Study on Acceleration for Evolutionary Computation

余, 俊

https://doi.org/10.15017/2534456

Doctoral Dissertation of Engineering

博士 (工学) 学位論文

Study on Acceleration for Evolutionary Computation

進化計算高速化に関する研究

2019 年 9 月

YU Jun

余　俊

Graduate School of Design
Kyushu University

九州大学大学院芸術工学府

Abstract

Evolutionary computation (EC) algorithms have attracted the attention of many researchers and have successfully solved lots of real-world applications thanks to their excellent characteristics, such as their robustness, simplicity and intelligence. As the real-world problems which need to be solved become more complicated, the demand for high performance EC algorithms grows. Finding ways of accelerating convergence with lower cost consumption has thus become one of the hottest topics in the EC community. In this dissertation, we track this topic while proposing several effective approaches for accelerating EC search from three research directions.

The first direction is to accelerate EC by using the estimated convergence point(s) as elite individual(s) to replace poorer individual(s) in the current population. Firstly, we propose two novel strategies to further improve the accuracy of an estimated convergence point, e.g., use historical information and introduce weights. To generalize the basic estimation method to various types of optimization problems, we propose a separation method for multimodal tasks to estimate multiple global/local optima and then use them to accelerate EC. We also apply the method to multi-objective tasks to speed up the construction of Pareto optimality. Finally, we investigate the feasibility of using an estimated convergence point to accelerate interactive EC and reduce user fatigue.

The second direction is to integrate new search mechanisms into EC algorithms to enhance their performance. We develop three powerful variants of the fireworks algorithm (FWA) for searching spaces efficiently, using a scouting strategy, an amplitude reduction strategy, and others. We combine the basic estimation method and FWA to accelerate its search and propose a distance-based exclusion strategy to develop a niche FWA. Moreover, we propose two competitive strategies into a differential evolution framework to accelerate the elimination of poor individuals and keep high potential individuals by increasing competition among individuals.

The third direction is to develop a new EC algorithm with a stronger performance. Vegetation evolution (VEGE) borrows from the survival and reproduction strategies used by various plants and simulates two different periods of vegetation, a *growth period* and a *maturity period*, repeatedly to find the global optimum, i.e. it is an algorithm which involves cyclically switching between exploitation and exploration. We analyze the effect of each component of VEGE and give some general experience for configuring the parameter settings. Based on our analysis results, we further improve the performance of VEGE by proposing two new strategies to increase the diversity of the population and accelerate convergence of individuals.

Contents

Chapter 1

Introduction

1.1 Background and Remaining Problems

Optimization is an enduring subject that has been explored and studied for a long time. As early as the 17th century, British scientist Newton invented calculus and proposed the concept of the extremum. In 1847, French scientist Cauchy proposed the steepest descent method to solve a system of nonlinear equations [29]. Subsequently, a series of traditional optimization methods, such as linear programming [31, 170] and nonlinear programming [177, 15], were proposed and received much attention. They did not, however, form an independent and systematic discipline until the 1930s. Thanks to the rapid development of technology, especially the rise of computer science, modern optimization theory and algorithms have been developed and formed a new discipline as an important branch of mathematics, which aims to satisfying not only all the design requirements but also to reduce computational costs.

The demands of a concrete application are always an important factor in promoting the development of science. In recent decades, the optimization problems encountered have become more complicated, consisting of complex characteristics, such as e.g. being non-differentiable, containing discontinuities, having constraints, noise, being dynamic, etc. They are difficult to transform into a mathematical model, and traditional methods e.g. exhaustive methods and calculus-based methods, are powerless to solve them successfully. Meanwhile, humans obtain lots of inspiration from nature and use meta-heuristic algorithms to solve these complex problems perfectly. Evolutionary computation (EC), as one of these branches, is also applied to various industrial applications and has achieved satisfactory results.

The ground-breaking work of EC began in the 1950s, and EC algorithms became popular in the late 1980s [78]. In the early stages of EC history, it simulated biological evolution and survival of the fittest repeatedly to find the global optimum, and several foundational EC algorithms were proposed, including the genetic algorithm [62, 154, 63, 150], genetic programming [49, 7, 84] and evolution strategy [82, 59, 16]. With continuous research, practitioners have also come up with other effective EC algorithms inspired by natural phenomena, human and social behavior and others. For example, Storn and Price proposed the well-known differential evolution (DE) [152], which uses differential information to perturb individuals such that they evolve toward an optimal area. There are many other EC algorithms which also have strong optimization capabilities, such as particle swarm optimization (PSO) [77], the fireworks algorithm (FWA) [161], the artificial bee colony algorithm (ABC) [41] and others [113, 184, 185, 52]. Some researchers classify these algorithms as swarm intelligence, but no such distinction is made in this dissertation because all these algorithms are population-based optimization techniques. So far, hundreds of EC algorithms have sprung up like mushrooms and shown a powerful optimization ability in solving engineering problems which are often expensive, large scale, multimodal, etc.

EC algorithms have similar optimization frameworks, but they have different optimization characteristics by simulating different mechanisms. In general, they generate the initial population randomly, then use iterative progress, e.g. the recombination operation, mutation operation and selection operation, to gradually evolve the population into areas of higher potential. Finally, the found optimum is output when the termination condition is satisfied. Such iterative processes form the core of EC algorithms, which can guarantee convergence and distinguish from others. The extensive success of EC algorithms has also prompted them to be extended to various application scenarios. Niching techniques, as a highly focused branch, extend EC algorithms to multimodal optimization problems, which can find multiple different optima during one trial run. There are many well-known niching methods that have been proposed in the past few decades, including clearing [119], restricted tournament selection [60], crowding [66], fitness sharing [53] and others [88, 93, 85]. Some researchers have applied EC algorithms to multi-objective optimization problems and developed a series of effective multi-objective evolutionary algorithms (MOEAs), which handle several conflicting objectives to be optimized simultaneously by maintaining a set of Pareto optimal solutions. So far, many famous MOEAs, e.g. NSGA-II [40], NSGA-III [39, 68] and MOEA/D [203], have been proposed and attracted much research effort. Additionally, practitioners have developed corresponding EC algorithms for various other types of problems, such as large-scale optimization; dynamic and uncertain enviroments; and robust optimization. As EC algorithms receive more and more attention, they also intersect with other disciplines, resulting in a variety of new research directions, e.g. evolved neural networks, evolutionary fuzzy system, evolutionary cyber physical system, and others.

Interactive EC (IEC) plays an important role in the EC community. It can embed human knowledge, experience, preference, or as a more general term, KANSEI, into EC algorithms. Usually, any EC algorithm can be developed into an IEC, and only use human subjective evaluations as the fitness to replace its evaluation function. The pioneering work for IEC was presented in 1986 by Prof. Dawkins [137]. Although IEC is a young research field, many researchers have devoted themselves and proposed various IEC types, including interactive evolution strategy [61], interactive genetic algorithm [26, 100], interactive genetic programming [145, 75], interactive particle swarm optimization [104] and others [159, 83]. IEC algorithms have also been applied to many real-world applications, such as industrial problems [79, 33], art education [107, 106], games and therapy [171, 110], geophysics [20, 19], and others [155, 158, 157]. Because humans are involved in the system optimization, IEC algorithms exhibit human sensibility, which makes them very successful at solving design problems. However, evaluation noise arising from human subjectivity cannot be avoided, and IEC user fatigue remains a major problem for IEC.

Although EC and IEC algorithms have shown great potential and powerful optimization capability for various complex problems, their performance still needs to be further improved to address new and emerging concrete problems. Some researchers have focused on the underlying EC algorithms, e.g. schema theories [38], Markov chains [57], problem difficulty [105] and performance models [54], to understand their optimization principles. Others have concerned themselves with the application level; for example, they have analyzed optimization problems and then selected an appropriate EC algorithm for optimization [5, 102], hybrid multiple EC algorithms to obtain stronger robustness and wider applicability [160, 3]. In summary, how to improve the performance of the EC and reduce IEC user fatigue are the primary topics and study subjects which we will focus on in this dissertation.

1.2 Objectives and Approaches

The primary focus of this dissertation is to improve the performance of EC algorithms, i.e. obtain faster convergence and higher convergence accuracy with lower computational cost consumption, and a small amount of attention is discussing on a possibility of reducing IEC user fatigue. We try to achieve these objectives by pursuing research in three directions which builds on the theoretical principles underlying EC/IEC algorithms. Multiple approaches are explored in each research direction, and Figure 1.1 shows the relationship among them.

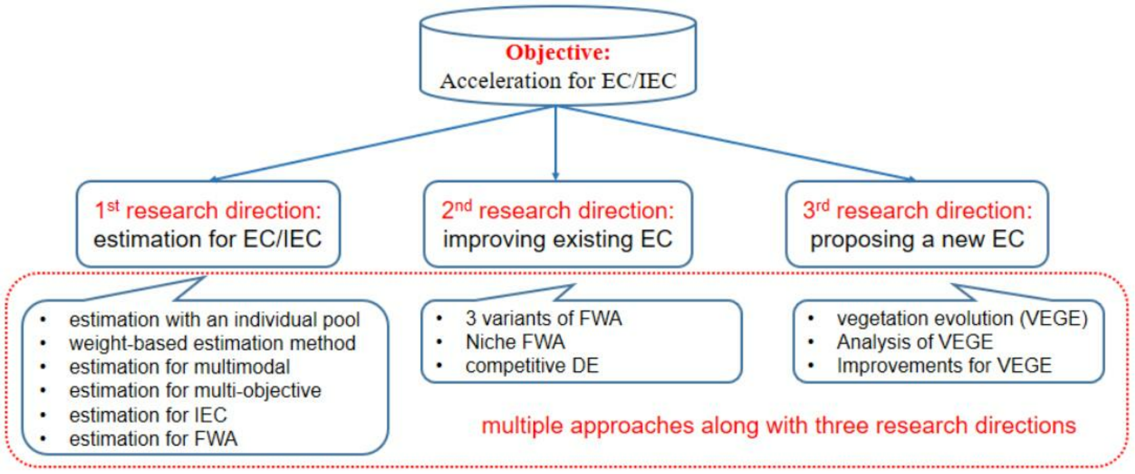


Figure 1.1: A relation diagram of multiple approaches along with three research directions to accelerate EC/IEC.

Three research directions of this work are:

1. using evolutionary information between two generations to estimate convergence points of the population, then using them to accelerate EC/IEC,

2. proposing new search strategies and schemes to enhance the performance of existing EC algorithms,
3. developing a new powerful EC algorithm inspired by vegetation growth and analyze its components to understand its optimization mechanisms.

The first research direction focuses on how to calculate the convergence points of the population mathematically and use them to accelerate convergence speed, which can be regarded as a reuse of the fitness landscape information to assist EC/IEC search. As a preliminary attempt, we propose two novel strategies to further improve the accuracy of an estimated convergence point. Since the basic estimation method is suitable for unimodal optimization, we are committed to extending it to accelerate various application scenarios, such as multimodal optimization, multi-objective optimization and IEC. Furthermore, we propose a type of fireworks, *synthetic fireworks*, which allows us to use the estimation method to accelerate FWA.

The second research direction focuses on improving the performance of two existing EC algorithms, FWA and DE, by integrating new search strategies. According to the characteristics of the explosion operations of FWA, we propose two novel explosion schemas to enhance the local search ability of fireworks, some other effective strategies to select the next generation and tune FWA parameters, and extend FWA to multimodal optimization by integrating a proposed distance-based exclusive strategy to find multiple optima. Additionally, we introduce two competitive strategies into DE to increase competition among individuals and accelerate its convergence.

The third research direction focuses on developing a new optimization framework inspired by natural plants to implement a new EC algorithm, vegetation evolution (VEGE). VEGE simulates vegetation growth and seed dispersion repeatedly to find the global optima. We investigate and analyze the impact of each component of the VEGE and summarize several general rules with regards to its parameter settings. Based on the analysis results, we propose two new strategies to improve the VEGE performance by increasing the diversity and accelerating growth of individuals. Finally, we point out some potential research topics which are for open discussion.

The core objective of this dissertation is to accelerate the performance of EC algorithms. The term "accelerated" here means that our proposal can find the global optimum with less fitness evaluations or find a better solution with the same number of fitness evaluations. If our proposed strategies have statistically better performance than previous methods, we can say that our proposed strategies have succeeded in achieving an accelerated effect.

1.3 Chapter Structure

Following this introductory chapter, we present an overview of related EC techniques, detailed implementation mechanisms of DE, PSO and FWA, as well as some published works about accelerating EC/IEC in Chapter 2. We propose two strategies to further improve the accuracy of an estimated convergence point based on the basic estimation method in Chapter 3. We combine the estimation method and EC/IEC together to speed up the convergence of EC algorithms and apply the basic estimation method into multimodal tasks, multi-objective tasks, and IEC tasks in

Chapter 4. We propose a new type of fireworks to combine the basic estimation method and FWA together to speed up the convergence of FWA in Chapter 5. We propose several novel strategies for FWA to enhance its performance and develop it to niche FWA in Chapters 6 and 7, respectively. We introduce competitive strategies into DE to speed up its convergence in Chapter 8. In Chapter 9, we propose a new EC algorithm inspired by vegetation growth and seed dispersion and analyze its performance, and a improved version is also presented by introducing two proposed strategies to further improve performance. Finally, we conclude our works and suggest several potential research topics for discussion in Chapter 10.

From the structural perspective, Chapters 3 to 9 form the innovative parts of this dissertation, which aims to accelerate EC/IEC from three different research directions. Figure 1.2 illustrates the chapter structure of this dissertation graphically.

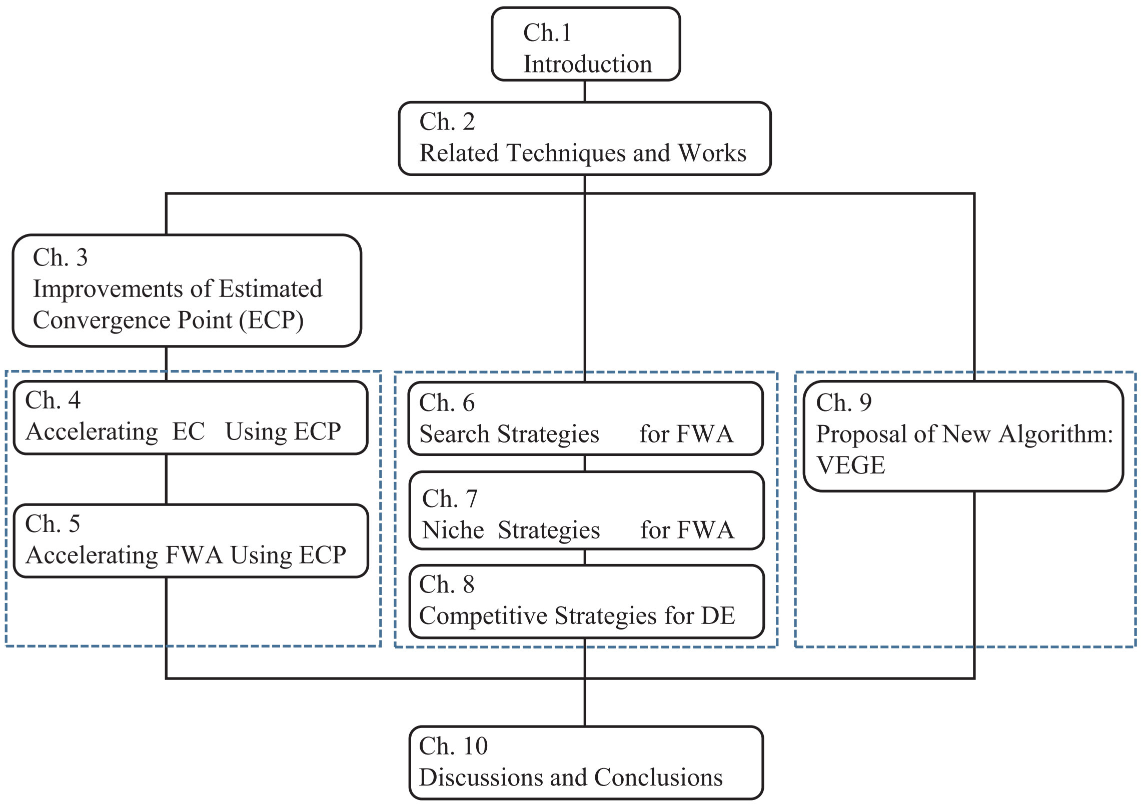

Ch.1
Introduction
Ch. 2
Related Techniques and Works
Ch. 3
Improvements of Estimated Convergence Point (ECP)
Ch. 4
Accelerating EC Using ECP
Ch. 5
Accelerating FWA Using ECP
Ch. 6
Search Strategies for FWA
Ch. 7
Niche Strategies for FWA
Ch. 8
Competitive Strategies for DE
Ch. 9
Proposal of New Algorithm: VEGE
Ch. 10
Discussions and Conclusions

Figure 1.2: The chapter structure of this dissertation.

Chapter 2

Related Techniques and Research

2.1 Related Techniques

2.1.1 Evolutionary Computation

On the Origin of Species, [34] published by Charles Darwin in 1859, is considered to be the foundation of evolutionary biology. It was Darwin who first put forward the theory of evolution and attempted to prove that the evolution of species is achieved through natural selection and the survival of the fittest. The core concept is that variation generates diverse individuals, i.e. gene mutations cause individuals to have different phenotypes, for natural selection. Although the probability of gene mutation in any given individual is quite low, the large population base ensures that there are enough mutated individuals to generate. Because of the randomness and uncertainty of variation, the emerging characteristics of variant individuals may or may not adapt to the current environment. When their living environment changes, some variant individuals may adapt to the new environment and survive, and unadaptable individuals are eliminated by natural selection. The survivors can get more resources and tend to generate more offspring, enabling them to spread their genes more rapidly in the population. Species can survive well and evolve stronger capabilities in the face of volatile environments thanks to the diversity of their populations.

EC algorithms simulate the above processes inspired by biological evolution, i.e., mutation, crossover, and selection, to find the global optimum. With the popularity of EC algorithms, practitioners also get inspiration from other phenomena and model these mechanisms to solve various complex optimization problems. For example, crystals can be heated to a high temperature and then slowly cooled to reduce defects in the crystal during this process and minimize the system energy, a process which is similar to optimization. Thus, simulated annealing [81] simulates the annealing in metallurgy and introduces random factors to increase the probability of obtaining the optimum. In the 19th and 20th centuries, many countries in the West started their colonial eras and thus transported their cultures to the countries they colonized. Throughout, the imperialist countries also competed with each other for colonization. Inspired by these social behaviors, the imperialist competitive algorithm [10] simulates the colonial assimilation mechanism and the imperial competition mechanism repeatedly to find the global optimum. So far, hundreds of remarkable EC algorithms have been proposed inspired by a variety of creatures or phenomena, which can be categorized as population-based stochastic problem solvers. Fig. 2.1 illustrates the general optimization framework used by EC algorithms.

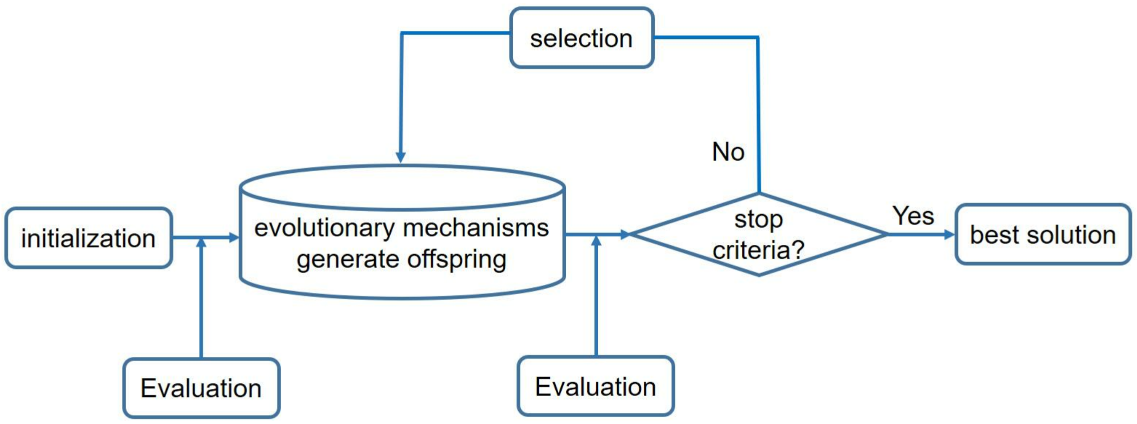


Figure 2.1: The general optimization framework of EC algorithms.

- **Initialization:** The term "individual" used in biology is also used in EC algorithms, and together all the individuals form a population. Generally, a candidate solution is encoded as an individual regardless of whether an optimization task is a combination problem or a continuous problem, then random initialization is widely used to generate the initial population at the beginning of an EC algorithm. Some practitioners have also proposed other initialization methods, such as opposition-based initialization [164, 133], clustering-based initialization [13] and others [76], to improve the convergence accuracy of EC algorithms.
- **Evaluation:** We use the terms "fitness function", "evaluation function" and "objective function" for those functions which evaluate the quality of individuals. These evaluations can be performed using mathematical equations, fuzzy logic systems, artificial neural networks, etc. In the case where human subjectivity is used to evaluate the individuals, the optimization algorithms can be regarded as IEC.
- **Evolutionary mechanisms:** This part forms the core of the EC algorithms and directly affects their performance. Because EC algorithms use evolution-inspired optimization principles iteratively to evolve the population, in this dissertation we use the term "evolutionary mechanisms" to refer to their core operations, such as crossover and mutation in genetic algorithms [175]. Although EC algorithms have different optimization characteristics, the common theme is that they balance the capabilities of *exploration* and *exploitation* to efficiently generate potential offspring. In short, the ultimate objective is to gradually improve the initial population and eventually converge to the global optimum.
- **Stop criteria:** There are two universal criterions used for terminating EC algorithms: (1) a predetermined fixed convergence accuracy; (2) a predetermined maximum number of evaluations. Usually these two criteria are used

together; EC algorithms are terminated once either one of the two is satisfied.

- **Selection:** This operation removes less desired individuals and selects individuals with potential for the next generation to improve the overall quality of the population. It is also one of the factors that affect EC performance because inappropriate selection methods may slow down or even hinder convergence. So far, many effective selection methods have been proposed, such as, tournament selection [22], stochastic universal sampling [162], linear ranking selection [18], and others [176].
- **Best solution:** The found optimum is output when the EC algorithm ends.

2.1.2 Estimation Method of a Convergence Point

The estimation method calculates a convergence point of the population mathematically using the information of two subsequent generations rather than multiple iteration [103]. Since all individuals evolve and converge towards optimal area in unimodal tasks, the estimation method tries to calculate the point to minimize total distance from it to all moving vectors. Here, a moving vector is defined as pointing from a poor parent to its better offspring (as shown in the Figure 2.2).

Let us begin by defining symbols. $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$ in the Figure 2.2 are the i-th parent individual and its offspring individual, respectively ($\boldsymbol{a}_i, \boldsymbol{c}_i \in \mathbb{R}^d$). Then, the i-th moving vector is defined as the direction vector, $\boldsymbol{b}_i = \boldsymbol{c}_i - \boldsymbol{a}_i$. The unit direction vector of the $\boldsymbol{b}_i$ is given as $\boldsymbol{b}_{0i} = \boldsymbol{b}_i/||\boldsymbol{b}_i||$, i.e. $\boldsymbol{b}_{0i}^T\boldsymbol{b}_{0i} = 1$.

