and humbly hope that my research process and findings can provide inspiration and some insights for other students or researchers in this field.

## 6.2 Future issues of this research

This study still has some limitations. Firstly, the framework for solving multi-objective optimization problems using evolutionary computation methods is not limited to MOEA/D; other models such as NSGA-II, NSGA-III, and more are widely used in this field. However, in this study, other models were not introduced as comparative algorithms. Secondly, in the numerical experiments of MOEA/D-HH, we observed an excessive use of the IDE operator, and due to the low level of SRR for IDE, we suspect that the performance of MOEA/D-HH on the WFG_2D problem may be affected by this. Thirdly, although the WFG test suite is widely applied in the field of multi-objective optimization, it may not represent all scenarios, especially those in the real world that can be abstracted into multi-objective optimization problems. Fourthly, some studies suggest that as the number of objective functions increases, methods based on Pareto dominance may not effectively find solutions for multi-objective optimization problems. This is because, with an increasing number of objective functions, the conditions for determining a dominance relationship become more stringent. In other words, when the number of objective functions is large enough, any solution may not dominate or be dominated by other solutions.

Therefore, addressing the aforementioned limitations, future research may focus on the following aspects: Firstly, continuous optimization and modification of the MOEA/D-HH model to mitigate potential negative impacts caused by the excessive

use of the IDE operator. Comparative experiments could be conducted in subsequent trials, introducing advanced algorithms such as NSGA-III for benchmarking. Additionally, exploring alternative test tools beyond the WFG test suite, including the DTLZ suite, may offer a more comprehensive evaluation. Secondly, attempting to apply the MOEA/D-HH algorithm to address real-world problems or theoretical computations in other domains, such as the theoretical calculation of multi-physics field problems, facilitating the translation from theory to practical applications. Thirdly, staying abreast of the latest research developments in the field of multi-objective optimization, particularly seeking new methods that surpass Pareto dominance on many objective optimization problems.

# Acknowledgment

Seven years ago, when I set foot on the land of Japan, I carried with me a curiosity for the unknown and expectations for the future. The journey of these seven years has been a blend of enduring length and fleeting moments, marked by moments of confusion and adversity, as well as happiness and joy. Now, as I approach the completion of my doctoral journey, I stand on the threshold of a future that promises even greater challenges.

First and foremost, I wish to express my profound gratitude to my advisor, Associate Professor Shinya Watanabe. Throughout my academic pursuits, Professor Watanabe has been a steadfast pillar of support and guidance. His patient and meticulous mentoring not only contributed to my academic growth but also extended to unwavering care in my personal life. It is under his dedicated guidance that I have found the fortitude to navigate the academic path with greater determination.

Here, I extend my heartfelt thanks to my dearest mother. It is her boundless support, both spiritually and materially, that has enabled me to successfully complete my academic journey. Her selfless dedication and unwavering support have been the driving force behind my academic achievements.

I am grateful to all my colleagues at the Intelligent Computing Laboratory, especially the seniors and juniors in the Evolutionary Computation (EC) group. Their intelligence has sparked countless inspirations and their valuable advice has provided crucial assistance in my research. Within this close-knit community, I have experienced the strength of unity and collaboration.

Lastly, I wish to thank Mr. Guanming Shao and Professor Junichi Kisigami for providing me with the opportunity to pursue advanced studies at Muroran Institute of Technology. It is here that I have had the chance to delve into a broader spectrum of knowledge and explore deeper layers of research.

A special acknowledgment goes to Ms. Jianting Xu, whose encouragement has been my greatest source of inspiration, reinforcing my confidence in pursuing a doctoral degree. In moments of uncertainty, her unwavering support and encouragement have been the wellspring of strength propelling me forward.

As I reach the culmination of this journey, I feel a deep sense of gratification and reflection. This dissertation not only serves as a summary of the past seven years but is also an expression of gratitude to all those who have supported and accompanied me. Thank you, for it is you who have enriched my journey with vibrancy. In the future, I will continue to strive, channeling my acquired knowledge and experiences back into society and dedicating myself to the pursuit of my ideals.

# References


[1] Coello, C.A.C.; Lamont, G.B. Applications of Multi-Objective Evolutionary Algorithms; World Scientific: Singapore, 2008; pp. 154–196. https://doi.org/10.1142/5712.

[2] Deb, K. Multi-Objective Optimization Using Evolutionary Algorithms; Springer: London, UK, 2011. https://doi.org/10.1007/978-0-85729-652-8_1.

[3] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.

[4] Coello Coello C A. A comprehensive survey of evolutionary-based multiobjective optimization techniques[J]. Knowledge and Information systems, 1999, 1(3): 269-308.