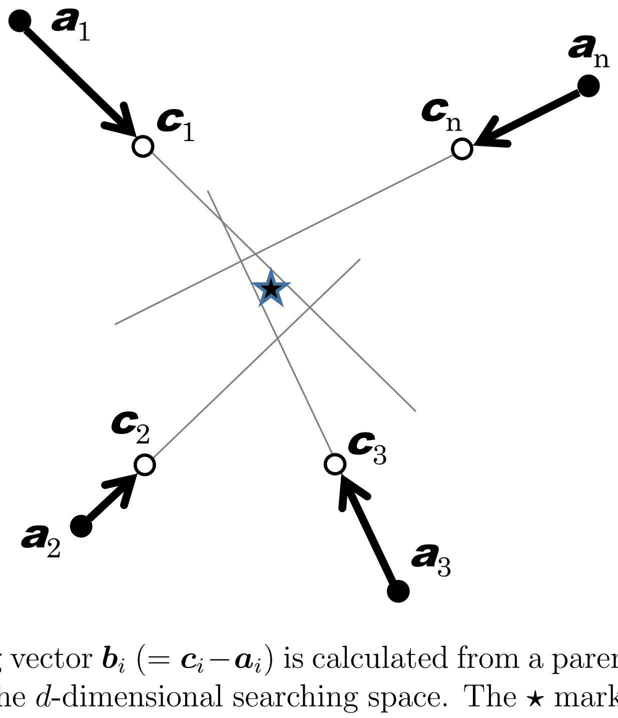


Figure 2.2: Moving vector $\boldsymbol{b}_i$ ($= \boldsymbol{c}_i - \boldsymbol{a}_i$) is calculated from a parent individual $\boldsymbol{a}_i$ and its offspring $\boldsymbol{c}_i$ in the d-dimensional searching space. The $\star$ mark is the convergence point for these moving vectors.

Estimation Method

Let $\boldsymbol{x} \in \mathbb{R}^d$ be the point that is the nearest to the n extended directional line segments, $\boldsymbol{a}_i + t_i\boldsymbol{b}_i$ ($t_i \in R$). By *nearest*, we mean that the total distance from $\boldsymbol{x}$ to the n extended directional line segments, $\mathrm{J}(\boldsymbol{x}, \{t_i\})$ in Eq.(2.1), becomes the minimum.

As the minimum line segment from the convergence point $\boldsymbol{x}$ to the extended directional line segments is the orthogonal projection from $\boldsymbol{x}$, we may insert an orthogonal condition, Eq. (2.2), into the Eq. (2.1) and thus remove t_i.

$$min\,\{\mathrm{J}(\boldsymbol{x}, \{t_i\})\} = \sum_{i=1}^{n} \|\boldsymbol{a}_i + t_i\boldsymbol{b}_i - \boldsymbol{x}\|^2 \tag{2.1}$$

$$\boldsymbol{b}_i^{\mathrm{T}}(\boldsymbol{a}_i + t_i\boldsymbol{b}_i - \boldsymbol{x}) = 0 \quad \text{(orthogonal condition)} \tag{2.2}$$

Then, we can obtain the Eq. (2.3).

$$\mathrm{J}(\boldsymbol{x}) = \sum_{i=1}^{n} \left\| \boldsymbol{a}_i + \frac{\boldsymbol{b}_i^T(\boldsymbol{x} - \boldsymbol{a}_i)}{\|\boldsymbol{b}_i\|^2}\boldsymbol{b}_i - \boldsymbol{x} \right\|^2 \tag{2.3}$$

From the $\|\cdot\|$ part of the Eq. (2.3) to obtain the below Eq. (2.4).

$$\boldsymbol{a}_i + \frac{\boldsymbol{b}_i^T(\boldsymbol{x} - \boldsymbol{a}_i)}{\|\boldsymbol{b}_i\|^2}\boldsymbol{b}_i - \boldsymbol{x} = \left\{ \frac{\boldsymbol{b}_i\boldsymbol{b}_i^T}{\|\boldsymbol{b}_i\|^2} - \boldsymbol{I}_d \right\} (\boldsymbol{x} - \boldsymbol{a}_i) \tag{2.4}$$

where I_d is an identity matrix. Here, let us define H_i as:

$$\frac{\boldsymbol{b}_i\boldsymbol{b}_i^T}{\|\boldsymbol{b}_i\|^2} - \boldsymbol{I}_d = \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^T - \boldsymbol{I}_d = \boldsymbol{H}_i \tag{2.5}$$

Next, we can obtain the below objective function from Eq. (2.3) where we have eliminated the term $\{t_i\}$.

$$\begin{aligned}\mathrm{J}(\boldsymbol{x}) &= \sum_{i=1}^{n} \|\boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i)\|^2 = \sum_{i=1}^{n} \|\boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i)\| \, \|\boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i)\| \\ &= \sum_{i=1}^{n} \{\boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i)\}^T \boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i) = \sum_{i=1}^{n} (\boldsymbol{x} - \boldsymbol{a}_i)^T \boldsymbol{H}_i^T \boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i)\end{aligned} \tag{2.6}$$

Our goal is obtained by minimizing $\boldsymbol{J(x)}$ with regard to $\boldsymbol{x}$. Estimated $\boldsymbol{x}$, i.e. $\hat{\boldsymbol{x}}$, is obtained by partially differentiating each element of $\boldsymbol{x}$ and setting them equal 0.

$$\begin{aligned}\frac{\partial \boldsymbol{J(x)}}{\partial \boldsymbol{x}} &= 2\sum_{i=1}^{n} \boldsymbol{H}_i^T \boldsymbol{H}_i(\boldsymbol{x} - \boldsymbol{a}_i) \\ &= 2\left\{ \left(\sum_{i=1}^{n} \boldsymbol{H}_i^T \boldsymbol{H}_i \right) \boldsymbol{x} - \left(\sum_{i=1}^{n} \boldsymbol{H}_i^T \boldsymbol{H}_i \boldsymbol{a}_i \right) \right\} = 0\end{aligned} \tag{2.7}$$

Thus, the estimation of Eq. (2.8) is obtained.

$$\hat{\boldsymbol{x}} = \left(\sum_{i=1}^{n} \boldsymbol{H}_i^T \boldsymbol{H}_i \right)^{-1} \left(\sum_{i=1}^{n} \boldsymbol{H}_i^T \boldsymbol{H}_i \boldsymbol{a}_i \right) \tag{2.8}$$

Since $\boldsymbol{H}_i$ has the characteristic of $\boldsymbol{H}_i^T \boldsymbol{H}_i = \boldsymbol{H}_i^2 = \boldsymbol{H}_i$,i.e. a projection matrix, the Eq. (2.8) can be rewritten as the Eq. (2.9).

$$\hat{\boldsymbol{x}} = \left(\sum_{i=1}^{n} \boldsymbol{H}_i \right)^{-1} \left(\sum_{i=1}^{n} \boldsymbol{H}_i \boldsymbol{a}_i \right) \tag{2.9}$$

Finally, an estimated convergence point is calculated using following Eq. (2.10).

$$\hat{\boldsymbol{x}} = \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \right\}^{-1} \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \tag{2.10}$$

Approximative Estimation Method 1

Murata et. al [76] also provided approximative calcuration methods for programming language that cannot handle vectors and matrixes. Here, the Eq. (2.10) is expanded as a Neumann series and approximate it by using only the lower order terms. Thus, it is expanded as shown in the Eq.(2.11).

$$\begin{aligned} \hat{\boldsymbol{x}} &= \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \right\}^{-1} \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \\ &= \frac{1}{n} \left\{ \boldsymbol{I}_d + \left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) + \left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right)^2 + \ldots.. \right\} \times \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \end{aligned} \tag{2.11}$$

From the Eq. (2.11), we can notice that the lower the order used, the less the inverse matrix is calculated. When only the 0th order term is used, then it becomes the Eq. (2.12).

$$\hat{\boldsymbol{x}} \approx \frac{1}{n} \times \boldsymbol{I}_d \times \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \approx \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{a}_i - \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \boldsymbol{a}_i \tag{2.12}$$

where, $\boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^T$ is matrix, while $\boldsymbol{a}_i$ is a vector. Thus, we can do the following transformation to avoid matrix operations.

$$\boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \boldsymbol{a}_i = \boldsymbol{b}_{0i} (\boldsymbol{b}_{0i}^{\mathrm{T}} \boldsymbol{a}_i) = \boldsymbol{b}_{0i} (\boldsymbol{a}_i^T \boldsymbol{b}_{0i}) = (\boldsymbol{a}_i^T \boldsymbol{b}_{0i}) \boldsymbol{b}_{0i} \tag{2.13}$$

By insterting the Eq. (2.13) into the Eq. (2.12), we can obtain the following approximate equation as shown in the Eq. (2.14).

$$\hat{\boldsymbol{x}} \approx \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{a}_i - \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{a}_i^T \boldsymbol{b}_{0i}) \boldsymbol{b}_{0i} \tag{2.14}$$

Approximative Estimation Method 2

Let us consider an iterative method for alternately minimizing $\boldsymbol{x}$ and t_i in the Eq. (2.1), As the Neumann series expansion is performed at the convergence point of this method, we can say that this method finds an approximated solution by expanding the Neumann series in the previous approximative estimation method.

When a point $\boldsymbol{x}$ is given, let $\boldsymbol{y}_i(\boldsymbol{x})$ be the nearest point on the i-th line to $\boldsymbol{x}$, i.e. the projection of $\boldsymbol{x}$ on the i-th line.

$$\boldsymbol{y}_i(\boldsymbol{x}) = \boldsymbol{a}_i + t_i(\boldsymbol{x})\boldsymbol{b}_i \tag{2.15}$$

where $t_i(\boldsymbol{x})$ is given in the Eq. (2.16) from the Eq. (2.2).

$$t_i(\boldsymbol{x}) = \frac{\boldsymbol{b}_i^T(\boldsymbol{x} - \boldsymbol{a}_i)}{\|\boldsymbol{b}_i\|^2} \tag{2.16}$$

On the other hand, when $\boldsymbol{y}_i(\boldsymbol{x}), (i = 1, 2, ..., n)$ is given, let $\boldsymbol{x}^{'}$ be the point for which the total sum of distances to them is the shortest. Then, $\boldsymbol{x}^{'}$ is the gravity point of $\boldsymbol{y}_i(\boldsymbol{x})$ and is given by the Eq.(2.17).

$$\boldsymbol{x}^{'} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{y}_i(\boldsymbol{x}) \tag{2.17}$$

According to the above, the following iterative method is obtained:

Step1: Initialize $\boldsymbol{x}$. For example, $\boldsymbol{x} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{c}_i$.
Step2: Obtain $t_i(x)$ by the Eq.(2.16) and then $y_i(x)$ by the Eq.(2.15).
Step3: Replace x with $x^{'}$, which is the gravity point obtained by the Eq. (2.17).
Step4: Go to Step 2 until convergence is reached.

Because of the update of the Eq. (2.15), the below Ed. (2.18) is also established. Thus, it is guaranteed that the iterative method in this section converges.

$$\sum_{i=1}^{n} \|\boldsymbol{y}_i(\boldsymbol{x}) - \boldsymbol{x}^{'}\|^2 \leqslant \sum_{i=1}^{n} \|\boldsymbol{y}_i(\boldsymbol{x}) - \boldsymbol{x}\|^2 \tag{2.18}$$

2.1.3 Differential Evolution

DE is a type of population-based EC algorithm used to solve real-valued continuous optimization problems [152, 123, 124]. The optimization process of DE can be summarized as follows: a differential vector consisting of two distinct individuals is selected from the population to perturb a randomly selected individual as a *base vector* to generate a *mutation vector*. Next, DE applies the crossover operation between the *mutation vector* and the *target vector*, i.e. a parent individual, to generate a *trial vertor* as a new offspring individual. Subsequently, the winner between the *target vector* and the *trial vertor* survives to the next generation. Thus, each individual in the population is used as a *target vector* and performs the above operation once to generate an offspring individual. When all the individuals have performed the operation, the update of the generation is complete. DE evolves the population iteratively until a termination condition is satisfied. Figure 2.3 illustrates the optimization principles of DE.

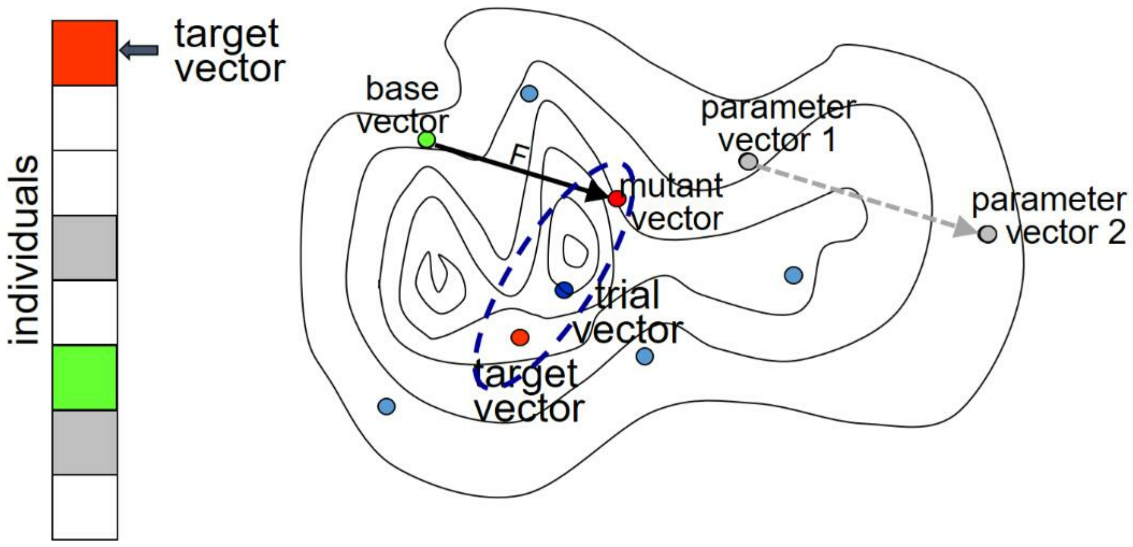


Figure 2.3: DE uses differential vectors to improve the quality of the population.

The novelty of the original DE was to find the global optimum by applying a differential-based simple mutation operation for generating offspring candidates and a one-to-one selection strategy for deciding surviving individuals. The idea of differential vectors is quite simple, but these directional vectors include the distribution information of the population. Suppose we collect all ${}_P\mathrm{C}_2$ differential vectors obtained from the P population size, and put their initial points on a *base vector*. Then, the distribution of their terminal points becomes similar to that of population. The lengths of the differential vectors becomes shorter according to the convergence of the population, but the distribution information of population is maintained.

Due to DE being both simple and highly efficient, practitioners have dedicated efforts to further improve DE performance and have achieved gratifying results. Thanks to their tireless efforts, DE has become one of the most popular algorithms in the EC community, and a large number of effective variants have been proposed, such as, adaptive DE [201], self-adaptive DE [127], opposition-based DE [132], and others [116, 37, 36]. Meanwhile, DE is also applied to various scenarios including large scale problems [186, 199], combination optimization [179, 56], dynamic optimization [97, 24], constrained optimization [202, 98], multi-objective optimization [99, 139], and multi-modal problems [163, 130]. Besides, DE has solved many complex real-world applications and achieved great success because it does not need any prior knowledge of the optimization problems. For example, in power plant control [169], clustering[35], electromagnetics [140], scheduling flow shops [108], engineering design [92], etc. In short, DE has attracted considerable attention, not only in academia, but also in industry.

We usually use DE/x/y/z to represent any form of DE, where x means the individual to be perturbed, e.g. random, best or current to $p$best; y means the number of difference vectors used in the perturbation of x; z means the crossover operator, and here there are two commonly used methods: *bin* and *exp*. Algorithm 1 shows the general framework of DE.

Algorithm 1 The general framework of the original DE. F: scaling factor, CR: crossover rate, PS: population size.

```
1: Initialize population randomly.
2: Evaluate the population.
3: while the termination condition is not satisfied do
4:   for $i = 1$ to $PS$ do
5:     Select an individual randomly as a base vector, $x_{base}$.
6:     Select two individuals randomly to construct a difference vector, $(x_{r1} - x_{r2})$,
       where $r1 \neq r2 \neq base \neq i$.
7:     Generate a mutation vector, $x_{mutation} = x_{base} + F \times (x_{r1} - x_{r2})$.
8:     Crossover $x_{mutation}$ and $x_i$ to generate a trial vector, $x_{trial}$.
9:     Evaluate the fitness of the trial vector, $x_{trial}$.
10:    The winner of $x_{trial}$ and $x_i$ can survive to the next generation.
11:  end for
12: end while
13: Output the found optimum.
```

2.1.4 Particle Swarm Optimization

PSO simulates the rules underlying bird flocks foraging behavior to find the global optimum and opened a new page in the research of developing EC algorithms inspired by social behaviors [77]. Although the behavior of an individual is simple and limited, cooperation with other individuals can lead to highly complex and intelligent performance. With PSO, a real bird is abstracted as a point with no mass and shape, i.e. a *particle*, and each particle has a velocity with which its position is then updated. PSO performs the above operations iteratively until a termination condition is satisfied.

The feature of the original PSO is to use the historical information of the particles themselves and the shared global optimal information to update their velocity. Figure 2.4 shows that the speed update of a particle is affected by three factors, (1) the current moving direction of the particle, (2) the past best position of the particle, (3) the past best position of all particles. Thus, each particle uses Eqs. (2.19) and (2.20) to update its velocity and position, respectively. Algorithm 2 outlines the general framework of PSO.

$$\boldsymbol{v}_i(t+1) = \omega \times \boldsymbol{v}_i(t) + c_1 \times rand() \times (\boldsymbol{pbest} - \boldsymbol{x}_i) + c_2 \times rand() \times (\boldsymbol{gbest} - \boldsymbol{x}_i) \quad (2.19)$$

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) \quad (2.20)$$

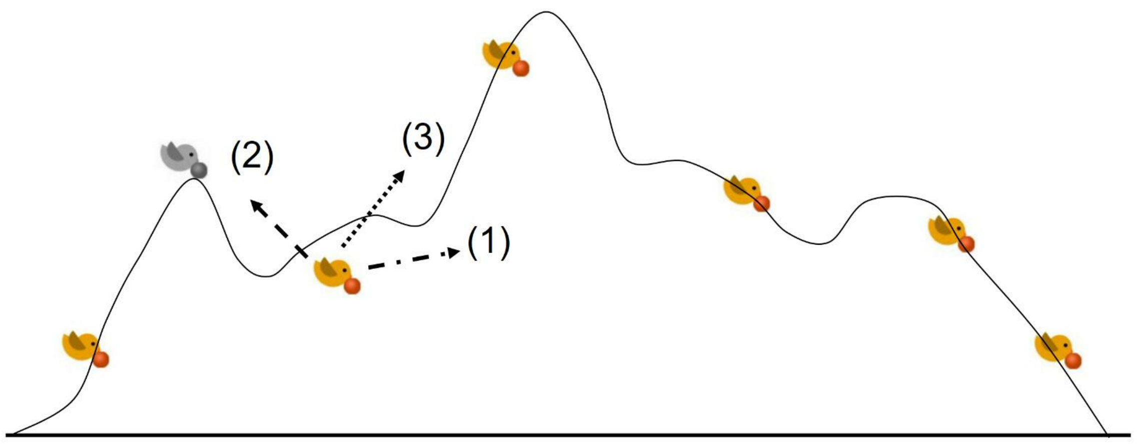


Figure 2.4: The velocity update of a particle.

where $\boldsymbol{v}_i(t)$ and $\boldsymbol{x}_i(t)$ are the velocity and position of i-th particle in the t generation, respectively. $\boldsymbol{v}_i(t)$ locates in $(\boldsymbol{v}_{min}, \boldsymbol{v}_{max})$ to limit the update speed so it is neither too large or too small. ***pbest*** means the past best position of i-th particle and ***gbest*** is the found best position so far. $rand()$ generates a random number between $(0, 1)$. ω is the inertia factor, c_1 and c_2 are learning factors and they are usually set to constants.

Algorithm 2 The general framework of the original PSO. PS: population size.

```
 1: Initialize the velocity and position of the population randomly.
 2: Evaluate the population.
 3: while the termination condition is not satisfied do
 4:     for $i = 1$ to $PS$ do
 5:         Update the velocity of $i$-th particle using the Eq. (2.19).
 6:         Update the position of $i$-th particle using the Eq. (2.20).
 7:         Evaluate the $i$-th particle.
 8:     end for
 9: end while
10: Output the found optimum.
```

Because PSO is simple and easy to use with few parameters, it has attracted significant attention and many powerful variants have been proposed so far. Some practitioners investigated the impact of parameter settings on performance [47, 136, 181]. Others introduced many novel strategies, e.g. fuzzy adaptive [144], linear time varying [209], acceleration coefficients [69], into the original PSO to control parameters reasonably and achieve better performance. Meanwhile, PSO has also been applied to various scenarios, including multimodal optimization [129, 90], dynamic optimization [182, 43], multi-objective optimization [180, 65], and hybridise with other EC algorithms to improve PSO performance by inheriting their advantages, such as GA-PSO [138], DEPSO [205], PSOSA [206], PSACO [143]. In addition, it has also been applied to various complex real-world applications successfully, such as, economic dispatch [50], PID controller design [51], optimal power flow [2], feature selection

[174] and others [45, 70, 6]. To summarize, PSO is one of the foundational works in the EC community which has not only demonstrated great potential itself, but has also promoted the development of subsequent social behavior-based EC algorithms.

2.1.5 Fireworks Algorithm

In real world, a firework is launched into the sky then many sparks are generated around it, which can be abstracted as a local search model around a specific point (firework). Inspired by these explosion process, FWA generates multiple initial fireworks individuals randomly, then each firework individual generates different number of spark individuals within different explosion amplitude according to its fitness. Subsequently, the best individual survivals to the next generation and remaining individuals in the next generation are selected from the current firework individuals and spark individuals based on their selection probability. Thus, FWA simulates the explosion process iteratively until a termination condition is satisfied. The Figure 2.5 demonstrates a general explosive model of the original FWA.

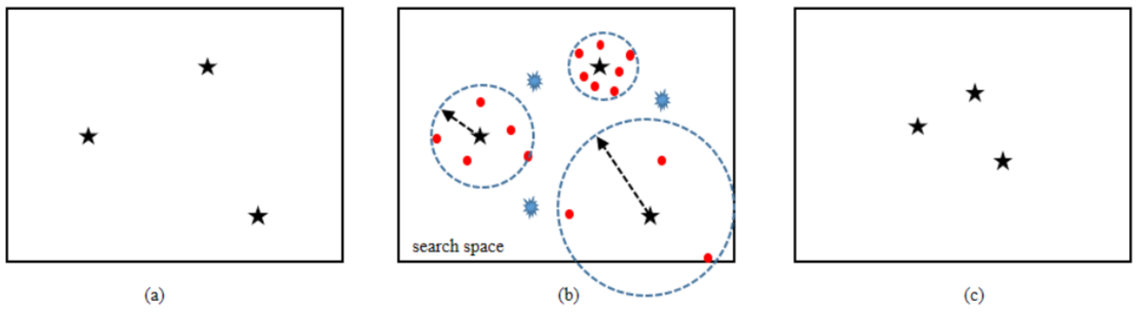


Figure 2.5: Search process of FWA. (a) Fireworks are random generated, (b) sparks (red solid poins) are generated around each firework, and mutation sparks (blue solid poins) are also generated, (c) new fireworks are selected to enter the next generation using the (b). The (b) and (c) are iterated until a termination condition is satisfied.

The feature of the original FWA is to assign different explosion amplitude and number of generated spark individuals to each firework individual to balance exploitation and exploration. There are two types of firework individuals in the original FWA: better firework and poor firework. A better firework is responsible for exploitation and may close to an optimum so that many sparks are generated around it within a smaller range, while a poor firework is responsible for exploration and may far from the optimum so a few sparks are generated and search range should be larger. The Algorithm 3 shows the general flow of FWA mainly consisting of three operations: explosion, mutation and selection.

- **Explosion Operation:** This operation is design to determine the explosion parameters of each firework individual, i.e. the number of generated spark individuals and a explosion amplitude. First of all, let's define symbols. For the minimization problem, x represents a feasible solution in the space. $f(x)$ is fitness function. n represents the number of fireworks in each generation. y_{max} and y_{min} denotes the worst and the best fireworks in current generation.

Algorithm 3 The general framework of FWA.

```
1: Initialize $n$ fireworks randomly.
2: Evaluate the fitness of each firework.
3: while a termination condition is not satisfied do
4:     Generate sparks for each firework.
5:     Generate Gaussian mutation sparks.
6:     if sparks are generated outside search area then
7:         Use a mapping rule to bring it back to search space.
8:     end if
9:     Evaluate the fitness of generated sparks.
10:    Select $n$ new fireworks for the next generation.
11: end while
12: Output found feasible solution.
```

ξ denotes the smallest constant in the computer, is utilized to avoid zero-division-error.

We use the Eq. (2.21) to calculate the generated spark individuals for each firework and the Eq. (2.22) to avoid generate too much or too little spark individuals.

$$s_i = m * \frac{y_{max} - f(x_i) + \xi}{\sum_{i=1}^{n}(y_{max} - f(x_i)) + \xi} \tag{2.21}$$

$$s_i^{'} = \begin{cases} round(a * m) & if \quad s_i < a \times m \\ round(b * m) & if \quad s_i > b \times m \\ round(s_i) & otherwise \end{cases} \tag{2.22}$$

where, m is a parameter controlling the total number of sparks generated by the n fireworks, i means the i-th fireworks, a and b are const parameters.

We use the Eq. (2.23) to calculate a explosion amplitude for i-th firework individual.

$$A_i = A_{max} * \frac{f(x_i) - y_{min} + \xi}{\sum_{i=1}^{n}(f(x_i) - y_{min}) + \xi} \tag{2.23}$$

where A_{max} denotes the maximum explosion amplitude.

Finally, spark individuals may undergo the effects of explosion from random selected z directions, and we obtain the number of the affected directions randomly using the Eq. (2.24). Then, all affected directions use the same displacement, $h = A_i \times rand(-1, 1)$, to generate spark individuals, x_{new}, with the Eq. (2.26).

$$z = round(d * rand(0, 1)) \tag{2.24}$$

$$x_{new} = x_i^z + h \tag{2.25}$$

where d means dimensionality, $rand(0,1)$ is an uniform distribution over [0,1]. We call the generated sparks by this way, explosion sparks.

- **Gaussian Mutation:** It is designed to increase the diversity of spark individuals. The Algorithm 4 shows the flowchart of Gaussian mutation. We call the generated sparks by this way, Gaussian sparks.

Algorithm 4 Gauss Mutation of FWA. $Gaussian(1,1)$ denotes a Gaussian distribution with mean 1 and standard deviation 1.

```
1: Initialization of fireworks $x_i$ randomly.
2: $z = round(d * rand(0,1))$.
3: Select $z$ dimensions randomly of $x_i$.
4: $g = Gaussian(1,1)$.
5: for Selected dimensions of $x_i^j$ do
6:     $x_i^j = x_i^j \times g$.
7:     if $x_i$ outside the feasible region then
8:         Use mapping rule back to feasible region.
9:     end if
10: end for
```

- **Selection:** we use a elitist strategy and immune density based selection method to obtain the next generation fireworks from all spark individuals and current firework individuals. The best firework or spark go to next generation directly and remaining firework individuals in the next generation are selected based on their selection probability that can be calculated using the Eqs. (2.26) and (2.27).

$$R(x_i) = \sum_{j \in K} ||x_i - x_j|| \tag{2.26}$$

$$p(x_i) = \frac{R(x_i)}{\sum_{j \in K} R(x_j)} \tag{2.27}$$

where K is the set of current firework individuals and spark individuals.

Unlike most EC algorithms, FWA simulates explosion of real fireworks to perform different levels of searching for local areas, which shows strong optimization capabilities. Thanks to its fascinating features, many researchers focus on further improving its performance by proposing various novel mechanisms, such as, the enhanced FWA (EFWA) [208], dynamic FWA (dynFWA) [207] and others [87]. Meanwhile, FWA is also applied to various scenarios including multimodal optimization [86], dynamic optimization [117], multi-objective optimization [200]. Beside, FWA has been applied to many real-world problems and achieved encouraging successes. For example, regional seismic waveform inversion [42], constrained portfolio optimization [11], web information retrieval [21], multilevel image thresholding [166], RFID network planning [167] and privacy preserving [131], etc. In a word, although FWA is a new family member of the EC community, there is still plenty of room to further improve FWA performance.

2.2 Related Research

2.2.1 Approximation Models

Although EC has attracted enough attention and achieved great success both in academia and industry, it is still difficult to solve optimization problems perfectly with limited computing resources because of their heavy computational cost and high-dimensions. Besides, the number of subjective evaluations of IEC users is also limited because of their human fatigue [155]. Thus, an approximate model as a potential approach has achieved many gratifying results in recent decades. It aims to build a cheaper approximate model to replace the original optimization problems with high cost fully and partially to achieve greater benefits at lower cost.

An approximation means that one thing is similar to another but not exactly the same. Generally, approximate models are applied in EC community mainly from three levels, i.e., problem approximation, functional approximation and special approximation techniques for EC.

- **Problem approximation:** It replaces the original complex problem with a much simpler and easier model to solve the problem roughly but easily. The used model does not change the characteristics of the original problem [23, 8]. It means that solving simple problems with lower costs is equivalent to solving a complex one roughly. Although it sounds fascinating and effective, it requires a sufficient prior knowledge of the original optimization problems. How to simplify complex problems accurately is a key, and the accuracy of the established models also directly affects the performance because low precision or incorrect models can hinder convergence.
- **Functional approximation:** It usually uses sample data to construct an alternate and explicit expression, and there are many effective techniques to build a functional approximation model [71]. For example, a polynomial model is a form of regression analysis where the relationship between inputs, outputs and noise is modelled as the n-th degree polynomial. Other commonly used technologies are kriging model, neural networks, support vector machines, comparative remarks and others. Honestly, it is not so easy to establish a high quality function model because it involves many factors, such as sample data, selection of training method and selection of error measures. There are still many researchers dedicated to building accurate approximation models to accelerate convergence.
- **Evolutionary approximation:** It is specific for EC algorithms. There are two well-known strategies, fitness inheritance [148] and fitness imitation [80], to reduce the number of fitness evaluations. In fitness inheritance, the fitness of an offspring individual can be estimated from its parent. While individuals are clustered into several groups in fitness imitation, an individual representing the entire group will be evaluated and other individuals in the same cluster will be estimated from the representative individual.

Regardless the methods used to obtain an approximate model, the ultimate objective is to use the model reasonably and accelerate EC/IEC. Many practitioners try to use approximate models in almost every element of EC algorithms to maximize benefits, such as, initialization [134], crossover [9], and mutation [1]. There is no doubt that the most used way is to use approximate models to evaluate individuals to reduce the number of real fitness calculations [73, 101, 121, 64]. Model management and update is also a hot topic, and there are three kinds of management methods from the viewpoint of evolution control: no evolution control [149], fixed evolution control [55] and adaptive evolution control [72]. Some researchers focus on developing different data sampling techniques to improve the accuracy of approximate models, such as, bagging and boosting [48], active data selection [122] and others [135, 58].

In short, approximate models show great potential in EC community, but they have not yet attracted sufficient attention, and there are still many open topics for discussion because it is still unclear in which way EC algorithms can benefit from an approximate model.

2.2.2 Approximation Using Fourier Transform

Joseph Fourier first proposed the basic idea of Fourier Transform that can use a set of combination of sinusoidal function to express any continuous periodical functions. In general, Fourier transform is a linear integral transform used to transform a signal between a time domain (or spatial domain) and a frequency domain. Nowadays, discrete Fourier transform (DFT) has been applied into physics and engineering everywhere.

Fourier analysis is not only a powerful mathematical tool but also an interesting mode of thinking. Pei got inspiration and used Fourier Transform to predict a global optimum then accelerate EC search [114]. It is a unique approach to filter its frequency components for approximating a fitness landscape. Firstly, it uniformly samples the entire fitness landscape, and the resampled points as well as the EC individuals are used to obtain the frequency information for the fitness landscape by using the discrete Fourier transform. We filter to isolate the major frequency component; thus we can obtain a trigonometric function approximating the original fitness landscape after the inverse DFT is applied. Subsequently, the elite is obtained from the approximated function and the EC search process is accelerated by replacing the worst EC individual with the elite.

The experimental results confirmed that introducing Fourier transform to approximate fitness landscape is effective and promising [114]. It can be further developed into a new niching method. We can use the DFT repeatedly to extract frequency of the heighest power, the second one and others until all the optimal areas are detected. Then, these obtained elite individuals can be used to accelerate EC, even the optimization problem is a complex multimodal optimization problem.

2.2.3 Hybrid Algorithms

Hundreds of EC algorithms have been proposed, and they have their own optimization characteristics. It is difficult to say that an EC algorithm is suitable for all types of problems as the no-free-lunch the theorem [178] claims. Thus, some researchers try to mix different EC algorithms together to obtain a greater robustness to solve the same problem more efficiently than a single EC algorithm. Design idea of hybrid algorithms is simple and expects to yield better performance by combining two or more other EC algorithms. Generally, any EC algorithm can be mixed with other EC algorithms or just introduce a part of operations, and hybrid algorithms usually have features of multiple mixed EC algorithms. However, improper hybrid may weaken the performance of algorithms or even hinder the convergence of EC algorithms. How to improve the performance of hybrid algorithms is one of core topics in this field.

There are so many EC algorithms that we cannot list all existing hybrid algorithms. We choose DE as an example and list several well-known hybrid algorithms of DE and other EC algorithms. A high-cited paper is a hybrid of DE and PSO to solve constrained numerical and engineering optimization [94]. DE is also combined with others, such as, harmony search algorithm [30], genetic algorithm [165], artificial bee colony algorithm [67], bacterial foraging optimization [17]. Many DE-based hybrid algorithms are used to solve engineering problems widely, such as, flow shop scheduling problems [173, 126], digital filter design [141, 172], economic dispatch [44, 168], and others [111, 125]. We searched Scopus database with the keywords of *differential evolution* and *hybrid algorithms* and retrieved 438 related papers are retrieved until writing this dissertation (March 2019), which also shows the prosperity of this field.

Many researchers also have worked on combining EC algorithms with other technologies. For example, practitioners introduce neural networks into EC algorithms to gain learning capabilities [14, 109] and use them to solve many real problems, e.g. real time clustering [96], fire recognition [74]. Fuzzy logic system as an important branch of computational intelligence is also introduced into EC algorithms to enhance their performance [28, 4]. Msot of these hybrid algorithms are used in multi-objective reactive power and voltage control [204], fuzzy modeling [120], university course timetable problem [32] and others. In short, hybridizing algorithms is an effective approach and a hot topic in EC community, and there is still a lot of room for further research.

2.2.4 Improvements of EC Search

When a new EC algorithm is proposed, a large number of improved versions are gradually proposed to improve its performance by integrating new search strategies and mechanisms. Although each EC algorithm has a different evolutionary mechanism to generate offspring individuals, we briefly summarize several common means of improvement on EC search as following.

- **Initialization:** It is the beginning stage of an EC algorithm, and usually random initialization is adopted to generate initial population. It is an important factor in determining the performance of an algorithm, thus many researchers have proposed various methods to initialize the population. A review of population initialization techniques listed several common methods [76], and this part has not caused enough attention compared to other improvement approaches.

- **Evolutionary mechanisms:** Most researchers are committed to proposing universal and effective strategies to replace or assist the original strategies to improve the performance of EC algorithms. Generally, they may change the optimization framework slightly, but not change the core idea of their optimization. Each EC algorithm has multiple variants, in the case of DE, as an example, two survey papers [37, 36] give detailed development history of DE and its a large number of improved versions. Besides, some researchers propose proprietary strategies based on the characteristics of EC algorithms. Honestly, this is a very effective way to improve performance of EC algorithms but is difficult to make breakthrough work.
- **Selection:** A appropriate selection strategy which can keep a good balance between convergence and diversity is conducive to the convergence of EC algorithms. Many famous seclection strategies mentioned in the previous section have achieved gratifying results, and some researchers proposed adaptive strategy to switch different selection strategies according to collected information in evolution [25, 27]. Although the research on this part is not popular, it has the value of further research.
- **Parameter setting:** It is an enduring topic in EC community, and there are no general rules to guide setting parameters. We try to run EC algorithms many times and set parameters according to our experiences, but it is difficult to ensure that EC algorithms always have a strong optimization ability. Thus, many methods of adaptively parameter tuning were proposed according to real-time optimization process, such as, self-adaptive [118], fuzzy adaptive [95] and others [147, 46]. Anyway, optimizing parameter setting is an important research topic.

2.3 Chapter Summary

In this chapter, we first introduced basic concepts of EC and gave a short mathematical derivation of an estimated convergence point. Then, we introduced the optimization framework of three EC algorithms used in our dissertation. Finally, we picked up several research related to our topics and briefly summarized their developments, contributions and limitations.

Chapter 3

Accuracy Improvements for an Estimated Convergence Point

3.1 Estimation Method with an Individual Pool

3.1.1 An Individual Pool

In the original estimation method descripbed in the Section 2.1.2, we just use two adjacent generations to estimate a convergence point and ignore the fact that the previous generations can also offer useful information. In fact, as the population gradually converges to the optimal area, the diversity of the population is also gradually lost and the similarity between individuals increases. This tendency is not conducive to the accurate estimation of a convergence point of the population, so we use historical individuals to alleviate the loss of diversity and improve the accuracy of an estimated convergence point. To achieve this, we introduce an individual pool storage mechanism [193] to preserve relatively outstanding individuals from previous generations. We furthermore develop an extension to this method which increases the communication between this individual pool and the population.

The Individual Pool Mechanism

The individual pool introduces a separate storage mechanism, different from the population, to save the better individuals from previous generations. Suppose the size of the individual pool is N; past individuals are copied to the individual pool in turn until the maximum limit N is reached. Then, better individuals from each subsequent generation are selected and replace the worse individuals in the pool. There are many ways to implement this replacement strategy. In this paper, we adopt a greedy replacement strategy, which means that selected individuals will replace the worst ones in the individual pool in turn if they are better than the worst ones. Finally, individuals coming from the individual pool - rather than those from the current generation - are used to estimate a convergence point of the population. Fig. 3.1 demonstrates how to use the pool to estimate a convergence point of the population. This framework for the proposed individual pool can be summarised as below in Algorithm 13.

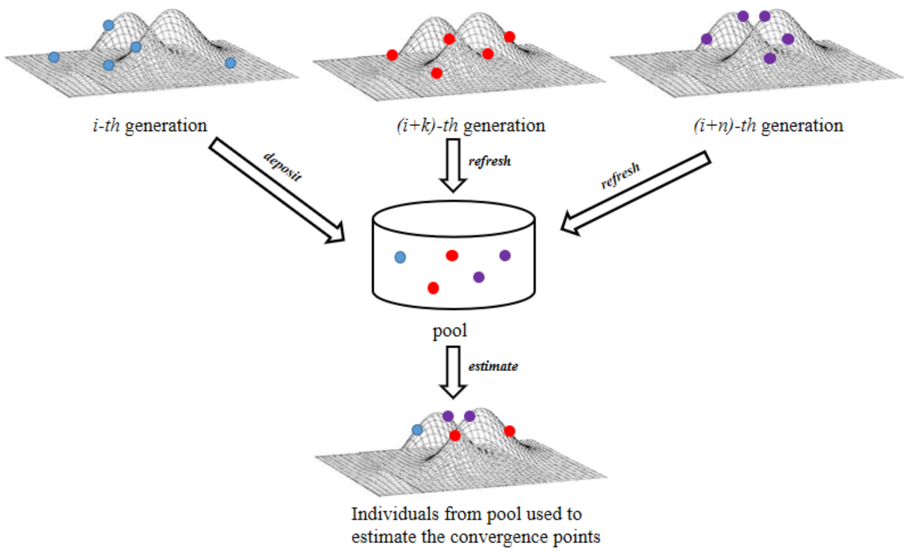


Figure 3.1: Individuals from a pool are used to estimate the convergence points.

Extended Version

The above outlined proposal emphasizes the use of past information to estimate a convergence point of the population but does not re-use the outstanding individuals in the past to guide the search of individuals in current generation. There should be many methods to further use historical information and enhance performance.

In this part, we develop an extended version to obtain a better acceleration effect by strengthening the interaction between the individual pool and the population. There are two new extra modifications based on the basic proposed pool version. The first approach is to make full use of the individuals generated from the first improvement proposed in [190]. In our previous research, the generated offspring around the parent is just used to construct the moving vector, and is then discarded. So, in the first approach, we replace the parent with its offspring if the offspring is better than its parent. Nevertheless, the population and the individual pool is independent, and there remains no interaction between them. The second approach therefore focuses on improving the interaction of the two storage mechanisms. This time, we first rank the population and remove the $a\% * M$ ranked lower individuals, where M is population size. Then, randomly select $a\% * M$ individuals from the pool, and copy them to population to keep the population size unchanged. In the following experiments, a is set to 10.

3.1.2 Experimental Evaluations

Fourteen benchmark functions from the CEC2005 test suite [153] are used in our evaluations. Their landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal. We test them with three dimensional settings, D =2-D, 10-D, and 30-D.

To test the effectiveness of the individual pool, we use exactly the same experimental settings as [188], where we investigated various combinations of proposed

Algorithm 5 The general framework for using a pool to estimate the convergence points for accelerating EC.

```
1: Generate an initial population.
2: for $G = 1$ to $MaxGeneration$ do
3:     if the maximum limit $N$ is not reached then
4:         copy individual to the pool in turn.
5:     end if
6:     if the maximum limit $N$ is reached then
7:         Obtain the estimated convergence point(s) using individuals from the pool.
8:         Replace worse individuals in the pool with the better ones from the following
           generations.
9:     end if
10:    Generate the offspring using EC.
11:    Replace the worse individuals in the population with the estimated point(s).
12: end for
13: return the optimum
```

improvements in [190]. The experimental results showed that the combination of *Improvements 1, 2, and 4* with DE can obtain the best performance in most cases. In this evaluation, we use the best combination as the baseline algorithm. Two experiments were designed where the individual pool is combined with the baseline algorithm. In the first experiment, we combine the baseline algorithm with basic individual pool, while in the second experiment, we combine the baseline algorithm with extended version. The DE experimental parameters are set as in Table 4.9. Additionally, the size of the pool is set to the same size as the that of population in each of the searching dimensions, 2-D, 10-D, and 30-D.

Table 3.1: DE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	20, 50, and 100
scale factor F	1
crossover rate	0.9
DE operations	DE/rand/1/bin
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	800, 10,000, 60,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

For fair evaluations, we evaluate convergence along the number of fitness calls instead of generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces and apply the Wilcoxon signed-rank test on the fitness values at the maximal number of fitness calculations, MAX_{NFC}, to check the significance of the proposal and baseline algorithm. Tables 3.2 show the statistical test result of experiment 1 and experiment 2. The Fig. 3.2 shows the average convergence curves of three methods with 30 trial runs on 30D.

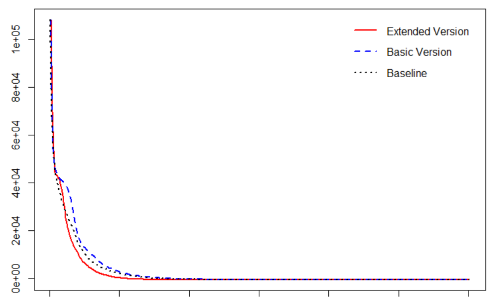
Table 3.2: Wilcoxon signed-rank test results on the fitness values at the maximal number of fitness calculations of two experiments for 30 trial runs for 14 functions. $\gg$ and $>$ mean that the left is better than the right with significance levels of 1%, 5% respectively, and $\approx$ means there is no significance. B represents the basic algorithm described in the previous section, P1 and P2 are basic version and extended version used in experiment 1 and experiment 2, respectively.

Func.	Experiment 1			Experiment 2		
	2-D	10-D	30-D	2-D	10-D	30-D
f_1	P1 $\gg$ B	B $\approx$ P1	B $\gg$ P1	P2 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B
f_2	P1 $\gg$ B	P1 $\approx$ B	B $\approx$ P1	P2 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B
f_3	P1 $\approx$ B	B $\approx$ P1	B $\approx$ P1	P2 $>$ B	B $\approx$ P2	P2 $\gg$ B
f_4	P1 $\gg$ B	P1 $\approx$ B	B $\approx$ P1	P2 $\gg$ B	B $>$ P2	B $\gg$ P2
f_5	P1 $>$ B	P1 $\gg$ B	P1 $\approx$ B	P2 $\gg$ B	B $\approx$ P2	B $\approx$ P2
f_6	B $\approx$ P1	B $\approx$ P1	B $\approx$ P1	B $\approx$ P2	B $\approx$ P2	P2 $\approx$ B
f_7	P1 $\approx$ B	P1 $\gg$ B	P1 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B
f_8	B $\approx$ P1	P1 $\gg$ B	P1 $\gg$ B	P2 $\approx$ B	P2 $\gg$ B	P2 $\gg$ B
f_9	B $\approx$ P1	P1 $\gg$ B	P1 $\gg$ B	B $\approx$ P2	P2 $\gg$ B	P2 $\gg$ B
f_{10}	P1 $\approx$ B	P1 $\gg$ B	P1 $\approx$ B	P2 $\approx$ B	P2 $\gg$ B	B $\approx$ P2
f_{11}	P1 $\gg$ B	P1 $\gg$ B	P1 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B	P2 $\gg$ B
f_{12}	P1 $>$ B	B $\approx$ P1	B $\approx$ P1	P2 $\gg$ B	B $\approx$ P2	B $\approx$ P2
f_{13}	B $\approx$ P1	P1 $\gg$ B	P1 $\gg$ B	B $\approx$ P2	P2 $\gg$ B	P2 $\gg$ B
f_{14}	B $\gg$ P1	B $\approx$ P1	P1 $\gg$ B	B $>$ P2	P2 $\gg$ B	P2 $\gg$ B

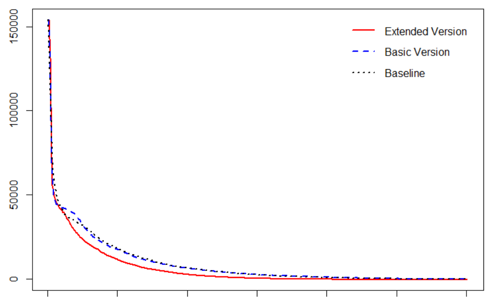
3.1.3 Discussion

We begin our discussion with an explanation of the superiority of our proposed strategies. In our previous research, only the current generation was used to estimate the convergence points; meanwhile the distribution of population became more and more concentrated with the evolution of population. It may be detrimental to estimate the convergence points from a population which is thus gradually converging. However, an individual pool can overcome the above limitation and preserve outstanding individuals from previous generations to ensure the diversity and breadth of the distribution. Although the proposed individual pool can flexibly use historical information, its size needs to be further studied; too large or too small and it is not helpful for estimating. In the next step, we will focus on dynamically adjusting the size according to the optimization problem.