[5] Cheng R, Jin Y, Olhofer M. Test problems for large-scale multiobjective and many-objective optimization[J]. IEEE transactions on cybernetics, 2016, 47(12): 4108-4121.

[6] Dunis C, Middleton P W, Karathanasopolous A, et al. Artificial intelligence in financial markets[M]. London: Palgrave Macmillan, 2016.

[7] Shapiro J. Genetic algorithms in machine learning[M]//Advanced Course on Artificial Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999: 146-168.

[8] Holland J H. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence[M]. MIT press, 1992.

[9] Eiben A E, Smith J E. Introduction to evolutionary computing[M]. Springer-Verlag Berlin Heidelberg, 2015.

[10] Fogel D B. Artificial intelligence through simulated evolution[M]. Wiley-IEEE Press, 1998.

[11] Das S, Suganthan P N. Differential evolution: A survey of the state-of-the-art[J]. IEEE transactions on evolutionary computation, 2010, 15(1): 4-31.

[12] Pant M, Zaheer H, Garcia-Hernandez L, et al. Differential Evolution: A review of more than two decades of research[J]. Engineering Applications of Artificial Intelligence, 2020, 90: 103479.

[13] Das S, Mullick S S, Suganthan P N. Recent advances in differential evolution–an updated survey[J]. Swarm and evolutionary computation, 2016, 27: 1-30.

[14] Wu G, Shen X, Li H, et al. Ensemble of differential evolution variants[J]. Information Sciences, 2018, 423: 172-186.

[15] Hauschild M, Pelikan M. An introduction and survey of estimation of distribution algorithms[J]. Swarm and evolutionary computation, 2011, 1(3): 111-128.

[16] Parallel Problem Solving from Nature-PPSN IV: International Conference on Evolutionary Computation. The 4th International Conference on Parallel Problem Solving from Nature Berlin, Germany, September 22-26, 1996. Proceedings[M]. Springer Science & Business Media, 1996.

[17] Estimation of distribution algorithms: A new tool for evolutionary computation[M]. Springer Science & Business Media, 2001.

[18] Zitzler E, Thiele L. Multiobjective optimization using evolutionary algorithms—a comparative case study[C]//International conference on parallel problem solving from nature. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998: 292-301.

[19] Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results[J]. Evolutionary computation, 2000, 8(2): 173-195.

[20] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on evolutionary computation, 2007, 11(6): 712-731.

[21] Črepinšek M, Liu S H, Mernik M. Exploration and exploitation in evolutionary algorithms: A survey[J]. ACM computing surveys (CSUR), 2013, 45(3): 1-33.

[22] Wolpert D H, Macready W G. No free lunch theorems for optimization[J]. IEEE transactions on evolutionary computation, 1997, 1(1): 67-82.

[23] Wolpert D H, Macready W G. No free lunch theorems for search[R]. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[24] Han J, Watanabe S. A New Hyper-Heuristic Multi-Objective Optimisation Approach Based on MOEA/D Framework[J]. Biomimetics, 2023, 8(7): 521.

[25] Das I, Dennis J E. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems[J]. Structural optimization, 1997, 14: 63-69.

[26] De Weck O L. Multiobjective optimization: History and promise[C]//Invited Keynote Paper, GL2-2, The Third China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems, Kanazawa, Japan. 2004, 2: 34.

[27] Li H, Zhang Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II[J]. IEEE transactions on evolutionary computation, 2008, 13(2): 284-302.

[28] Liang J, Ban X, Yu K, et al. A survey on evolutionary constrained multiobjective optimization[J]. IEEE Transactions on Evolutionary Computation, 2022, 27(2): 201-221.

[29] Storn R, Price K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of global optimization, 1997, 11: 341-359.

[30] Mallipeddi R, Suganthan P N, Pan Q K, et al. Differential evolution algorithm with ensemble of parameters and mutation strategies[J]. Applied soft computing, 2011, 11(2): 1679-1696.

[31] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE transactions on Evolutionary Computation, 2008, 13(2): 398-417.

[32] Mühlenbein H, Paass G. From recombination of genes to the estimation of distributions I. Binary parameters[C]//International conference on parallel problem solving from nature. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996: 178-187.

[33] Zitzler E, Brockhoff D, Thiele L. The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration[C]//Evolutionary Multi-Criterion Optimization: 4th International Conference, EMO 2007, Matsushima, Japan, March 5-8, 2007. Proceedings 4. Springer Berlin Heidelberg, 2007: 862-876.