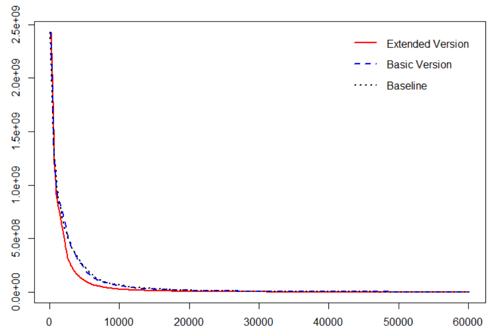
The extended version emphasizes the more efficient use of information and interaction between the population and the individual pool. In our previous studies, the improvement 1 proposed in [190] can be further utilized. In that improvement, the offspring will replace its parent if it is better. Here we have also introduced an interactive mechanism to strengthen the communication between the population and the pool. Using better individuals from the pool to replace worse ones in the population can accelerate the convergence of the population, which in turn will improve the quality of the pool. This virtuous circle, thus formed, can obtain better accelerated convergence.



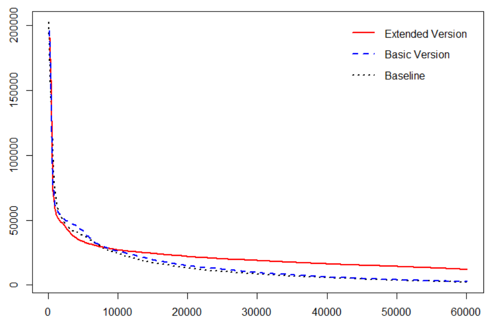
(a) f_1



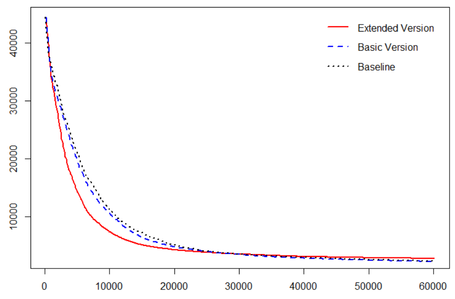
(b) f_2



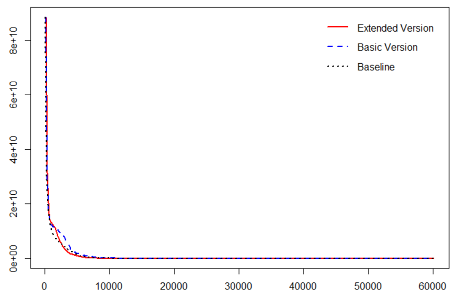
(c) f_3



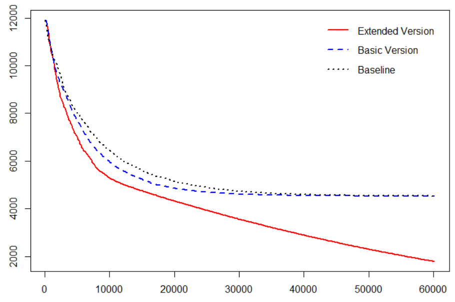
(d) f_4



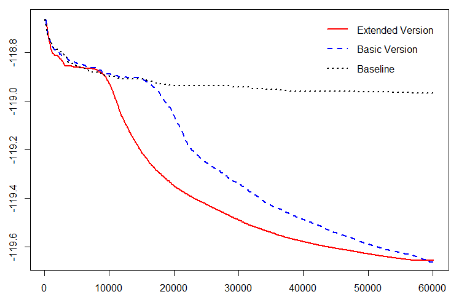
(e) f_5



(f) f_6

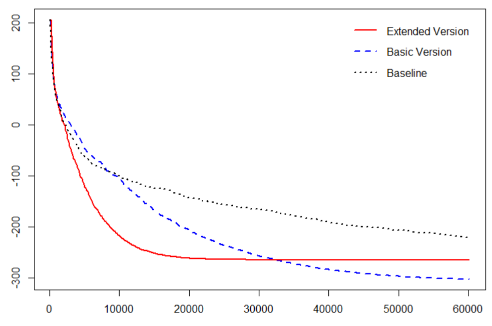


(g) f_7

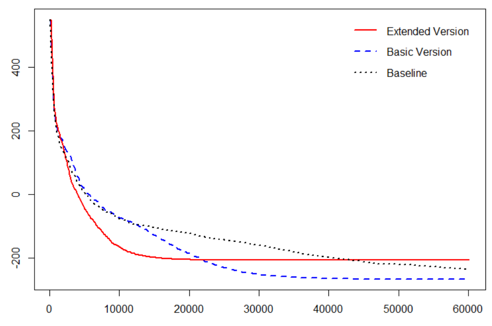


(h) f_8

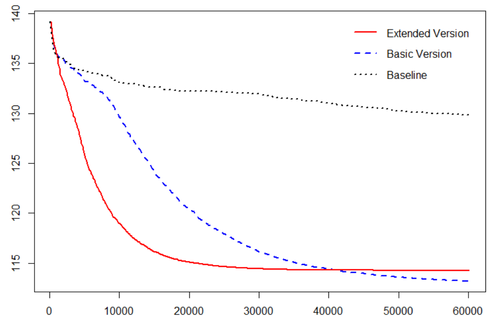
To analyze the performance of the two proposed versions, the Wilcoxon signed-



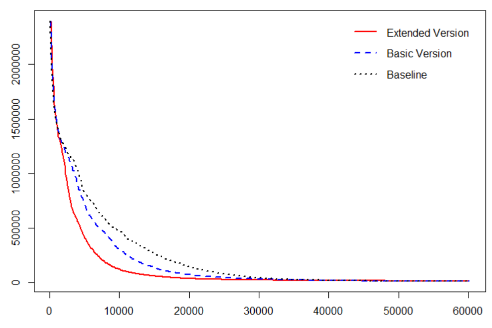
(a) f_9



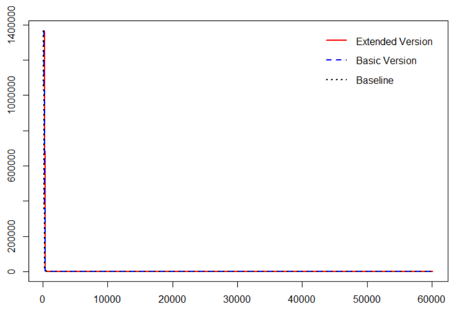
(b) f_{10}



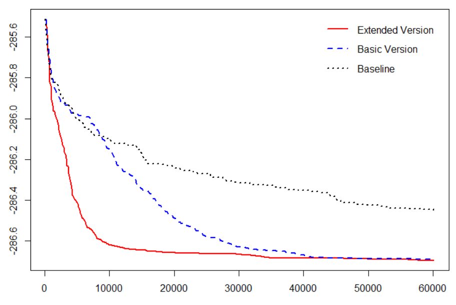
(c) f_{11}



(d) f_{12}



(e) f_{13}



(f) f_{14}

Figure 3.2: Convergence curves of 30-D f_1–f_{14}.

rank test were applied at the stop condition in three different dimensions. The statistical results reproduced in Table 3.2 show that either of the two proposed version can improve the performance of the original baseline algorithm, and the extended version can further improve performance in almost all evaluation cases.

From the results of statistical tests, we can see that the basic individual pool can accelerate multimodal optimization, while it does not achieve much acceleration effect in unimodal optimization. The extended version can achieve favorable results in both cases. This could be caused by the interaction between the population and the pool, while detailed reasons need to be depth studied in our future work. Otherwise, f_2 and f_4 are exactly the same benchmark function, with f_4 just adding noise interference based on f_2. The experimental results reveal that the noise may reduce the accuracy of the estimation such that no acceleration effect can be produced. It may be a big challenge to remove the noise and smooth the landscape to obtain better performance.

3.2 Weight-based Estimation Method

3.2.1 Estimation with Weighted Moving Vectors

Basic estimation method treats moving vectors equally to calculate a convergence point. However, their contribution to estimating a convergence point is different because of various factors; for example, some moving vectors go towards the optimal area directly, while others do not; some are closer to the optimal area, while others are far from there. Each moving vector must have a different reliability in estimating a convergence point according to its reliability. It means that higher importance should be placed on moving vectors having higher reliability, such as having a better direction to the global optimum or near the global optimum. Here, we introduce weights into the basic estimation method of a convergence point and propose two methods for calculating weights for moving vectors [187].

Suppose a proper weight w_i is given to the i-th moving vector, $\boldsymbol{b}_i$. All notations used in the following calculation are the same with those defined in previous section. Next, we show our stepwise calculation of the weight-based estimation method.

We want to find $\hat{\boldsymbol{x}}$ making the total distance, $\mathrm{J_w}(\boldsymbol{x}, \{t_i\})$ in the Eq.(3.1), minimal. Note that the weight is acting on the distance between the point $\hat{\boldsymbol{x}}$ and moving vectors.

$$\mathrm{J_w}(\boldsymbol{x}, \{t_i\}) = \sum_{i=1}^{n} w_i \|\boldsymbol{a}_i + t_i \boldsymbol{b}_i - \boldsymbol{x}\|^2 \tag{3.1}$$

From the Eq. (2.2), Eq. (3.2) is obtained.

$$\mathrm{t_i} = \frac{\boldsymbol{b}_i^T(\boldsymbol{x} - \boldsymbol{a}_i)}{\|\boldsymbol{b}_i\|^2} \tag{3.2}$$

Let put Eq. (3.2) into Eq. (3.1) to delete t_i and obtain the Eq. (3.3).

$$\begin{aligned} \mathrm{J_w}(\boldsymbol{x}) &= \sum_{i=1}^{n} w_i \left\| \boldsymbol{a}_i + \frac{\boldsymbol{b}_i^T(\boldsymbol{x} - \boldsymbol{a}_i)}{\|\boldsymbol{b}_i\|^2} \boldsymbol{b}_i - \boldsymbol{x} \right\|^2 \\ &= \sum_{i=1}^{n} w_i \left\{ (\boldsymbol{x} - \boldsymbol{a}_i)^T (\boldsymbol{I}_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^T)^T (\boldsymbol{I}_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^T)(\boldsymbol{x} - \boldsymbol{a}_i) \right\} \end{aligned} \tag{3.3}$$

Next, $\hat{\boldsymbol{x}}_w$, is obtained by partially differentiating each element of $\boldsymbol{x}$ and setting them equal 0. Finally, we can obtain the weight-based estimated convergence point $\hat{\boldsymbol{x}}_w$ using the Eq. (3.4).

$$\hat{\boldsymbol{x}}_w = \left\{ \sum_{i=1}^{n} w_i \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^{\mathrm{T}} \right) \right\}^{-1} \left\{ \sum_{i=1}^{n} w_i \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \tag{3.4}$$

3.2.2 Two Methods for Calculating Weights

Weights have been imported into the basic estimation method successfully. The next key problem is how to calculate weights values. Here, we propose two different perspectives to determine weights. Of course, there must be other ways of calculating the weights.

Fitness Gradient of Moving Vectors

Moving vectors themselves contain a lot of information, e.g. evolutionary direction, length of moving vectors, fitness changes, and others. As the first attempt, we use a fitness gradient between the starting point and the terminal point of a moving vector to evaluate the reliability of the moving vector. Bigger fitness gradient of a moving vector means that the direction has a higher probability of approaching the global optimal area. Thus, higher weights should be given to favorable moving vectors. Conversely, lower weights should be given to moving vectors with less fitness gradients. Suppose the fitness change of the i-th moving vector, Δ_i, can be calculated as $\Delta_i = f(\boldsymbol{c}_i) - f(\boldsymbol{a}_i)$. Then, the fitness gradient information of the i-th individual can be calculated as $G_i = \frac{\Delta_i}{||\boldsymbol{c}_i - \boldsymbol{a}_i||}$, and the Eq. (3.5) can be used to give a weight to each moving vector.

$$w_i = \frac{G_i}{\sum_{i=1}^{n} G_i} \quad , \tag{3.5}$$

where n represents the total number of moving vectors.

Fitness of Parents

Generally, individuals with higher fitness may be closer to the global optimal area, while poorer individuals stay away from the area and need more iterations to converge. Poorer individuals are interfered more. For example, the directions of moving vectors in valleys among local minima are influenced by multiple hills and valleys in a fitness landscape and do not always go toward the same local minimum. It is easy to imagine that estimation errors of a convergence point become bigger. Based on this assumption, we roughly think that the closer individuals are to the optimal area, the higher given reliability, i.e. higher weight, should be. Thus, we use the Eq. (3.6) to give a different weight to each moving vector for minimum optimization tasks.

$$w_i = \frac{MaxF - f(\boldsymbol{a}_i)}{\sum_{i=1}^{n} MaxF - f(\boldsymbol{a}_i)} \tag{3.6}$$

where $MaxF$ is the worst fitness in the current generation, and $f()$ returns the fitness of an incoming individual.

We introduce weights successfully and explain how to determine weights in detail. The next work is to combine our proposed weight-based estimation method with EC algorithms to accelerate their convergence. Here, the Algorithm 6 shows generic flowchart of the EC algorithms combined with our proposal.

Algorithm 6 Weight-based estimated convergence point to accelerate EC. G: generation.

```
1: Generate an initial population.
2: Evaluate the fitness of each individual.
3: for $G = 1$ to $MaxGeneration$ do
4:     Obtain the next generation using an EC algorithm.
5:     Calculate moving vectors
6:     Calculate weights for all moving vectors.
7:     Obtain the estimated point using weighted moving vectors.
8:     Evaluate the fitness of the estimated convergence point.
9:     if its fitness is better than that of the worst individual in current population then
10:        use the estimated point as an elite individual, and replace the worst individual.
11:    end if
12: end for
13: return the optimum
```

3.2.3 Experimental Evaluations

Here, we define shorten names of three methods for comparison:

- basic method: original estimation method of a convergence point using moving vectors without weighting.
- method 1: proposed method for estimating a convergence point using moving vectors weighted by fitness gradient.
- method 2: proposed method for estimating a convergence point using moving vectors weighted by fitness of parents.

We combine each of three methods with DE and compare them:

Experiment 1: Visual Evaluation of Estimated Convergence Points

The first experiment is to visually show the effect of method 1 and 2 by employing a 2-dimensional Gaussian function as a test function. Detailed experimental settings are as follows.

The used Gaussian function is represented by the Eq. (3.7). Its experimental parameters are shown in the Table 3.3.

$$f(x) = -\left\{ a \exp\left(-\sum_{i=1}^{2} \frac{(x_i - \mu_i)^2}{2\sigma_i^2} \right) \right\} \tag{3.7}$$

The objective of this Experiment 1 is to evaluate the accuracy of our proposal using exactly the same moving vectors. To do it, we generate a searching point randomly around each individual within a tiny area, compare fitness of the individual and the generated point, and make the poorer and better ones be the starting point and the terminal point of a moving vector, respectively. Then, all moving vectors

Table 3.3: Parameters of Eq. (3.7).

dimensions	2-D
population size	20
search ranges	[-6,6] of all 2 variables
a	3.1
$\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$	(2.0,2.0)
$\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$	(0.0,0.0)

aim the direction of climbing down to the local optimum or the global optimum. This method generating moving vectors without using calculated individuals but generating them in tiny areas increases the number of fitness calculations. However, it does not influence by local hills thanks to moving vectors in tiny areas and is better for exact evaluation for the Experiment 1.

These evaluations run 30 times over 10 generations. The Fig. 3.3 shows the average convergence curve of fitness error between the estimated convergence point and the global optimum. The Fig. 3.4 shows convergence points estimated by three methods and exactly the same moving vectors used for three methods.

We apply the Friedman test and the Holm multiple comparison test at each generation to check significant difference among three methods. The results are showed in the Table 3.4.

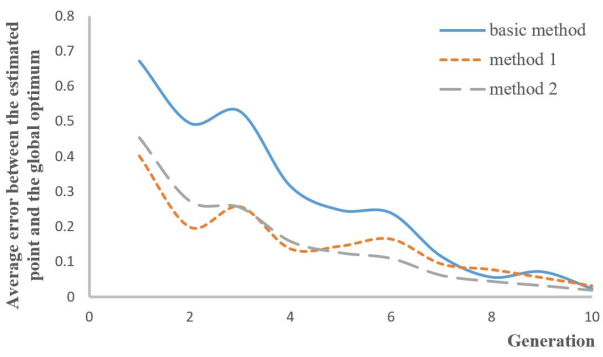


Figure 3.3: The average convergence curve of fitness error between the estimated point and the global optimum. See the definition of three method names at the top of this section.

Experiment 2: Evaluation of EC Acceleration with an Estimated Convergence Point

The second experiment is designed to analyze the acceleration effect of proposed weighting methods for EC algorithms, where an estimated convergence point is used as an elite individual and replace the worst individual when its fitness is better than the worst one. We use 28 benchmark functions from the CEC2013 test suite [91] in this evaluation experiment.

We select DE and PSO as our test baseline algorithms and combine them with our proposals with the parameter setting as described in the Tables 4.9 and 4.2.

Table 3.4: Statistical test results of the Friedman test and the Holm multiple comparison test for average fitness values of 30 trial runs of 3 methods. $A < B$ mean that B is significantly better than A with significance level 5%, and $A \approx B$ means that there is no significant difference between A and B.

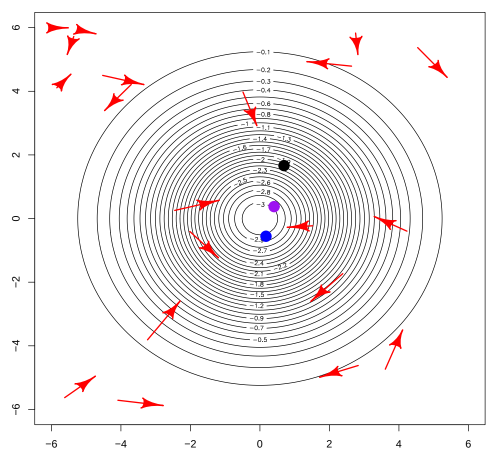
Generation	basic method vs. method 1 and 2
1	basic method $\approx$ method 2 $\approx$ method 1
2	basic method $<$ method 2 $<$ method 1
3	basic method $<$ method 1 $\approx$ method 2
4	basic method $\approx$ method 2 $\approx$ method 1
5	basic method $<$ method 1 $\approx$ method 2
6	basic method $\approx$ method 1 $\approx$ method 2
7	basic method $\approx$ method 1 $\approx$ method 2
8	method 1 $\approx$ basic method $\approx$ method 2
9	basic method $\approx$ method 1 $<$ method 2
10	method 1 $\approx$ basic method $\approx$ method 2

Table 3.5: DE algorithm parameter settings.

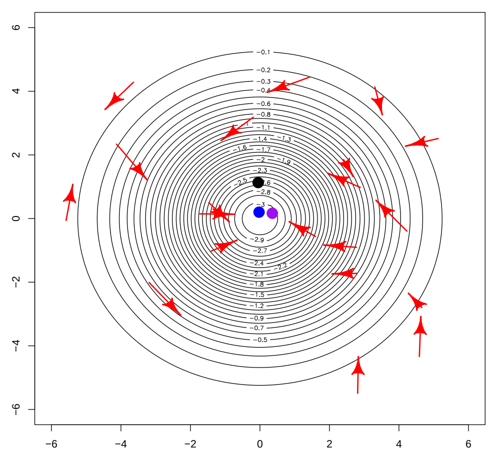
population size for 2-D, 10-D, and 30-D search	80
scale factor F	0.9
crossover rate	0.9
DE operations	DE/rand/1/bin
# of trials	51
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000 × D

Unlike the Experiment 1, moving vectors are made using existing individuals not to increase fitness calculation cost. Since each target vector in DE generates a trial vector, we set the better one as the terminal point of a moving vector and the poorer one as the starting point of a moving vector; a similar approach is used for PSO. It ensures that each individual can make a moving vector. These moving vectors are weighted and used to estimate a convergence point.

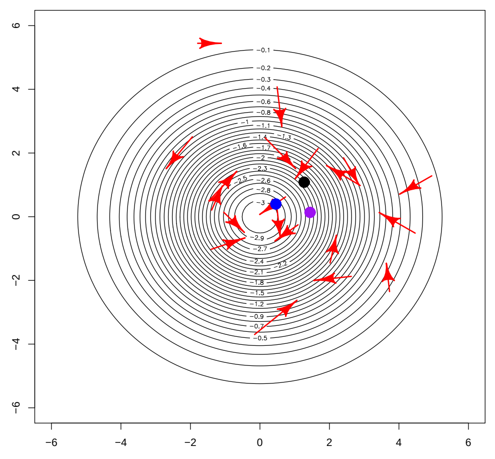
For fair evaluations, we evaluate convergence against the number of fitness calls rather than generations. We test each benchmark function with 51 trial runs in 4 different dimensional spaces. We apply the Friedman test and the Holm multiple comparison test on the fitness values at the stop condition, i.e. the maximum number of fitness calculations, to check whether there is a significant difference among all algorithms. The Tables 3.7 and 3.8 show their result of the statistical tests.



(a) First generation at first trial run



(b) Third generation at first trial run



(c) Fifth generation at first trial run

Figure 3.4: black dot, purple dot and blue dot represents the estimated convergence point that without use any weight method, use method 1 for handling weight and use method 2 for handling weight, respectively.

3.2.4 Discussion

Analysis of Proposed Estimation Method Using Weighted Moving Vectors

The first discussion is the superiority of our proposed methods for estimating a convergence point. Basic estimation method gives the same weight to moving vectors. Actually, the reliability of moving vectors is different because of various factors, e.g. evolutionary direction error, and their contribution to the estimated convergence point is also different. Thus, we use weights to enhance or weaken the influence of moving vectors to improve the accuracy of the estimated convergence point; higher weights are given to more reliable moving vectors. Note that the proposed method adjusts weights adaptively based on the current searching situation without intro-

Table 3.6: PSO algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	80
inertia factor w	1
constant c_1 and c_2	1.4962 and 1.4962
max. and min. speed V_{max} and V_{min}	1 and -1
# of trials	51
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000 × D

Table 3.7: Statistical test result of the Friedman test and the Holm multiple comparison test for average fitness values of 51 trial runs of 4 methods. $A > B$ mean that A is significantly better than B with significant a level of 5%. $A \approx B$ means that there is no significant difference between A and B. DE0, DE1, and DE2 mean (DE + basic method), (DE + method 1), and (DE + method 2), respectively. DE means conventional DE without any estimated convergence point.

	2D	10D	30D	50D
F_1	$DE2 \approx DE1 > DE \approx DE0$	$DE1 > DE2 > DE0 \approx DE$	$DE1 > DE2 > DE0 > DE$	$DE1 > DE2 > DE0 > DE$
F_2	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE2 > DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 > DE2 > DE0 \approx DE$
F_3	$DE2 \approx DE1 \approx DE0 \approx DE$	$DE2 \approx DE1 > DE \approx DE0$	$DE2 \approx DE1 > DE \approx DE0$	$DE1 \approx DE2 > DE \approx DE0$
F_4	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE2 \approx DE1 > DE \approx DE0$	$DE1 \approx DE2 > DE0 > DE$	$DE1 \approx DE2 > DE \approx DE0$
F_5	$DE2 \approx DE1 > DE \approx DE0$	$DE1 > DE2 > DE \approx DE0$	$DE1 > DE2 > DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$
F_6	$DE2 \approx DE \approx DE0 \approx DE1$	$DE2 \approx DE1 > DE0 \approx DE$	$DE1 > DE2 > DE0 \approx DE$	$DE2 \approx DE1 > DE0 > DE$
F_7	$DE2 \approx DE1 \approx DE \approx DE0$	$DE2 \approx DE1 > DE0 \approx DE$	$DE2 \approx DE1 > DE \approx DE0$	$DE2 > DE1 > DE \approx DE0$
F_8	$DE1 \approx DE2 \approx DE \approx DE0$	$DE2 \approx DE \approx DE0 \approx DE1$	$DE \approx DE0 \approx DE2 \approx DE1$	$DE1 \approx DE2 \approx DE0 \approx DE$
F_9	$DE0 \approx DE1 \approx DE2 \approx DE$	$DE \approx DE1 \approx DE0 \approx DE2$	$DE \approx DE2 \approx DE1 \approx DE0$	$DE2 \approx DE1 \approx DE0 \approx DE$
F_{10}	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 > DE2 > DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 > DE$
F_{11}	$DE1 \approx DE2 > DE0 \approx DE$	$DE2 \approx DE1 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 > DE$
F_{12}	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE2 \approx DE1 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 \approx DE$
F_{13}	$DE2 \approx DE1 > DE0 \approx DE$	$DE2 \approx DE1 > DE \approx DE0$	$DE2 \approx DE1 > DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$
F_{14}	$DE2 \approx DE \approx DE1 \approx DE0$	$DE \approx DE0 \approx DE1 \approx DE2$	$DE \approx DE0 \approx DE1 \approx DE2$	$DE1 \approx DE2 \approx DE \approx DE0$
F_{15}	$DE2 \approx DE1 \approx DE \approx DE0$	$DE2 \approx DE1 \approx DE0 \approx DE$	$DE2 \approx DE1 \approx DE \approx DE0$	$DE1 \approx DE2 \approx DE \approx DE0$
F_{16}	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE0 \approx DE1 \approx DE \approx DE2$	$DE \approx DE2 \approx DE0 \approx DE1$
F_{17}	$DE \approx DE2 \approx DE0 \approx DE1$	$DE2 \approx DE1 > DE0 > DE$	$DE1 \approx DE2 > DE0 > DE$	$DE2 \approx DE1 > DE \approx DE0$
F_{18}	$DE1 \approx DE \approx DE2 \approx DE0$	$DE2 \approx DE1 > DE0 > DE$	$DE2 \approx DE1 > DE0 > DE$	$DE1 \approx DE2 > DE0 \approx DE$
F_{19}	$DE2 \approx DE \approx DE1 \approx DE0$	$DE2 \approx DE1 > DE0 \approx DE$	$DE1 \approx DE2 > DE0 > DE$	$DE1 > DE2 > DE0 > DE$
F_{20}	$DE1 \approx DE0 \approx DE2 \approx DE$	$DE1 \approx DE2 \approx DE \approx DE0$	$DE1 \approx DE2 > DE \approx DE0$	$DE2 \approx DE1 > DE \approx DE0$
F_{21}	$DE2 \approx DE1 \approx DE0 \approx DE$	$DE2 \approx DE1 > DE0 > DE$	$DE1 > DE2 > DE0 > DE$	$DE1 > DE2 \approx DE > DE0$
F_{22}	$DE1 \approx DE0 \approx DE \approx DE2$	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE0 \approx DE \approx DE2$
F_{23}	$DE2 \approx DE1 \approx DE \approx DE0$	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE2 \approx DE0 > DE$	$DE2 \approx DE \approx DE0 \approx DE1$
F_{24}	$DE1 \approx DE \approx DE0 \approx DE2$	$DE2 > DE1 > DE \approx DE0$	$DE1 \approx DE2 > DE > DE0$	$DE1 \approx DE2 > DE \approx DE0$
F_{25}	$DE2 \approx DE \approx DE1 \approx DE0$	$DE2 \approx DE1 > DE \approx DE0$	$DE2 \approx DE1 \approx DE \approx DE0$	$DE2 \approx DE \approx DE0 \approx DE1$
F_{26}	$DE1 \approx DE2 \approx DE0 \approx DE$	$DE1 \approx DE2 > DE0 \approx DE$	$DE2 \approx DE1 > DE0 \approx DE$	$DE1 \approx DE2 > DE \approx DE0$
F_{27}	$DE1 \approx DE2 \approx DE \approx DE0$	$DE1 \approx DE2 > DE \approx DE0$	$DE1 \approx DE2 > DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$
F_{28}	$DE1 \approx DE2 \approx DE \approx DE0$	$DE1 \approx DE2 > DE0 \approx DE$	$DE1 > DE2 > DE0 > DE$	$DE1 > DE0 \approx DE > DE2$

ducing any new control parameters or increasing fitness calculation cost. Increasing the estimation precision of a convergence point may accelerate to find the optimal solution. At least, it is better than the replaced worst individual. From the cost-performance point of view, we can say that our proposed method is a low risk strategy and easy to use.

The second discussion is on calculation of weights for moving vectors which affects the performance of our proposal directly. Since incorrect weights reduce the accuracy of an estimated convergence point, how to give the right weight is a key issue. We proposed two methods in the section 3.1.1 to calculate weights for moving vectors from different perspectives in this paper. The method 1 considers the fitness

Table 3.8: Statistical test result of the Friedman test and the Holm multiple comparison test for average fitness values of 51 trial runs of 4 methods. PSO0, PSO1, and PSO2 mean (PSO + basic method), (PSO + method 1), (PSO + method 1), respectively. *PSO* means conventional PSO without any estimated convergence point. The symbols used in this Table have same mean with the Table 3.7.

	2D	10D	30D	50D
F_1	$PSO1 \approx PSO2 > PSO \approx PSO0$	$PSO1 > PSO2 > PSO0 > PSO$	$PSO1 > PSO2 > PSO0 \approx PSO$	$PSO1 > PSO2 > PSO0 \approx PSO$
F_2	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO2 \approx PSO1 > PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO1 \approx PSO0 \approx PSO2 \approx PSO$
F_3	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 > PSO0 \approx PSO$	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$
F_4	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO \approx PSO0 \approx PSO1 \approx PSO2$	$PSO > PSO1 \approx PSO0 \approx PSO2$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$
F_5	$PSO2 \approx PSO1 > PSO0 \approx PSO$	$PSO0 \approx PSO1 \approx PSO \approx PSO2$	$PSO1 > PSO0 > PSO2 > PSO$	$PSO1 > PSO2 > PSO \approx PSO0$
F_6	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO2 \approx PSO1 \approx PSO \approx PSO0$	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO2 \approx PSO1 \approx PSO \approx PSO0$
F_7	$PSO1 \approx PSO2 > PSO0 \approx PSO$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$	$PSO \approx PSO0 \approx PSO1 \approx PSO2$	$PSO1 \approx PSO0 \approx PSO2 \approx PSO$
F_8	$PSO2 \approx PSO1 > PSO0 \approx PSO$	$PSO1 \approx PSO \approx PSO0 \approx PSO2$	$PSO2 \approx PSO0 \approx PSO \approx PSO1$	$PSO2 \approx PSO0 \approx PSO1 \approx PSO$
F_9	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO \approx PSO2 \approx PSO1 \approx PSO0$	$PSO2 \approx PSO \approx PSO1 \approx PSO0$
F_{10}	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO2 \approx PSO1 > PSO0 > PSO$	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO0 > PSO$
F_{11}	$PSO2 \approx PSO1 > PSO \approx PSO0$	$PSO > PSO0 \approx PSO1 \approx PSO2$	$PSO > PSO0 \approx PSO2 \approx PSO1$	$PSO > PSO1 \approx PSO2 \approx PSO0$
F_{12}	$PSO2 \approx PSO1 > PSO \approx PSO0$	$PSO > PSO0 \approx PSO2 \approx PSO1$	$PSO > PSO0 \approx PSO1 \approx PSO2$	$PSO > PSO2 \approx PSO1 > PSO0$
F_{13}	$PSO2 \approx PSO1 \approx PSO \approx PSO0$	$PSO > PSO0 \approx PSO1 \approx PSO2$	$PSO > PSO0 \approx PSO1 \approx PSO2$	$PSO > PSO1 \approx PSO0 \approx PSO2$
F_{14}	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO \approx PSO0$	$PSO \approx PSO1 \approx PSO2 \approx PSO0$	$PSO1 \approx PSO \approx PSO0 \approx PSO2$
F_{15}	$PSO0 \approx PSO2 \approx PSO1 \approx PSO$	$PSO0 \approx PSO2 \approx PSO1 > PSO$	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 > PSO \approx PSO0$
F_{16}	$PSO0 \approx PSO2 \approx PSO \approx PSO1$	$PSO2 \approx PSO \approx PSO1 \approx PSO0$	$PSO2 \approx PSO \approx PSO1 \approx PSO0$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$
F_{17}	$PSO1 \approx PSO2 \approx PSO \approx PSO0$	$PSO > PSO0 \approx PSO2 \approx PSO1$	$PSO > PSO1 \approx PSO2 \approx PSO0$	$PSO > PSO2 \approx PSO1 > PSO0$
F_{18}	$PSO2 \approx PSO1 \approx PSO \approx PSO0$	$PSO > PSO1 \approx PSO2 \approx PSO0$	$PSO > PSO1 \approx PSO2 \approx PSO0$	$PSO > PSO1 \approx PSO2 \approx PSO0$
F_{19}	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO2 \approx PSO0 \approx PSO1 \approx PSO$	$PSO2 \approx PSO0 \approx PSO1 \approx PSO$	$PSO0 \approx PSO1 \approx PSO2 \approx PSO$
F_{20}	$PSO \approx PSO2 \approx PSO1 \approx PSO0$	$PSO > PSO1 \approx PSO2 \approx PSO0$	$PSO > PSO2 \approx PSO1 \approx PSO0$	$PSO > PSO2 \approx PSO1 > PSO0$
F_{21}	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO > PSO1 \approx PSO2 > PSO0$	$PSO1 > PSO > PSO2 > PSO0$	$PSO0 > PSO2 > PSO > PSO1$
F_{22}	$PSO0 \approx PSO2 \approx PSO1 \approx PSO$	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO \approx PSO1 \approx PSO2 \approx PSO0$	$PSO \approx PSO1 \approx PSO0 \approx PSO2$
F_{23}	$PSO2 \approx PSO1 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO1 \approx PSO2 \approx PSO \approx PSO0$	$PSO2 \approx PSO1 \approx PSO \approx PSO0$
F_{24}	$PSO1 \approx PSO2 \approx PSO0 > PSO$	$PSO0 \approx PSO \approx PSO2 \approx PSO1$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$	$PSO \approx PSO0 \approx PSO1 \approx PSO2$
F_{25}	$PSO1 \approx PSO2 \approx PSO \approx PSO0$	$PSO2 \approx PSO0 \approx PSO1 \approx PSO$	$PSO \approx PSO0 \approx PSO1 \approx PSO2$	$PSO2 \approx PSO1 \approx PSO \approx PSO0$
F_{26}	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$	$PSO \approx PSO0 \approx PSO1 \approx PSO2$
F_{27}	$PSO0 \approx PSO2 \approx PSO1 \approx PSO$	$PSO0 \approx PSO2 \approx PSO \approx PSO1$	$PSO \approx PSO0 > PSO1 \approx PSO2$	$PSO \approx PSO0 \approx PSO2 \approx PSO1$
F_{28}	$PSO1 \approx PSO2 \approx PSO0 \approx PSO$	$PSO > PSO1 \approx PSO0 \approx PSO2$	$PSO > PSO2 \approx PSO0 \approx PSO1$	$PSO > PSO1 \approx PSO2 \approx PSO0$

gradient information of a moving vector, and the method 2 uses the fitness of parent generation. Both of them increase the weights of potential moving vectors to improve the precision of an estimated convergence point.

Actually, since optimization problems have many complex features, no method is all-powerful for any problems, even for different search periods of the same problem. For example, fitness gradients on some flat places do not work well to measure the reliability of the directions of moving vectors. Besides, when individuals converge to the optimal area, they become similar and the directions of moving vectors may be even more important. Adaptive choice of the best method among multiple methods may be the best way to calculate weights.

Finally, we discuss on the use of the estimated convergence point to accelerate EC algorithms. Generally, this is an approach of a low risk and a high return. In this paper, we used it to replace the worst individual in the current population. Since only one individual is replaced, it does not change the diversity of the population drastically. Because different EC algorithms have their own characteristics, the impact of our proposal depends on EC algorithms. For DE, an estimated convergence point replaces only the worst individual in the current population and has less impact on other individuals. However, when the estimated convergence point becomes the best individual, it affects all other particles in PSO. Thus, it is accompanied by certain risks, it may hinder convergence when its accuracy is low. The experimental results also confirmed that our proposal is easier to fall into local areas in some cases when it is applied to PSO. It may be a good choice to use an estimated convergence point every several generations for PSO to reduce its impact on other individuals to avoid premature. Actually, there are many other ways to use an estimated convergence point to balance convergence and diversity. Thus, how to use the estimated convergence point reasonably and efficiently is one of our future works.

Analysis of Experimental Evaluations

The Experiment 1 is designed to visually evaluate the performance of our proposal. From the Fig. 3.4, we can see that introducing weights can increase the accuracy of an estimated convergence point obviously, which implies that appropriate weights can reduce the impact of various errors. Statistical test results show that using proposed weighting method can obtain more accurate estimated convergence point in the early stage but becomes less effective in the later stage. Anyway, using weights does not reduce the accuracy of the estimated convergence point.

There was also no significant difference between the method 1 and the method 2. Maybe individuals become similar according to the convergence, and their difference becomes less significant. We have not been able to completely analyze what fitness landscape situation and environment make our proposed methods less effective. Detail analysis is one of our future works, which may lead to a more appropriate approach to designing weights.

The Experiment 2 is designed to evaluate the acceleration effect of the convergence point estimated by our proposed method. We designed a set of control experiments to evaluate EC vs. (EC + basic method) vs. (EC + method 1) vs. (EC + method 2). In this paper, we used the estimated convergence point as an elite individual and replace the worst one to accelerate EC convergence. The experimental results confirmed that EC < (EC + basic method) < (EC + method 1 or 2); a convergence point estimated by weighted moving vectors can further improve acceleration performance. As the dimension increases, the acceleration effect of our proposal becomes more significant; an estimated convergence point plays a more important role for more complex problems.

As we mentioned at the end of the previous subsection, the effectiveness of our proposed method for PSO is different from that for DE. One of our next works is how to combine our method with the characteristics of EC algorithms to accelerate them effectively taking advantage of its applicability to any EC algorithms.

Clustering methods are needed to divide population into multiple subgroups according to each local optimum area of multimodal tasks automatically to further improve performance of our proposal. After then, we apply our proposal to each clustered subgroup, obtain the global or local optimum, and use multiple estimated convergence points to accelerate EC convergence. It will be also one of our future works.

Potential and Future Topics

Through these above analyses, we have known that our proposal has achieved satisfactory effects and has great potential. As a new optimization approach, estimating a convergence point, there are still many aspects of improvements. Here, we list a few open topics.

How to construct moving vectors

It is crucial to determine the acceleration performance of the estimated point. In our paper, we use parent-child relationships to make moving vectors, where one parent generates one offspring individual, and a moving vector starts from a poorer individual to the better one. However, some EC algorithms do not have a one-to-one relationship but a one-to-many or many-to-many relationship. For example, firework algorithm has a one-to-many relation, and genetic algorithm has a many-to-many relationship. Thus, how to construct moving vectors for these EC algorithm is a new topic.

We can use a distance measure to find the nearest individual regardless a parent-offspring relation to make the best moving vectors between two generations. Other methods for making moving vectors are also one of the worth topics to study.

How to set parameters reasonably

We believe that the number of moving vectors affects the performance of our proposal. Too many or too few may reduce the accuracy of the estimated point. Thus, one of the next research directions is to investigate the relationship among the number of moving vectors, population size and dimensions.

How to improve the accuracy of the estimated point

It is the main topic of this paper. As described in the above, combining with a cluster algorithm may be a feasible approach for further improvement. Besides, there must be other potential methods to achieve this objective, e.g. other methods to calculate weights and different approaches to use an estimated point. We hope that these open topics may give some inspirations to other researchers and attract them to tackle these topics.

3.3 Chapter Summary

In this Chapter, we propose two methods to further improve the precision of an estimated convergence point. The first one is to introduce an individual pool to increase the availability of historical information, and an extended version was also proposed to further improve the interaction between the pool and the population. The experimental results show that these improvements can enhance the convergence speed and accuracy.

Then, we introduced weights into moving vectors based on their reliability to improve the accuracy of an estimated convergence point. The experimental results confirmed that weighting moving vectors can enhance the accuracy of the estimated convergence point and can further accelerate convergence of EC algorithms. In our future works, we focus on improving the cost-performance of our proposal, investigating the relations between the accuracy of the estimated convergence point and a population size or dimensions. We also try to use the estimated convergence point in various ways to accelerate convergence and develop other methods for weighting moving vectors.

Chapter 4

Accelerating Evolutionary Computation using Estimated Convergence Points

4.1 Accelerating Multimodal Optimization

4.1.1 Multimodal Optimization

A multimodal optimization problem contains multiple optimal or sub-optimal solutions, as opposed to a single global optimum. It is an important branch of EC, however not enough attention has been paid to this kind of niche in the EC community, and most practitioners focus on only finding the global optimum and ignore other acceptable candidate solutions.

Most realistic optimization problems are actually multimodal, and sometimes is useful to find their many different optima or sub-optima rather than just the one optimum. Actually, some users not only want to get the global optimum but also expect to have several available sub-optima for making a final decision. Niching techniques offer an effective approach and can find multiple optima during one trial run, and there are many well-known niching methods that have been proposed in the past few decades. Although these methods have shown satisfactory performance, they also introduce new parameters to track multiple optima. This increases the complexity and computational cost, and there are still some limitations that need to be addressed, e.g. low convergence, high computational cost and parameter tuning.

One of the objectives of this dissertation is to extend the basic estimation method so as to accelerate its convergence for multimodal problems and to develop a universal framework for estimating multiple global or local optima simultaneously by implementing a separation method for automatically dividing individuals according to each local optimum area.

4.1.2 Separation Method for Multimodal Tasks

The basic estimation method is unquestionably effective for unimodal tasks, but the same can not always be said for multimodal tasks because the moving vectors go towards different local optima. For an unknown multimodal problem (refer to Fig. 4.1), if all moving vectors are used to estimate a convergence point, there is a high probability of obtaining a wrong global convergence point due to the interaction among the moving vectors belonging to different subgroups. To make this method applicable to general multimodal optimization tasks, we must first automatically separate the population into several sub-groups according to each local optimum area and then apply the basic estimation method to estimate each local minimum. Subsequently, each sub-group can estimate its convergence point, and these estimated points are introduced into the current population with the expectation that this will accelerate convergence. In short, an effective separation method becomes the key to realizing the acceleration effect. If moving vectors are wrongly separated, it may lead the population to evolve in the wrong direction and slow down the convergence. Solving the separation problem is also one of the main contributions of this chapter.

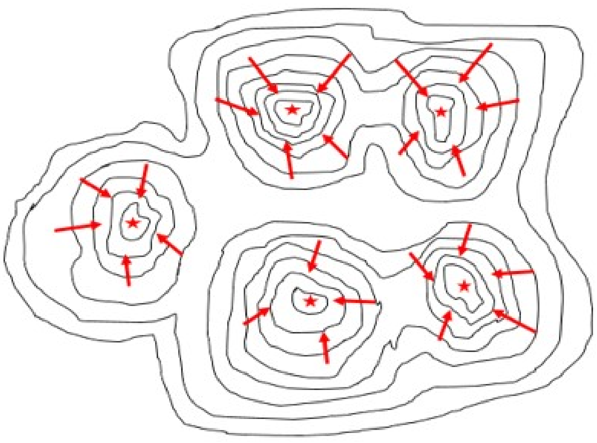


Figure 4.1: Moving vectors distribution in a general multimodal optimization.

In this part, a new separation method is proposed to automatically divide the whole population into several sub-groups according to each local optimum area by introducing a new indicator, which consists of the difference of two ranks: distance rank and fitness rank. Generally, there is only one optimal solution for unimodal optimization problems. Consider the optimal solution as a benchmark point; the farther an individual is from it, the worse that individual's fitness gets. Thus, distance rank and fitness rank have a high correlation. However, the fitness of some distant individuals becomes better because of the existence of local areas in multimodal optimization problems while the global optimal solution is still used as a benchmark point. This leads to a lower correlation between the two ranks. Fig. 4.2 reveals the correlation in different optimization scenarios, which may aid in understanding the relationship between the two ranks. In principle, the more valleys there are, the lower the correlation. We try to use the correlation between the two ranks to separate individuals.

For any given multimodal problem, the best individual in the current population has a high probability of being located in the optimal area that has been discovered so far. Thus, the current best individual is used as the benchmark point to calculate two ranks: distance rank and fitness rank. In this paper, we introduce a new indicator to evaluate the correlation of the two ranks for each individual, which is the value of its fitness rank minus its distance rank. If the value is negative, the lower

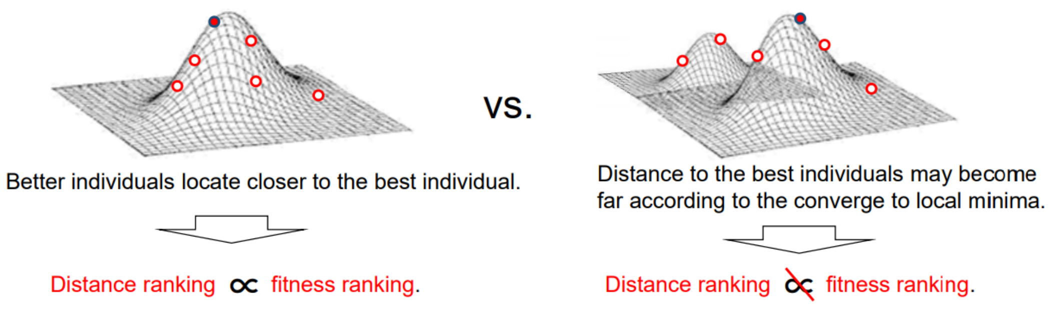

Better individuals locate closer to the best individual.
VS.
Distance to the best individuals may become far according to the converge to local minima.
Distance ranking ∝ fitness ranking.
Distance ranking ∝ fitness ranking.

Figure 4.2: Demonstrating the correlation of the two ranks in unimodal and multimodal optimization problems.

the correlation is. Thus, the proposed separation method divides the population into positive and negative sub-groups to identify individuals in different local areas in turn. Then, all individuals in the positive sub-group are considered to belong to the same local area and the remaining individuals in negative sub-group repeat the above operation until the termination condition is satisfied.

Note that the proposed method separates the current population instead of separating moving vectors directly. Next, an individual in a subgroup is paired with another individual, which may be either its offspring or a newly generate one. Here, we will explain in detail how to realize the separation according to each local optimum area. Fig. 4.3 demonstrates the framework of our proposed separation method for any given general multimodal optimization.

Step 1: Determine whether the optimization is unimodal or multimodal. If it is unimodal, stop. Otherwise, go to Step 2. The method used to make the determination is described below.

Step 2: The current best individual is used as a reference point and the distance variation between all other individuals and the reference point is calculated. They are then ranked to obtain the distance rank.

Step 3: The same reference point with Step 2 is used to calculate the fitness variation between all other individuals and the reference point. They are then ranked to obtain the fitness rank.

Step 4: Use the fitness rank minus the distance rank to obtain the difference of the two ranks. According to the positive/negative sign of the subtraction of two ranks, individuals are separated into a positive group and a negative group.