[34] Tang L, Dong Y, Liu J. Differential evolution with an individual-dependent mechanism[J]. IEEE Transactions on Evolutionary Computation, 2014, 19(4): 560-574.

[35] Zhang J, Sanderson A C. JADE: adaptive differential evolution with optional external archive[J]. IEEE Transactions on evolutionary computation, 2009, 13(5): 945-958.

[36] Hansen N, Müller S D, Koumoutsakos P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)[J]. Evolutionary computation, 2003, 11(1): 1-18.

[37] Hansen N. The CMA evolution strategy: a comparing review[J]. Towards a new evolutionary computation: Advances in the estimation of distribution algorithms, 2006: 75-102.

[38] Linxin T, Congqian Q, Xiaoqing Z. TBS evaluation method for product conceptual design[J]. Chinese Journal of Construction Machinery, 2007, 3: 357-360.

[39] Price K, Storn R M, Lampinen J A. Differential evolution: a practical approach to global optimization[M]. Springer Science & Business Media, 2006.

[40] Babu B V, Jehan M M L. Differential evolution for multi-objective optimization[C]//The 2003 Congress on Evolutionary Computation, 2003. CEC'03. IEEE, 2003, 4: 2696-2703.

[41] Gämperle R, Müller S D, Koumoutsakos P. A parameter study for differential evolution[J]. Advances in intelligent systems, fuzzy systems, evolutionary computation, 2002, 10(10): 293-298.

[42] Pahner U, Hameyer K. Adaptive coupling of differential evolution and multiquadrics approximation for the tuning of the optimization process[J]. IEEE Transactions on magnetics, 2000, 36(4): 1047-1051.

[43] Mezura-Montes E, Velázquez-Reyes J, Coello C A C. Modified differential evolution for constrained optimization[C]//2006 IEEE International Conference on Evolutionary Computation. IEEE, 2006: 25-32.

[44] Cai Y, Wang J. Differential evolution with neighborhood and direction information for numerical optimization[J]. IEEE Transactions on Cybernetics, 2013, 43(6): 2202-2215.

[45] Beyer H G. The theory of evolution strategies[M]. Springer Science & Business Media, 2001.

[46] Hartikainen M, Miettinen K, Wiecek M M. PAINT: Pareto front interpolation for nonlinear multiobjective optimization[J]. Computational optimization and applications, 2012, 52: 845-867.

[47] Li G, Lin Q, Cui L, et al. A novel hybrid differential evolution algorithm with modified CoDE and JADE[J]. Applied Soft Computing, 2016, 47: 577-599.

[48] Wang Y, Cai Z, Zhang Q. Differential evolution with composite trial vector generation strategies and control parameters[J]. IEEE transactions on evolutionary computation, 2011, 15(1): 55-66.

[49] Zhang S X, Zheng S Y, Zheng L M. An efficient multiple variants coordination framework for differential evolution[J]. IEEE transactions on cybernetics, 2017, 47(9): 2780-2793.

[50] Tind J, Wiecek M M. Augmented Lagrangian and Tchebycheff approaches in multiple objective programming[J]. Journal of Global Optimization, 1999, 14: 251-266.

[51] Estimation of distribution algorithms: A new tool for evolutionary computation[M]. Springer Science & Business Media, 2001.

[52] Igel C, Hansen N, Roth S. Covariance matrix adaptation for multi-objective optimization[J]. Evolutionary computation, 2007, 15(1): 1-28.

[53] Zapotecas-Martínez S, Derbel B, Liefooghe A, et al. Injecting CMA-ES into MOEA/D[C]//Proceedings of the 2015 annual conference on genetic and evolutionary computation. 2015: 783-790.

[54] Sato H, Aguirre H E, Tanaka K. Controlling dominance area of solutions and its impact on the performance of MOEAs[C]//International conference on evolutionary multi-criterion optimization. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007: 5-20.

[55] Huband S, Barone L, While L, et al. A scalable multi-objective test problem toolkit[C]//Evolutionary Multi-Criterion Optimization: Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings 3. Springer Berlin Heidelberg, 2005: 280-295.

[56] Ishibuchi H, Masuda H, Nojima Y. A study on performance evaluation ability of a modified inverted generational distance indicator[C]//Proceedings of the 2015 annual conference on genetic and evolutionary computation. 2015: 695-702.

[57] Huband S, Hingston P, Barone L, et al. A review of multiobjective test problems and a scalable test problem toolkit[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(5): 477-506.

[58] Del Ser J, Benítez-Hidalgo A, J Nebro A, et al. jMetalPy: a Python Framework for Multi-Objective Optimization with Metaheuristics[J]. 2019.