Step 5: The positive group is removed and kept as the first subpopulation, the negative group forms a new population. If the new population is unimodal, then stop. Otherwise, go to Step 6.

Step 6: The same operation is applied to the new population (go to Step 2), and the newly obtained positive group is kept as the second group. This iteration is repeated until all individuals are divided.

In this paper, we introduce a new parameter θ to detect whether the optimisation is unimodal or not. First, we calculate the average of the difference of two ranks of all individuals that are not divided yet. If the average value is less than the

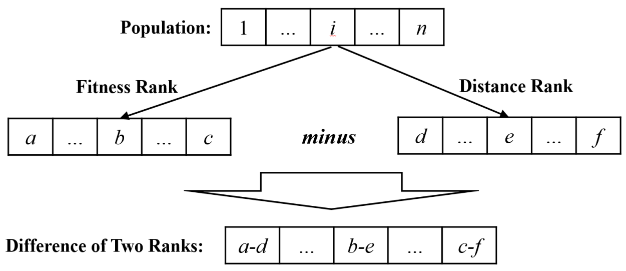


Figure 4.3: The framework of proposed method for moltimodal optimization.

predefined parameter θ, we think all undivided individuals belonging to the same peak or valley. Otherwise, this iteration is repeated until all individuals are divided.

The separation method uses a new indicator, the difference value of the two ranks, to detect peaks or valleys. Due to the existence of local optima, the individual fitness may get better around local optima as the individual moves away from the best individual while the distance to the best one is become farther and farther away. This leads to the difference of two ranks becoming greater around the local optima. Thus the new indicator can be used to detect and divide each subgroup one by one starting from the optimal subgroup and continuing to the worst subgroup. Fig. 4.4 shows the results of the successive separation of multiple local areas one by one.

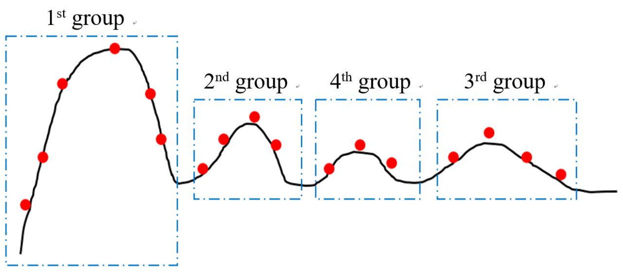


Figure 4.4: The result of successive separation of multiple local areas.

For any given multimodal optimization, the separation method can automatically divide the whole population into multiple subgroups. Then, the strategy for building moving vectors is adopted. In our evaluation experiments, we use *improvement 1* to create moving vectors and *improvement 2* to filter better individuals. Subsequently, the basic estimation method is applied to each sub-group to obtain multiple global or local estimated convergence points simultaneously. Algorithm 7 shows the flowchart of the EC algorithm combined with the basic estimation method and the proposed separation method to accelerate EC convergence.

4.1.3 Two Improvements for the Separation Method

Based on our previous research, there is a great potential to apply the estimation method to accelerate EC searches. Since different EC algorithms have their own characteristics, e.g. one parent generates one offspring individual in DE, while one

Algorithm 7 Estimated convergence points to accelerate EC. G: generation.

```
1: Generate an initial population.
2: Evaluate the fitness for each individual.
3: for $G = 1$ to $MaxGeneration$ do
4:     Obtain the next generation using the EC algorithm.
5:     Use the separation method to divide whole population into one or multiple subgroups.
6:     Use improvement 2 mentioned (Optional).
7:     Use improvement 1 mentioned (Optional).
8:     Obtain the estimated points and evaluate their fitness values.
9:     if their fitness values are better than those of the worst individuals in the population then
10:        use the estimated point(s) as elite individual(s), replace the worst individuals with the estimated points
11:    end if
12: end for
13: return the optimum
```

parent generates many offspring individuals in FWA, the following two improvements are proposed to better extend the estimation method to a variety of EC algorithms.

Improvement 1: Creating a moving vector in a tiny area

Because different EC algorithms use different strategies to generate their offspring, the length of th moving vectors cannot be guaranteed. Sometime, parents and their offspring are located in the same local area. Moving distance between a parent individual and its offspring sometimes becomes so long that the offspring may locate beyond the local minimum near the parent individual. Although the moving vector may aim to a different local minimum by chance, it is not adequate to use it to calculate a convergence point because it does not have the information required to go towards the local minimum near its parent individual.

A solution which allows us to create a moving vector going towards the local minimum near its parent individual is to create it in a tiny area that can be approximated with a hyperplane. Concretely speaking, we generate $\boldsymbol{c}_i$ within $1/p$ of each searching range around $\boldsymbol{a}_i$ offset by a uniform random number with the following formula. In our following experiment, we set $p = 200$.

$$c_i[j] = a_i[j] + rand(-1, 1) \times \frac{(\max\{v[j]\} - \min\{v[j]\})}{p}$$

where $a_i[j]$ and $c_i[j]$ are the j-th parameter value of $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$; $1 \leq j \leq d$, d is the dimension number. $\max\{v[j]\}$ and $\min\{v[j]\}$ are upper boundary and lower boundary of the j-th parameter, respectively.

Let two individuals with higher and lower fitness value be $\boldsymbol{c}_i$ and $\boldsymbol{a}_i$, respectively. Here, we can determine a moving vector $\boldsymbol{b}_i$. In ordinary EC, there is no guarantee that the fitness of the offspring will be better than that of its parent individual, and about half of the vectors from a parent individual to its offspring cannot be used to calculate their convergence point. However, with this improvement that lets the better individual between generated two individuals in a tiny area be $\boldsymbol{c}_i$, we can always generate a moving vector $\boldsymbol{c}_i$ going towards a local minimum. As all moving vectors can be used to calculate their convergence point, the precision of the local minimum estimate increases.

Although creating an offspring very near its parent individual slows convergence from the EC search perspective, using a well estimated convergence point as an elite individual speeds up convergence. One choice is to separate the EC search process from the process of creating moving vectors and estimating local minima. That is, the offspring generated by the EC process are not used to create a moving vector, but we let a randomly generated single point in a tiny area be $\boldsymbol{c}_i$, obtain a moving vector, and calculate the convergence point near the local minimum.

Improvement 2: Clustering and calculating a convergence point using only top moving vectors

The directions of moving vectors in valleys among local minima are influenced by multiple hills and valleys in the fitness landscape and do not always go toward one local minimum. It is easy to imagine that the estimation errors of local minima become bigger if these moving vectors are used for the estimation. Additionally, poor directions of moving vectors influences the clustering result and the calculation of the convergence point, so that the precision of the estimated local minima also become poorer.

The second proposed improvement is to use only the moving vectors from top individuals for clustering and the calculation of convergence points in the following improvements. This idea is premised on the idea that the fitness of individuals in these poorer areas is lower than those near the local minima. We may thus assume that the directions of moving vectors for individuals located closer to the local minima go towards their nearest local minima more correctly. In the following experimental evaluation, we use the average fitness as the criterion for selecting top individuals.

4.1.4 Experimental Evaluations

In our evaluations, we use 20 benchmark functions from the CEC2013 benchmark test suite [91] which is devoted to the approaches, algorithms and techniques for real parameter single objective optimization. Their landscape characteristics include shifted, rotated, unimodal and multi-modal. We test them with three dimensional settings, D = 2 (2-D), 10 (10-D), and 30 (30-D), respectively.

To analyze the acceleration effects of the proposed estimation framework, we combine it with DE and PSO in turn. Two controlled experiments are designed. The first experiment contrasts conventional DE and conventional DE with the proposed estimation framework, and the second experiment contrasts conventional PSO and conventional PSO with the proposed estimation framework. Then, one or multiple estimated convergence points are obtained and used as elite individuals to introduce into the population and replace the same number of worst individuals in

Table 4.1: DE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	20, 40, and 100
scale factor F	0.8
crossover rate	0.9
DE operations	DE/rand/1/bin
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	800, 10,000, 40,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

Table 4.2: PSO algorithm parameter setting.

population size for 2-D, 10-D, and 30-D search	20, 40, and 100
inertia factor w	1
constant c_1 and c_2	2 and 2
max. and min. speed V_{max} and V_{min}	1 and -1
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	800, 10,000, 40,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

the current population such that the population size is not changed . The DE and PSO experimental parameters are set as in Table 4.9 and Table 4.2, respectively.

Sine the proposed improvements and the basic estimation method will increase the number of fitness calculations, we evaluate DE and PSO convergence along the number of fitness calls instead of generations so that there is a fair comparison. In our evaluation experiments, both of the two *improvements* are used in DE, while only *improvement 2* is used in PSO. Furthermore, we always ensure that moving vector points from worse individual point to better ones to construct as many moving vectors as possible.

We test each benchmark function with 30 trial runs in 3 different dimensional spaces and apply the Wilcoxon signed-rank test on the fitness values at an earlier stage, a medium stage and a later stage of fitness calculations for both conventional EC and conventional EC with the proposed estimation framework. The statistical test results of experiment 1 and experiment 2 are presented in Table 4.3, Table 4.4 and Table 4.5, to show whether there is an significant difference in the different convergence periods between conventional EC methods and conventional EC methods with the proposed estimation framework.

Table 4.3: Statistical results of the Wilcoxon signed-rank test for average fitness values of 30 trial runs and 2 experiments in the early stage of the algorithm. $A \gg B$ and $A > B$ means that A is significantly better than B with significance levels of 1% and 5%, respectively. $A \approx B$ means that there is no significant difference between A and B. FEN means the number of fitness evaluations in performed at the time of evaluation. EXP1 refers to the first experiment, EXP2 refers to the second experiment. Numbers in the table represent 1: conventional DE, 2: conventional DE + universal estimation framework, 3: conventional PSO, and 4: conventional PSO + universal estimation framework.

Func.	2-D (FEN=100)		10-D (FEN=1,000)		30-D (FEN=5,000)	
	EXP1	EXP2	EXP1	EXP2	EXP1	EXP2
F_1	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_2	$1 > 2$	$4 \approx 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_3	$1 \gg 2$	$4 \approx 3$	$1 \gg 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_4	$1 > 2$	$4 > 3$	$2 > 1$	$4 \gg 3$	$2 \gg 1$	$3 \approx 4$
F_5	$1 \gg 2$	$4 > 3$	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_6	$1 > 2$	$3 \approx 4$	$2 \approx 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_7	$1 \gg 2$	$4 > 3$	$1 > 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_8	$1 \gg 2$	$4 \approx 3$	$1 \approx 2$	$4 \approx 3$	$2 \approx 1$	$3 \approx 4$
F_9	$1 \approx 2$	$4 \approx 3$	$1 \approx 2$	$4 \gg 3$	$2 \approx 1$	$4 \gg 3$
F_{10}	$2 \approx 1$	$4 \approx 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_{11}	$1 \approx 2$	$4 \approx 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_{12}	$1 \approx 2$	$4 > 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_{13}	$1 > 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$
F_{14}	$1 \gg 2$	$4 \approx 3$	$1 \approx 2$	$4 > 3$	$1 \approx 2$	$4 > 3$
F_{15}	$1 > 2$	$4 \approx 3$	$1 \approx 2$	$4 \gg 3$	$2 \approx 1$	$4 \gg 3$
F_{16}	$1 \gg 2$	$3 \approx 4$	$1 \approx 2$	$4 \approx 3$	$1 \approx 2$	$4 \approx 3$
F_{17}	$1 \gg 2$	$4 > 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \approx 3$
F_{18}	$1 \approx 2$	$4 \approx 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \approx 3$
F_{19}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_{20}	$1 \approx 2$	$4 \approx 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$4 \approx 3$

4.1.5 Discussions

Analysis of the estimation method

Most EC algorithms locate the global optimal solution through multiple repeated iterations, while the proposed estimation method has a high possibility to calculate the potential location of the global optimum roughly. It does not need to know any prior information about the optimization problems and does not have high cost consumption. Thus, the estimation method has great potential, especially for high-dimensional or high-cost optimization problems.

When the estimated point is used as an elite individual to accelerate EC search, it will replace the worst individual in the current population. If they have good fitness and are close to an optimal area, they may quickly find the optimal solution in subsequent searches, thus simultaneously accelerating the convergence speed and

Table 4.4: Statistical results of the Wilcoxon signed-rank test for average fitness values of 30 trial runs of 2 experiments in the middle stage of the algorithm. The symbols used in this Table have same meaning as in Table 4.3.

Func.	2-D (FEN=400)		10-D (FEN=5,000)		30-D (FEN=20,000)	
	EXP1	EXP2	EXP1	EXP2	EXP1	EXP2
F_1	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$
F_2	$1 \gg 2$	$4 \gg 3$	$2 > 1$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$
F_3	$1 \gg 2$	$3 \gg 4$	$2 \gg 1$	$4 \approx 3$	$2 \gg 1$	$4 \approx 3$
F_4	$1 \gg 2$	$4 \gg 3$	$1 \approx 2$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_5	$1 \gg 2$	$4 > 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_6	$1 \gg 2$	$3 \approx 4$	$2 \approx 1$	$3 > 4$	$2 \gg 1$	$3 \approx 4$
F_7	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$3 > 4$	$2 \gg 1$	$4 \approx 3$
F_8	$2 > 1$	$4 \gg 3$	$1 > 2$	$3 \approx 4$	$1 \approx 2$	$3 \approx 4$
F_9	$2 \approx 1$	$4 \gg 3$	$2 \gg 1$	$4 \approx 3$	$1 \approx 2$	$4 \approx 3$
F_{10}	$1 \gg 2$	$4 \gg 3$	$2 \gg 1$	$4 > 3$	$2 \gg 1$	$3 \approx 4$
F_{11}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{12}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{13}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{14}	$1 \approx 2$	$4 > 3$	$2 \approx 1$	$4 > 3$	$1 \approx 2$	$4 \gg 3$
F_{15}	$2 \approx 1$	$4 \gg 3$	$1 \approx 2$	$4 \gg 3$	$2 \approx 1$	$4 \gg 3$
F_{16}	$1 \gg 2$	$3 \approx 4$	$1 \approx 2$	$3 \approx 4$	$1 \gg 2$	$3 \approx 4$
F_{17}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \approx 4$
F_{18}	$1 > 2$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{19}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \approx 3$	$2 \gg 1$	$3 \approx 4$
F_{20}	$1 \approx 2$	$4 \approx 3$	$2 > 1$	$3 \gg 4$	$2 \gg 1$	$3 > 4$

improving the accuracy of the found optimal solution. When they are not close to the global optimum, unfortunately, they may not be remarkable - but they are still better than the worse individuals they replace. The increased cost is acceptable, but the performance improvement is enormous. Consequently, introducing the estimated convergence points is a valuable investment from the cost-performance point of view, and we can say that it is a *low risk, high return* strategy.

Analysis of compositions of proposal

The first discussion is the superiority of our proposed separation method. It does not require additional fitness computation costs and automatically divides the population into several subgroups as the population converges. When it is combined with EC algorithms, it also does not require modification of their optimization framework. Furthermore, it divides from the best local area to the poorest local area gradually, which leads to the found best local area being more favored and places priority on guaranteeing its separation accuracy. Thus, we can say that the proposed separation method is low cost and easy to use.

The second discussion involves analyzing our proposed generic estimation framework, which is mainly composed of three parts: the separation method, the EC

Table 4.5: Statistical test result of the Wilcoxon signed-rank test for average fitness values of 30 trial runs of 2 experiments in the last stage of the algorithm. The symbols used in this Table have same meaning as in Table 4.3.

Func.	2-D (FEN=800)		10-D (FEN=10,000)		30-D (FEN=40,000)	
	EXP1	EXP2	EXP1	EXP2	EXP1	EXP2
F_1	$2 \approx 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$3 \gg 4$
F_2	$1 \gg 2$	$4 \gg 3$	$1 > 2$	$4 > 3$	$2 \gg 1$	$3 \gg 4$
F_3	$1 \gg 2$	$4 \gg 3$	$2 \gg 1$	$3 \approx 4$	$2 \gg 1$	$3 \approx 4$
F_4	$1 \gg 2$	$4 \gg 3$	$1 \gg 2$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_5	$1 \gg 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$
F_6	$1 \gg 2$	$3 \approx 4$	$1 \gg 2$	$3 \approx 4$	$2 \gg 1$	$3 \approx 4$
F_7	$1 \approx 2$	$4 > 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$4 > 3$
F_8	$2 \gg 1$	$4 \gg 3$	$1 \approx 2$	$4 \approx 3$	$1 \approx 2$	$3 \approx 4$
F_9	$2 \approx 1$	$4 > 3$	$2 \gg 1$	$3 \approx 4$	$2 \approx 1$	$3 \approx 4$
F_{10}	$1 \approx 2$	$4 \gg 3$	$2 \gg 1$	$4 \gg 3$	$2 \gg 1$	$3 > 4$
F_{11}	$1 \approx 2$	$3 \approx 4$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{12}	$1 \approx 2$	$4 > 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{13}	$1 \approx 2$	$3 \approx 4$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{14}	$1 \approx 2$	$4 \approx 3$	$2 \approx 1$	$4 \approx 3$	$1 > 2$	$4 \gg 3$
F_{15}	$2 \approx 1$	$3 \approx 4$	$2 \approx 1$	$4 \approx 3$	$2 \approx 1$	$4 \gg 3$
F_{16}	$1 \gg 2$	$3 \approx 4$	$1 \approx 2$	$4 \approx 3$	$1 \gg 2$	$3 \approx 4$
F_{17}	$1 \approx 2$	$4 > 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 > 4$
F_{18}	$1 \approx 2$	$4 \approx 3$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$
F_{19}	$2 \approx 1$	$4 \approx 3$	$2 \gg 1$	$4 \approx 3$	$2 \gg 1$	$4 \approx 3$
F_{20}	$2 \approx 1$	$3 \approx 4$	$2 \gg 1$	$3 \gg 4$	$2 \gg 1$	$3 \gg 4$

algorithms and the acceleration method. They are independent of each other and will not interact with each other. Thus, we can switch to different strategies in these three parts when applied to different characteristics of optimization problems. For example, different EC algorithms or other separation methods can be used. Thus, we can say that the proposed framework is versatile and easy to use. Although the proposed framework adds one or more additional fitness calculations to obtain estimated convergence points, it is worthwhile to obtain estimated individuals that have a high probability of being close to the real global solution. In this paper, we replace the worse individuals with the estimated convergence points to accelerate EC convergence. Actually, there are many other ways to use them to accelerate EC. For example, the estimated point can be used as a base vector in DE, or it can be used to generate offspring individuals. Hence, how best to use estimated points efficiently will also be a new topic.

The third discussion focus on how to improve the cost-performance. Here, the *improvement 1* and *improvement 2* are adopted in our evaluation experiments, where *improvement 1* will increase the additional fitness calculations while *improvement 2* does not add any. Suppose that the population size is P, the *improvement 1* need P extra fitness calls. In fact, since *improvement 2* is executed first, it may reduce the fitness cost consumption by about half. The experimental results revealed that those two improvments can further enhance acceleration effect in spite of increasing cost. We should find a balance between the computational cost and the convergence performance, taking into account the complexity of the tasks and the computational cost of the fitness calculations. Some methods for controlling the balance include: using only some of the improvements, applying our proposed method at every k-th generation, and others.

The fourth discussion addresses the applicability of the proposed estimation framework. In this paper, two evolutionary algorithms, DE and PSO, are used and combined with our proposed framework, respectively, where both algorithms have their own different characteristics. The best individual in the current population has a faint effect in the DE/rand/bin/1, however it has a strong impact in PSO, so that all the poor individuals will slowly converge toward the current best one in DE and rapid converge to the best one in PSO. The two control experiments use the same initial population, but, the individual distribution of PSO becomes more and more close to that of DE as the algorithm converges. We can speculate that if the estimated points become the current best individuals, then this will accelerate the movement of the surrounding points to them. However, these situations do not occur easily in DE. This can also be confirmed by referring to Table 4.3, Table 4.4 and Table 4.5; we find that PSO more easily falls into local solutions than DE in the high dimensional spaces. Based on the above considerations, we do not recommend applying the estimation method to an algorithm where the best individual has a strong impact to others. Thus, how to use estimated points more rationally will also become a a promising topic in our future works.

We apply the Wilcoxon signed-rank test between conventional EC and conventional EC with our proposed method at different searching periods in both controlled experiments. From the results of this statistical test, we can see that PSO realizes a better acceleration effect, while DE does benefit, or even performs worse in low dimensional spaces. This may be because DE requires extra computational cost and the low dimensional problems are not complicated. Thus, the cost consumed reduces the speed of convergence without realising the desired performance improvement. However, the situation is just the opposite in the high dimensional space. With the increase in dimensions, the complexity of the optimization problem increases rapidly, and the local optima become more numerous. In the early stages of the algorithm, PSO still maintains a fast convergence rate, but it is more likely to fall into a local solution as time goes on and not play a role in accelerating promotion. On the contrary, DE can always maintain a strong acceleration effect, although to do so it requires additional computations. This also shows that the balance of cost-performance needs to be considered for different optimization problems.

Analysis of the parameter settings of the proposal

Through the above analysis, we can notice that the accuracy of the estimated points depends heavily on the results of the separation method. In this paper, we introduce a new parameter, θ to determines when remaining individuals belong to the same subgroup and terminate the separation method. However, there is no unified method to guide the parameter tuning; in practical applications and various optimization problems, we often do not have sufficient prior knowledge and it is difficult to choose the appropriate parameters. In our evaluation experiments, the parameter setting for θ is based on experience. If the θ is too large or too small, it will be disadvantageous to the division of the population. Thus, it is more reasonable to adjust θ automatically according to the characteristics of optimization problems, rather than to fix the parameter based on experience. Next, we also plan to introduce some mechanisms to dynamically adjust θ. Furthermore, we have currently only applied the proposed separation method to the current population; it may be better to apply it to multiple successive generations and to assign individuals to the same subgroup which are now assigned to the same group in different generation. This can improve the accuracy of separation by making multiple confirmations. In short, improving the precision of individual separation as much as possible is an important topic.

We use the difference between the two ranks to gradually divide the population. In fact, it is better to introduce another new parameter, ξ instead of using 0 to determine whether an individual belongs to a positive group or a negative group. If the absolute value of the difference between two ranks is less than ξ, then this individual can be assigned to a positive group. We hope that this new parameter, ξ can improve anti-interference because some optimization problems have noise. It can also increase the fault tolerance of the separation and is expected to reduce the possibility of an erroneous separation. Although this is all currently our supposition, we will soon investigate and confirm whether we are correct.

Finally, we want to discuss the relationship between dimension and the number of moving vectors used to estimate the convergence points. Too many or too small moving vectors may affect the accuracy of the estimated convergence points. Honestly, we can not give an explicit relationship between them; we intend to investigate the effect of different combinations of dimension and the number of moving vectors on different functions through a large number of investigations in future works, which should eventually give a rough suggestion as to their relationship.

Potential of the method and future topics

Although this paper is only a preliminary extension of the basic estimation method, it shows that our proposed estimation framework is effective and promising. There are still many open topics and improvements waiting to be studied. The following is a non-exhaustive list of some future topics.

(1). The construction of the moving vector. Whether moving vectors correctly point to the optimal area affects the accuracy of estimated point. Thus, it is essential to build a reasonable moving vector. In this paper, a moving vector is paired through the relationship between parent and its offspring or randomly generating a new searching point. In fact, EC algorithms for parent-child relationships that are not one to one, i.e GA and FWA, can also be combined with our proposed frameworks. In these scenarios, we can use the shortest distance to match the moving vectors between two successive generations, or the other mechanisms of pairing. Besides, instead of using two consecutive generations, we can use the average evolutionary path of multiple successive generations to construct moving vectors. This may remove some randomness and the enhance anti-interference capability, which helps to improve the accuracy of the estimated point. In general, there are still many construction methods waiting to be developed by us.

(2). Improve performance and reduce cost. So far, we have proposed several improvements to further improve the performance of our proposal. Although the experimental results show that the more improvements are made, the more significantly the performance is improved. However, some improvements increase greatly the number of extra fitness calculations, which leads to a failure to improve in some cases, due to the high cost of calculating and the limited number of fitness calculations. For these cases, it is a good choice to use outstanding individuals from past generations to construct moving vectors, using e.g an optional external archive and an individual pool. Anyway, using known information efficiently and proposing effective but lower-cost improvements are also a direction that we continue to adhere to.

4.2 Accelerating Multiobjective Optimization

4.2.1 Multiobjective Optimization

Multi-objective optimization problems lie in many real-world applications and they contain multiple optimization objectives that conflict with each other. This makes conventional optimization algorithms (deterministic optimization method), difficult to apply when solving these problems. One solution of the multi-objective optimization problems is to transform multiple objectives into a single objective by assigning different weights to each objective for a combination. This requires us to have a degree of deep understanding of multi-objective optimization problems.

Currently, more popular approaches use evolutionary multi-objective optimization (EMO) algorithms because of various well defined features and characteristics, such as strong robustness, ease of use, intelligence and others. Almost of these pay attention to finding a set of trade-off optimal solutions (known as Pareto optimal solutions), instead of a single optimal solution. The Pareto dominance and diversity of solutions are two primary subjects in EMO research. One attempts to obtain many non-dominated Pareto solutions, and the other tries to obtain Pareto solutions in a wide areas on Pareto solution front.

4.2.2 Acceleration Framework for Multiobjective

There are two subjects studied in the EMO algorithm research field; one study is the Pareto dominance issue, and the other one is EMO solution diversity issue. Almost all research on these two issues focuses on special handling in a objective space of an EMO algorithm, and frequently ignore the search condition in a parameter space. EMO algorithms try to find more non-dominated solutions with diversity. The solutions on the first Pareto solution frontier from the last generation to the next in an objective space supports information on how moving the variables in a parameter space can find promising Pareto solutions.

In Figure 4.5, we can find a set of the pairs of moving vectors in a parameter space in accord with the Pareto dominance information obtained in an objective

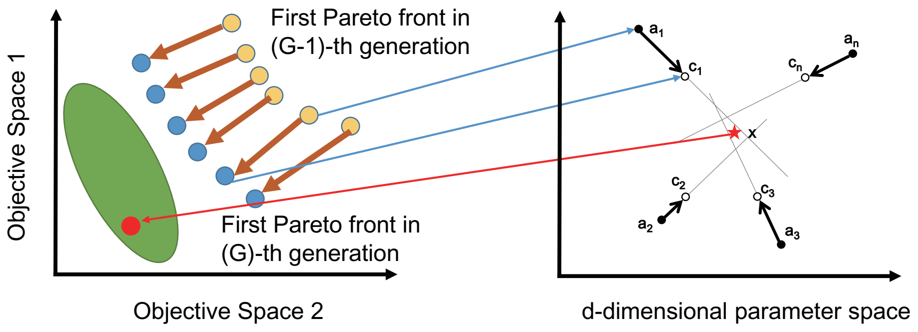


Figure 4.5: Estimation of promising Pareto solution area in parameter space using the dominance information from objective space to enhance EMO search.

space. We can use the moving vector information to estimate a convergence point that presents a promising area where Pareto solutions would be in the a parameter space. We put such an estimated convergence point of a parameter space into EMO search, and remove one of dominated solutions. EMO search should be enhanced considering such search information, and hopefully, EMO algorithm can find more non-dominated Pareto solutions quickly. This is a study hypothesis and motivation of our proposal that utilities an estimated convergence point to accelerate EMO search.

There are three primary steps and/or issues in the proposed method to enhance EMO search. The first step/issue is how to make pairs of moving vectors in a parameter space from Pareto improvement information obtained in an objective space. We make two candidate groups of non-dominated solutions in the current generation and in the last generation, so Pareto solution improvement information can be obtained from these two group individuals. Here, we design two methods to make moving vector pairs.

- We pick up one of non-dominated solutions in an objective space from one group, and find the nearest non-dominated solution in the other group, and then find their corresponding individuals in a parameter space to make these two solutions form a pair. (Estimation in objective space)
- We pick up one of non-dominated solutions in an objective space from one group, and find its corresponding individual in a parameter space, and then find this individual's nearest individual in a parameter space to make these two solutions form a pair. (Estimation in parameter space)

After this, we delete these two solutions from the two groups, and we repeat this processing until one of groups becomes empty.

The second step estimates a convergence point in a parameter space using the moving vector pairs obtained from the first step. The estimated point has high potential in the non-dominated Pareto solution frontier, and can therefore accelerate EMO search. Besides estimating only one estimated point, we can also estimate one point from only one single objective space each by each individually and use them together to accelerate EMO search (Estimation in each single objective space).

In the third step, we put the estimated convergence point as a search elite individual into EMO algorithms, and delete one/more of the dominated solutions in the current generation to enhance EMO search. This is the primary implementation within our proposal.

Table 4.6: Multi-objective benchmark function used in evaluation [?]. All of the Pareto frontier are $g(x) = 1$.

Functions	Definition
ZDT1	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{\frac{x_1}{g(x)}}]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$
ZDT2	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - (\frac{x_1}{g(x)})^2]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$
ZDT3	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)} \sin(10\pi x_1)]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$
ZDT4	$f_1(x) = x_1$ $f_2(x) = g(x)[1 - (\frac{x_1}{g(x)})^2]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}[x_i^2 - 10\cos(4\pi x_i)]$
ZDT6	$f_1(x) = 1 - \exp(-4\pi x_1)\sin^6(6\pi x_1)$ $f_2(x) = g(x)[1 - (\frac{f(x_1)}{g(x)})^2]$ $g(x) = 1 + 9[\frac{\sum_{i=2}^{n} x_i}{n-1}]^{0.25}$

4.2.3 Experimental Evaluations

Experiment Setting

We use five multi-objective benchmark functions from the ZDT test suite [211] to evaluate our proposed methods. We embed our proposed method into conventional NSGA-II [40] with different constructing methods of moving vector, and compare our proposed method with NSGA-II. Table 4.6 presents the benchmark function's mathematical expressions. We examine these functions with three dimensional settings, i.e. two dimension (2-D), 10-D, and 30-D. Table 4.7 shows the parameter settings of conventional NSGA-II algorithm used in the evaluation experiments.

Three experiments are designed where different methods of constructing moving vectors, and these combined with the conventional NSGA-II algorithm. The legends displayed in figures and tables have the following meanings.

- **NSGA-II:** conventional NSGA-II algorithm;
- **Estimation in objective space:** we construct moving vectors from two subsequent non-dominated solution set in an objective space;
- **Estimation in parameter space:** we find the nearest offspring individual for each one in a parent generation, and make pairs in a parameter space; and

- **Estimation in each single objective space:** we consider each objective independently and estimation convergence point for each objective, where the estimated points may not be best on all objectives, but they have good potential in some objectives.

Table 4.7: NSGA-II algorithm parameter setting.

population size for 2-D, 10-D, and 30-D	20, 50, and 100
crossover rate	0.8
mutation rate	0.05
max. # of fitness evaluations, MAX_{NFC}, for2-D, 10-D, and 30-D search	400, 1,000, and 10,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

Evaluation Metrics

We set the stop conditions of each evaluation using the number of fitness calls instead of generations for fair evaluation, because our proposed methods increases additional fitness cost consumption. We set the stop conditions as 400 times, 1000 times, and 10000 times of fitness evaluations in 2-D, 10-D, and 30-D problems, respectively. Besides, we test each benchmark function with 30 trial runs in three different dimensional settings.

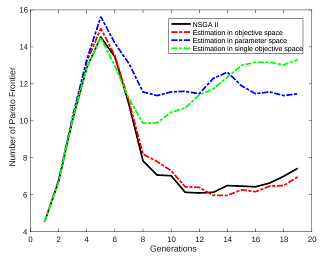
Conventional NSGA-II is adopted as an example algorithm, definitely, other EMO algorithms can be also applied. Although there are many different ways to generate estimated points, the greedy replacement strategy, where the estimated points will replace with the worst ranked and low diversity individuals to keep the same population size, is adopted in the proposed acceleration framework. To analyse the effect of the proposed acceleration framework, we calculate the number of non-dominated Pareto solutions in each generation shown in Figures 4.6, 4.7, and 4.8.

Hyper volume [210] is used to evaluate the diversity and acceleration performance of our proposal. Table 4.8 presents the hyper volume values of our proposed method and conventional NSGA-II algorithm at the stop condition in three different dimensional settings. We apply Wilcoxon signed-rank test for 30 trail runs data to evaluate the significance of hyper volume obtained by conventional NSGA-II and our proposal. Some functions without hyper volume value is due to reference point [-1, 1] setting.

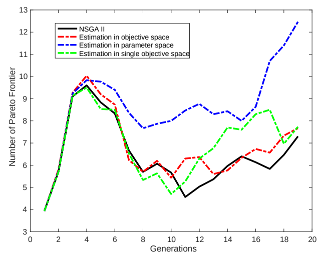
4.2.4 Discussions

Pareto Improvementation of the Proposal

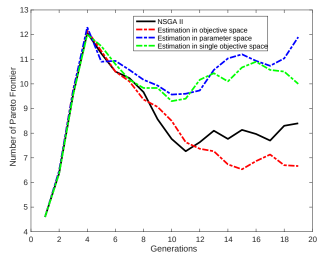
Pareto dominance and Pareto solution diversity are two metrics to evaluate the performance of EMO algorithms. In this work, we calculated the average number of Pareto solutions in every generation for each benchmark problem; see Figures 4.6, 4.7, and 4.8. This is one of evaluation metrics for Pareto dominance in EMO. We



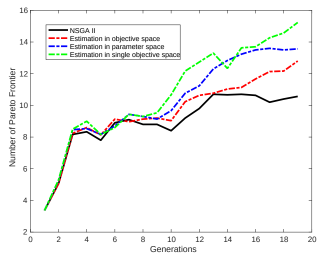
ZDT1



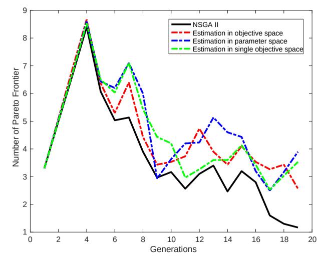
ZDT2



ZDT3

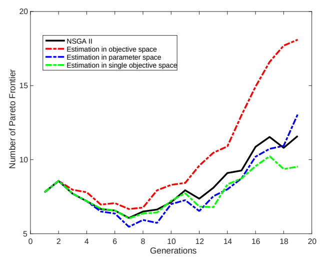


ZDT4

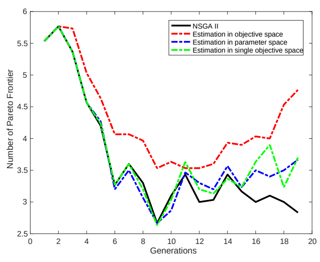


ZDT5

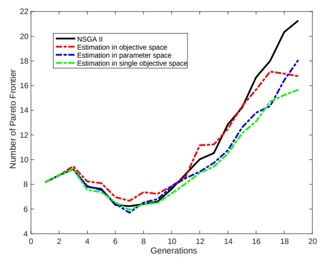
Figure 4.6: The number of Pareto solutions in every generation for 2-D benchmark problems. We can observe that proposed method can obtain more Pareto solutions for the most of cases.



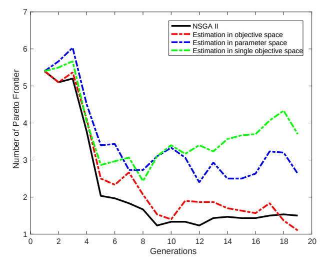
ZDT1



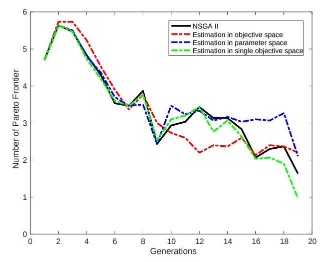
ZDT2



ZDT3



ZDT4



ZDT5

Figure 4.7: The number of Pareto solutions in every generation for 10-D benchmark problems. We can observe that proposed method can obtain more Pareto solutions for the most of cases.

Table 4.8: The average hypervolume values from 30 trials running of 4 methods in 2 dimension (2-D), 10-D, and 30-D. Symbol † means that there is a significant difference between NSGA-II and Proposed method, i.e. NSGA-II + Estimation Point. The reference point is [1,1]. Obj., Para., and SinglePara. present objective space, parameter space, and each single parameter space, respective.

2-D tasks

Func.	NSGA-II	Estimation in Obj.	Estimation in Para.	Estimation in SinglePara.
ZDT1	0.414567	0.417300	0.453833†	0.480533 †
ZDT2	0.109833	0.117367	0.124367	0.113867
ZDT3	0.556733	0.552433	0.622733†	0.622600†
ZDT4	0.255733	0.261600	0.284933†	0.337167†
ZDT6	0.000033	0.002033	0.001833	0.000967

10-D tasks

Func.	NSGA-II	Estimation in Obj.	Estimation in Para	Estimation in SinglePara
ZDT1	0.328033	0.337767	0.345533	0.339433
ZDT2	0.008633	0.012367	0.010100	0.008933
ZDT3	0.564067	0.545767	0.589400	0.588933

30-D tasks

Func.	NSGA-II	Estimation in Obj.	Estimation in Para	Estimation in SinglePara
ZDT1	0.647167	0.654000	0.651533	0.648633
ZDT2	0.183333	0.157567	0.187700	0.190433
ZDT3	0.796133	0.792600	0.791533	0.794767
ZDT6	0.066233	0.068433	0.069633	0.066733

also calculated hyper volume values at the maximal number of function calls for each dimension setting, and applied Wilcoxon signed-rank test to verify the significant difference among hyper volume values in Table 4.8. This is a demonstration of Pareto solution diversity for each EMO algorithm. We analyse and discuss our proposed method using these results.

From Figures 4.6, 4.7, and 4.8, we can observe that methods estimating a convergence point in a parameter space and in a single objective space can obtain more Pareto solutions from all five multi-objective benchmark problems in 2-D setting. Method estimating a convergence point in an objective space fails in two benchmark functions, i.e. ZDT1 and ZDT3 in 2-D tasks. It indicates that moving vectors constructed from information of the nearest points in an objective space cannot exactly estimate the non-dominated Pareto frontier area in a parameter space. The same case can also be found in 10-D benchmark setting for ZDT3 and ZDT4, and 30-D benchmark setting for ZDT1, ZDT3, and ZDT4. We need to further consider to improve the estimation the accuracy of estimation method of a convergence point in an objective space.

That in a single objective space works well in most of cases because this method replaces more than one estimated convergence point, and increases the population diversity for EMO algorithms. This indicates that the better individuals in each objective can improve optimization performance of EMO algorithms, although there are conflicts among multi-objective functions when EMO searches for non-dominated Pareto solutions. From this viewpoint, elite strategy-based EC acceleration methods can be applied not only in single objective problems, but also have a potential to be

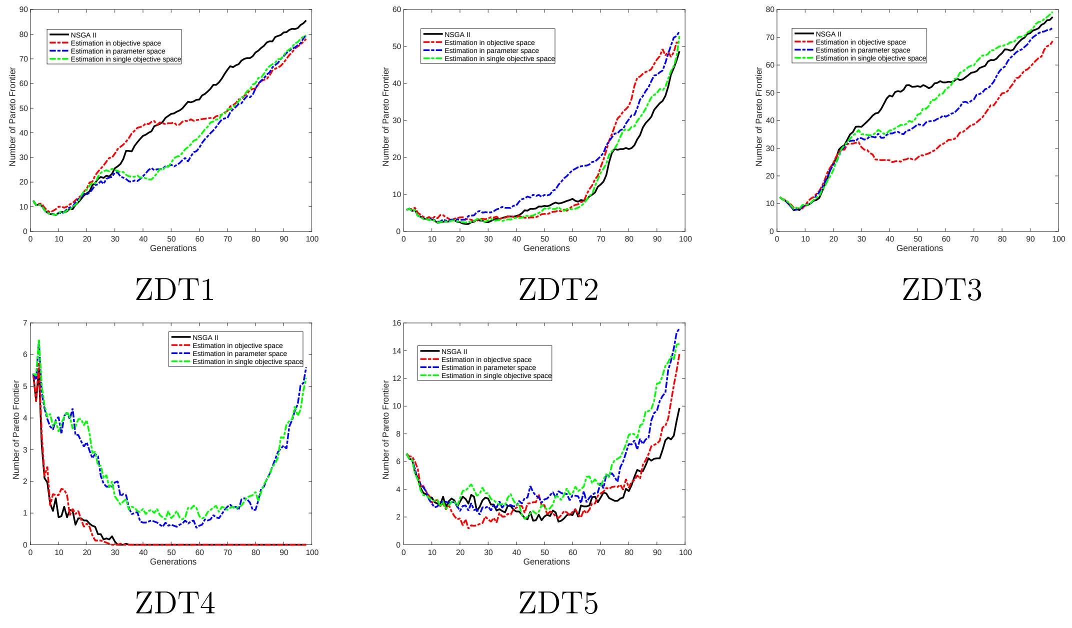


Figure 4.8: The number of Pareto solutions in every generation for 30-D benchmark problems. We can observe that proposed method can obtain more Pareto solutions for the most of cases.

applied in multi-objective problems

From observation of Table 4.8, the values of hyper volume from our proposed method are bigger than those from conventional NSGA-II algorithm for the most of tasks in 2-D benchmark problems. The Wilcoxon signed-rank test results showed a significant difference between our proposed method and the conventional NSGA-II algorithm in estimation in a parameter space and estimation in a single objective space. These results demonstrate that our proposed method can obtain non-dominated Pareto solution with more diversities for EMO algorithms. However, it is not significant shown in 10-D and 30-D benchmark problems. It is a limitation for our proposal, and we need to improve it in our future work.

Topological Structure of Moving Vectors and Modality Characteristic of Pareto Improvement

The basic philosophy of our proposed method to accelerate EMO search lies in three hypotheses. First, we can obtain the information to improve non-dominated Pareto solutions through Pareto solution evolutions from the last generation to the current generation in an objective space. Second, after we obtain the information, the moving vectors can be made in an objective space or in a parameter space. Third, the estimated convergence point of these moving vectors has a high possibility that locals in the non-dominated Pareto solution frontier area in a parameter space. From the modality viewpoint, the distribution of Pareto solutions shows a uni-modal characteristic in the objective space. In the case of the Pareto improvement from the last generation to the current generation, do the corresponding individuals also present a uni-modal distribution characteristic in a parameter space? We examine this question here.

We present improved EMO evolutions along three generations' EMO evolution condition both in an objective space and a parameter space for these benchmark functions with 2-D setting (see Figure 4.9). The arrows demonstrate the Pareto improvement directions between two generations in both spaces. From Figure 4.9, we observe that with regards to the directions of arrows in an objective space, all of them are towards almost the same direction, i.e. their angles are less than 90 degrees. However, in the parameter space, the arrows are not towards the same direction. It displays a multi-modal distribution characteristic, e.g. in the ZDT4 and ZDT6 benchmark problems. From these observations, it indicates that Pareto improvement in the objective space presents a uni-modal characteristic, while it presents a multi-modal characteristic in a parameter space. In Figure 4.6, the numbers of the first Pareto frontier solution from four methods are almost the same, but their acceleration performances are not significant. This is one our discovery on the modality characteristic of Pareto improvement in both an objective space and a parameter space.

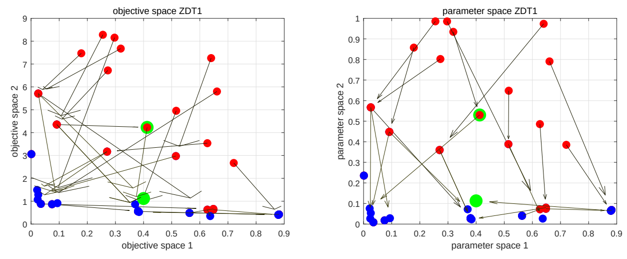
From Figures 4.9, there is a multi-modal characteristic in a parameter space when the Pareto improvement occurs from one generation to the next. The third hypothesis of proposed method is not always correct, therefore, the proposed method can works well in the uni-modal condition of Pareto improvement, and by chance well in the multi-modal one. From Table 4.8, there is not a significant difference between NSGA-II algorithm and our proposed method in ZDT6. This experiment's results verify our analysis and observations. The multi-modal characteristic of Pareto improvement in a parameter space is an issue when applying our proposal to enhance EMO search.

4.3 Accelerating Interactive Evolutionary Computation

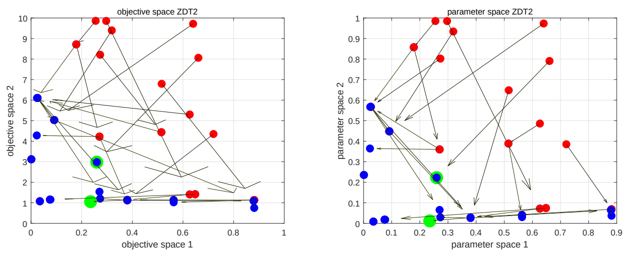
4.3.1 Interactive Evolutionary Computation

IEC is an extension of the traditional EC that integrates human subjective evaluations to design products that contain human preferences. Figure 4.10 shows a universal optimization framework of classic IEC algorithms. Since humans are an important part of the whole optimization system, IEC has a unique feature, i.e. human sensibility, that traditional EC algorithms do not have. Meanwhile, there are also new problems arising from the presence of a human. The most urgent problem is that human fatigue limits the performance of IEC because humans can not evaluate individuals for a long time. Many researchers have proposed various ways to alleviate this limitation, such as by improving user interface and reducing human-computer interaction [156]. Additionally, alternative models have also been used to reduce the number of user evaluations [115].

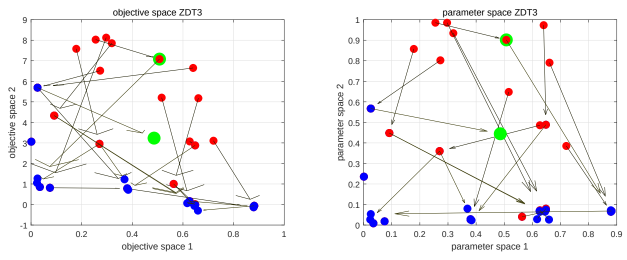
The estimation method has a high probability of estimating the area of the global optimum, so we try to extend it to accelerate real world IEC applications and thus reduce user fatigue. Because humans can not distinguish subtle differences between individuals as is done by computers, we first investigate the feasibility of using an estimated convergence point to accelerate IEC convergence before applying it to



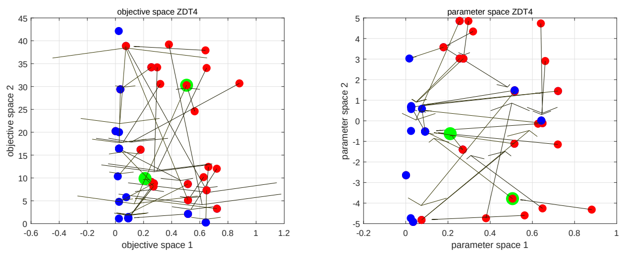
ZDT1



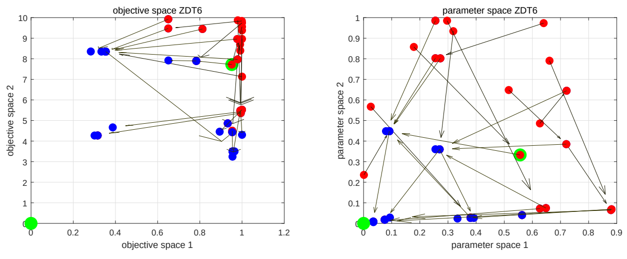
ZDT2



ZDT3



ZDT4



ZDT6

Figure 4.9: Two dimensional demonstration of Pareto solution improvement in an objective space (left) and their corresponding individuals in parameter space (right) of ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6. The arrows show directions of both Pareto solution improvement and moving vectors. We can observe that there is a uni-modal landscape for Pareto solution improvement in an objective space, however, it is a multi-modal landscape for Pareto improvement in a parameter space. The green point is the estimated convergence point, most of the red points and most of the blue points are in the first generation and in the third generation, respectively.

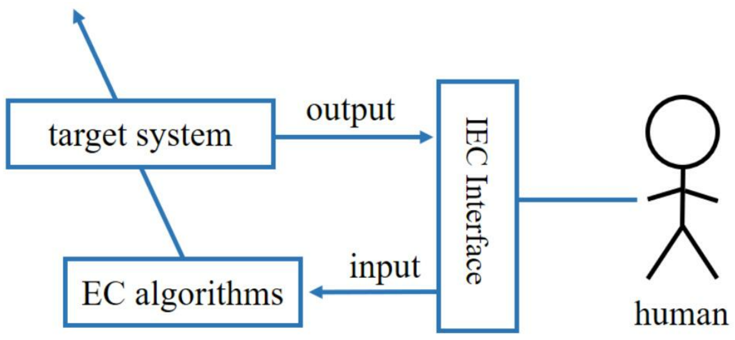


Figure 4.10: A universal optimization framework for IEC.

IEC applications.

4.3.2 Acceleration Framework for IEC

The basic estimation method can calculate a convergence point for the population and use it as an elite individual to replace the worst individual in the current population to speed up convergence. In fact, humans cannot distinguish small differences between two similar individuals, and they may thus be given the same score in IEC evaluations, while traditional EC can clearly distinguish these differences. Because we cannot say which one is the better for multiple individuals with same score, we employ a stochastic selection strategy in the proposed method. This means an estimated convergence point will replace a randomly selected individual from all worst individuals to keep the population size unchanged.

At the first step of applying the basic estimation method to accelerate IEC, we need to make enough moving vectors from parents to their offspring, which is critical to the estimation of the convergence point. In IEC applications, it often happens that there is no difference between a parent individual and its offspring when users cannot choose which is better and thus give both the same score. To increase the number of vectors, we reverse a parent individual and its offspring if the score of the parent is high than that of its offspring. Of course there are other ways to increase the number of vectors, for example, a moving vector is not limited to point from the parent individual to its own offspring, but also to other better offspring. At the current stage, we are not concerned with the latter method of constructing moving vectors, however, it will be used in one of our follow-up studies.

In the first attempt, we focus on investigating the feasibility of our proposed method. So we just use a traditional EC to simulate IEC, and do not use an actual human to perform the evaluations. Next, we explain how to convert the EC evaluation to an IEC evaluation to simulate an IEC experiment. Suppose that the EC is used to optimize a minimization problem and the higher the score, the better the individual in IEC. We use $f_{EC}(x_i)$ to represent the fitness value of the i-th individual and F_{max} and F_{min} means the worst and best individual in EC evaluation, respectively. So, we can use following Eq. (4.1) to complete the conversion from EC to IEC.

Algorithm 8 The general framework of the proposed method for accelerating IEC.

```
1: Generate an initial population.
2: Evaluate the fitness for each individual.
3: for $G = 1$ to $MaxGeneration$ do
4:     Obtain the next generation using the EC algorithm.
5:     Convert the EC evaluation to an IEC evaluation for two subsequent generations.
6:     Construct the vectors for estimating the convergence point and evaluate its fitness using the EC evaluation.
7:     Convert the EC evaluation of the estimated point to an IEC evaluation.
8:     if Its score is higher than that of the worst individuals then
9:         use the estimated point to replace a randomly selected individual from all the worst individuals.
10:    end if
11: end for
12: return the optimum
```

$$f_{IEC}(x_i) = ROUND(\frac{F_{max} - f_{EC}(x_i)}{F_{max} - F_{min}} * s) \tag{4.1}$$

Where, $ROUND()$ is a rounding function, $f_{IEC}(x_i)$ indicates that the i-th EC evaluation individual is converted to an IEC evaluation. s is the evaluation levels in the IEC and we set s to 5 in the following simulation experiments. Finally, we can obtain the proposed algorithm as shown in Algorithm 13.

4.3.3 Experimental Evaluations

To evaluate the effectiveness of the proposed method, the Gaussian mixture model given by Eq. (4.2) that consists of n d-dimensional (d =2,5 and 10) Gaussian functions (n =1, 2, 3 and 4) is used in our experimental evaluation.

$$f(x) = -\sum_{j=1}^{n} \left\{ a_j \exp\left(-\sum_{i=1}^{d} \frac{(x_{ij} - \mu_{ij})^2}{2\sigma_{ij}} \right) \right\} \tag{4.2}$$

The DE parameter settings are set as in Table 4.9. For fair evaluations, we evaluate simulated IDE convergence along with the number of fitness calls instead of generations. Because humans cannot work for a long time, so the maximum number of fitness evaluations is set to a value which would be acceptable for a human.

We test each evaluation function with 30 trial runs in 3 different dimensional spaces, and apply the Wilcoxon signed-rank test on the fitness values at the maximum number of fitness calculations, MAX_{NFC}, to check the significance between the proposal IDE and simulated IDE. The results of the statistical test are shown in Table 6.2. Additionally, Fig. 4.11 gives the average convergence curves of the proposal and simulated IDE in 10-dimensional space.

Table 4.9: DE algorithm parameter setting.

population size for all search space	20
scale factor F	0.8
crossover rate	0.9
DE operations	DE/rand/1/bin
max. # of fitness evaluations, MAX_{NFC}, for all search space	500
dimensions, D	2, 5, and 10
# of trial runs	30
search ranges of all variables	[-6,6]

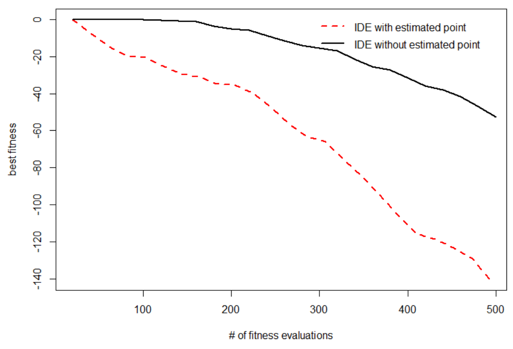
Table 4.10: Statistical test result of the Wilcoxon signed-rank test for average fitness values of 30 trial runs. $A \gg B$ means that A is significantly better than B with significance levels of 1%. $A \approx B$ means that there is no significant difference between A and B. A and B in the table represent that A: simulated IDE with an estimated convergence point, B: simulated IDE without an estimated convergence point.

	2-D	5-D	10-D
$GaussianFunction(n = 1)$	$A \gg B$	$A \gg B$	$A \gg B$
$GaussianFunction(n = 2)$	$A \gg B$	$A \gg B$	$A \gg B$
$GaussianFunction(n = 3)$	$A \approx B$	$A \gg B$	$A \gg B$
$GaussianFunction(n = 4)$	$A \approx B$	$A \gg B$	$A \gg B$

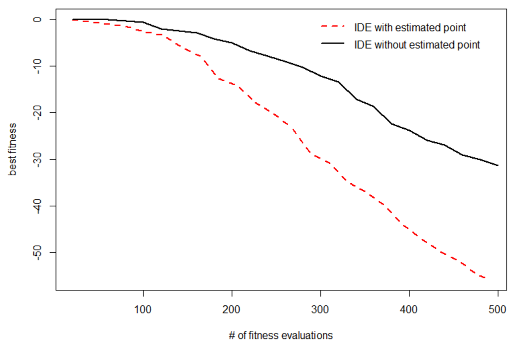
4.3.4 Discussions

We begin our discussion with an explanation of the superiority of our proposal. The estimated convergence point has a high probability to be close to the real optima and it is thus easier to find a better candidate in the next generation. Through introducing the point into the population, we hope it is possible to find an acceptable candidate quickly so that the number of user evaluations are reduced and user fatigue can also be alleviated. Although the estimated point will require an additional evaluation, the performance improvement is huge. So it is acceptable from the point of view of cost-performance.

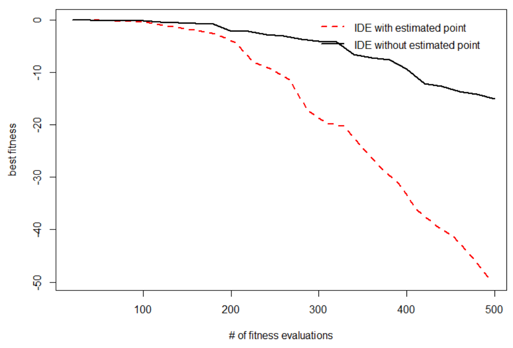
From the result of statistical test, we can see that the proposal can accelerate simulated IEC for almost all cases. In the simulation experiments, we use multiple Gaussian functions combined to form a multimodal problem. It is surely effective for unimodal tasks, but it is not always true for multimodal tasks because vectors go towards different local optima. So, the accuracy of the estimated point is not very accurate in the early stages of exploration for multimodal optimization. However, as the population gradually converges to the optimal area, it can be similar to a unimodal optimization in the later stages of exploration, and the accuracy of the estimated point is also getting higher and higher. This can be confirmed from the average convergence curves shown in Fig. 4.11. To improve the accuracy of the estimated point for multimodal optimization is one of the topics up for elaboration



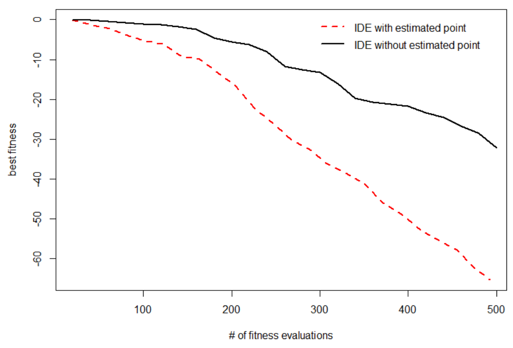
(a) $GaussianFunction(n = 1)$



(b) $GaussianFunction(n = 2)$



(c) $GaussianFunction(n = 3)$



(d) $GaussianFunction(n = 4)$

Figure 4.11: The average convergence curve of 30 trial runs of two methods in 10-D.

in future research, such as by clustering according to each local optimum area.

Next, we want to discuss the parameter settings in IEC applications. Due to the limitations of the human factor, the population size and evaluation levels cannot be set to large values. According to our experience, however, larger values are beneficial when it comes to constructing more vectors to estimate the convergence point. Therefore, we will pay more attention to the study of these parameters to achieve a better performance for accelerating IEC in our next study. Specifically, issues which need to be addressed include: the relationship of the dimension and the number of vectors; and how to construct a reasonable vectors using a limited number of individuals.

4.4 Chapter Summary

The basic estimation method is for unimodal optimization tasks and may not work well for optimization tasks with multiple local optima. Here, the reliability of the convergence point calculated from all moving vectors aiming at different local optima becomes low. To extend the estimation method's applicability to multimodal tasks, we propose a separation method to cluster moving vectors regarding their directions towards to their local optima. The experimental evaluations showed that using the estimated convergence point is beneficial for accelerating multimodal optimization from the cost-performance point of view. We try to introduce the estimated convergence point into a simulated IEC to accelerate the convergence and reduce user fatigue, at the cost of increasing by one the number of evaluations per generation. The experimental results show that it is possible to use the estimation method for real IEC applications.

We extend the basic estimation method to EMO algorithms to find promising non-dominated solution areas in search spaces using the estimation information from the objective spaces. We found that our proposed method can enhance EMO search in some benchmark problems, especially for the high dimensional and complex multi-objective problems which can obtain a greater number of Pareto solutions. We also analysed the modality of the Pareto improvement in both an objective space and a parameter space. We found that the Pareto improvement in the objective space demonstrates a unimodal characteristic, however demonstrates a multi-modal one in parameter space. In our next work, we look forward to applying our proposal to practical applications and analyzing those situations in which it can be used. Additionally, the parameter settings will also be further studied.

Chapter 5

Accelerating Fireworks Algorithm Using an Estimated Convergence Point

5.1 Synthetic Firework

We introduce a new kind of firework, named the synthetic firework [197], to make full use of the many generated sparks, which are otherwise only involved in the selection operation and then destroyed. The synthetic fireworks and fireworks of the current generation form many moving vectors which can be used to estimate a convergence point that is expected to locate near the global optimum. The estimated point is regarded as an elite individual and replaces the worst individual from the next generation if its fitness is better.

The method for calculating the synthetic fireworks is as follows. Each firework and its generated sparks form a subgroup, and we can construct many vectors between the firework and its generated sparks. If the firework is worse than one of the generated sparks, this vector's direction is considered to be promising. Otherwise, its antipode is used to calculate a synthetic firework. There are many methods to evaluate the potential of these directions. In this paper, we simply use the fitness difference between the endpoint and the start point of a vector to evaluate it. Thus, the larger the fitness difference is, the higher will be the weight of the vector. In order to not increase the number of fitness evaluations, we only calculate the antipode for a firework which is lacking a fitness evaluation if the antipodal direction is to be used. The fitness difference of the original vector is roughly used to evaluate the used antipodal direction. Finally, a synthetic firework can be roughly calculated by weighting those vectors with Eq. 5.1 in each firework group. The Figure 5.1 illustrates how a synthetic firework is thus formed.

$$v^i = \sum_{j=1}^{m} \frac{f(x^i) - f(s^i_j)}{\sum_{i=1}^{m} ||(f(x^i) - f(s^i_j))||} * (s^i_j - x^i) + x^i \tag{5.1}$$

where x^i and s^i_j represent the i-th firework and its j-th generated spark or antipodal point. v^i is the i-th synthetic firework of the i-th firework group; m is the number of generated sparks of the i-th firework; $f()$ is a fitness function.

We can obtain new synthetic fireworks up to the population of the the current firework generation. Since we do not increase the number of fitness evaluations

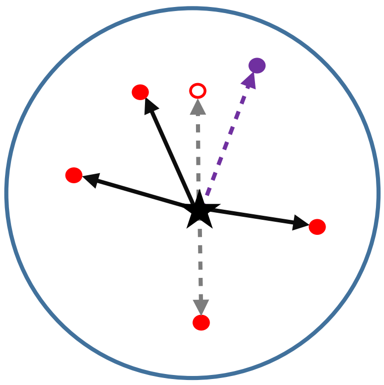


Figure 5.1: A synthetic firework is generated from a firework and its generated sparks. The black five-pointed star and the red solid points represent the firework and its generated sparks, respectively. The presence of a red hollow circle means that the antipode has been used. The purple solid point is the synthetic firework obtained by weighting these vectors.

and a new synthetic firework is expected to be better than the firework belonging to its subgroup, we will not evaluate the synthetic fireworks. A moving vector is calculated from the current firework to the newly generated synthetic firework in each subgroup, and the convergence point is estimated using these moving vectors with the basic estimation method. Algorithm 9 outlines the flow of EFWA using our proposed strategy.

Note that our proposed strategy does not change the structure of the original FWA when it is combined with other fireworks algorithms. It simply uses the fireworks and the generated sparks to build local gradient information, then uses this to estimate a convergence point to accelerate convergence.

5.2 Experimental Evaluations

We use 20 benchmark functions from the CEC2013 benchmark test suite [91] in our evaluations, which is designed for real parameter single-objective optimization. These landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal. We test them with 3 dimensional settings: $D = 2$, 10 and 30. We select EFWA as our test baseline and combine it with our proposal for this experiment using parameters as described in table 5.1.

For fair evaluations, we evaluate convergence against the number of fitness calls rather than generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces. We apply the Wilcoxon signed-rank test on the fitness values at the stop condition, i.e. the maximum number of fitness calculations, and compare EFWA with (EFWA + our proposed method). Table 5.2 shows the result of these statistical tests.

Algorithm 9 The framework for the fireworks algorithm using our proposed strategy. Steps 11 to 16 are from our proposal.

```
 1: Initialize $n$ fireworks randomly.
 2: Evaluate the fitness of each firework.
 3: while the termination condition is not satisfied do
 4:     Generate explosion sparks around each firework.
 5:     Use Gaussian mutation to obtain Gauss sparks.
 6:     if sparks are generated outside the search area then
 7:         Use a mapping rule to bring them back into the area.
 8:     end if
 9:     Evaluate the fitness of each generated spark.
10:     Select $n$ fireworks for the next generation from the generated sparks and the
        current fireworks.
11:     Calculate the synthetic fireworks for each subgroup.
12:     Obtain moving vectors using the synthetic fireworks and the current fireworks.
13:     Estimate a convergence point.
14:     if the estimated convergence point is better than the worst firework in the
        next generation then
15:         Replace the worst firework with the estimated point.
16:     end if
17: end while
18: end of program.
```

Table 5.1: Parameter setting of EFWA.

Parameters	Values
# of fireworks for 2-D, 10-D and 30-D search	5
# of sparks m	50
# of Gauss mutation sparks,	5
constant parameters	$a = 0.04$ $b = 0.8$
Maximum amplitude A_{max}	40
stop condition; MAX_{NFC}, for 2-D, 10-D, and 30-D search	1,000, 10,000, 40,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

5.3 Discussions

Most fireworks algorithm variants mainly use their computational resources for generating sparks, but the information from these sparks is not fully used. In our experimental evaluations, the total number of generated sparks was 10 times of that of the fireworks. It is clearly productive to consider how these many sparks can be used efficiently. We introduced a new type of firework, called the synthetic firework, to explore local gradient information on the fitness landscape. Thanks to the use of multiple vectors in each subgroup, the synthetic firework also has a certain anti-noise property, as its calculation cancels noise from the directions of the various moving vectors. This can help to improve the precision of the estimated convergence point.

Table 5.2: Statistical test results of the Wilcoxon signed-rank test for average fitness values of 30 trial runs of the proposal (EFWA + our proposed method) and conventional method (EFWA) at the stop condition, MAX_{NFC}. $A \gg B$ and $A > B$ mean that A is significant better than B with significant levels of 1% and 5%, respectively. $A \approx B$ means that although A is better than B, there is no significant difference between them.

Func.	2-D	10-D	30-D
f_1	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_2	proposal $\approx$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_3	proposal $\approx$ EFWA	proposal $>$ EFWA	proposal $>$ EFWA
f_4	EFWA $\approx$ proposal	proposal $\approx$ EFWA	proposal $\approx$ EFWA
f_5	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_6	proposal $\approx$ EFWA	EFWA $\approx$ proposal	proposal $\approx$ EFWA
f_7	proposal $>$ EFWA	EFWA $\approx$ proposal	proposal $\approx$ EFWA
f_8	proposal $\approx$ EFWA	EFWA $\approx$ proposal	proposal $\approx$ EFWA
f_9	EFWA $\approx$ proposal	EFWA $\approx$ proposal	EFWA $\approx$ proposal
f_{10}	proposal $>$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{11}	proposal $\approx$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{12}	EFWA $\approx$ proposal	proposal $\approx$ EFWA	proposal $\approx$ EFWA
f_{13}	proposal $\approx$ EFWA	proposal $\approx$ EFWA	proposal $\approx$ EFWA
f_{14}	proposal $\approx$ EFWA	proposal $>$ EFWA	proposal $\gg$ EFWA
f_{15}	proposal $\approx$ EFWA	proposal $\approx$ EFWA	EFWA $\approx$ proposal
f_{16}	EFWA $\approx$ proposal	proposal $\approx$ EFWA	proposal $\approx$ EFWA
f_{17}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{18}	proposal $>$ EFWA	proposal $\approx$ EFWA	EFWA $\approx$ proposal
f_{19}	proposal $\approx$ EFWA	proposal $\approx$ EFWA	proposal $\gg$ EFWA
f_{20}	proposal $\approx$ EFWA	proposal $\approx$ EFWA	EFWA $\approx$ proposal

In any case, the proposed method increases each generation's fitness calculations by only one - so we can say that it is a low risk, high return strategy.

What potential still remains for our proposed firework, the synthetic firework? Although we have used only the fitness difference between the two endpoints of a moving vector to evaluate it, we think that not only these fitness differences but also their lengths should be considered to understand the local gradient information more accurately, yielding further improvements in the estimate. Additionally, there are many other ways to weight moving vectors and increase the precision of the estimated convergence point. As an example, the fitness value at the beginning point or the end point of a moving vector can be used to evaluate it, which means that the lower the distance from the optimal area, the higher the weight given. A precise way of obtaining reasonable weights for the vectors is also a potential discussion topic.

We would like to point out that the new type of firework introduced can be used to speed up convergence. In this paper, we used synthetic fireworks to estimate a convergence point without evaluating their fitness. They have the potential to act as a new guide for individuals, helping move them toward a preferable evolutionary direction rather than random exploration. The new synthetic fireworks can be introduced into a population to improve the diversity and reduce selection pressure. How to use them reasonably is also a potential discussion topic.

We also performed an extra experiment to investigate the fitness of synthetic fireworks. We compared the synthetic firework with the firework individual belonging to its same subgroup. The experimental results show that in the early stages, synthetic fireworks are better than fireworks individuals, while the probability of a better synthetic firework decreases as the convergence progresses. For optimization problems with different characteristics, it seems reasonable to use a different method for assigning weights when creating the synthetic fireworks. Perhaps different optimization stages could use different weighting methods to obtain better synthetic fireworks. Summarizing the relationship between weighting method and optimization problem is thus also a potential topic for study.

From the results of the statistical tests, we find that the proposed method is beneficial for unimodal optimization problems ($f_1 - f_5$), while the performance on low-dimensional multimodal optimization problems is not obvious. This may be because the basic estimation method, which is clearly effective for unimodal optimization problems, is not always valid for multimodal problems where the moving vectors go toward different local optima. Further, the number of moving vectors is small (in this case, the number is 5), and even on some multimodal optimization problems, it is less than the number of peaks. Regardless, the proposed strategy does not show any deleterious effect. For the next stage, using past searching individuals to increase the number of moving vectors, and combining it with the clustering method may allow us to extend our proposal to multimodal optimization problems.

5.4 Chapter Summary

We propose a new kind of fireworks which uses the generated sparks to efficiently estimate a convergence point which can act as an elite individual to accelerate the fireworks algorithm. The controlled experiments confirm that the proposed strategy can significantly improve the performance of conventional EFWA, and the higher the dimension, the more obvious the effect.

In future work, we will further study the proposed synthetic fireworks and use them to beneficially guide the evolution of the population. Additionally, it is suggested that we can further improve the accuracy of the estimated point by using historical information to better understand the fitness landscape.

Chapter 6

Search Strategies for Fireworks Algorithm

6.1 Accelerating FWA with New Strategies

6.1.1 Amplitude Reduction Strategy

This strategy [191] is to decrease the amplitude sizes of all fireworks from one generation to the next regardless of their fitness. We use the formula of Eq.(6.1) to determine the amplitude of fireworks. The Figure 6.1 shows how the amplitude changes throughout the exploration period.

$$A_i = \begin{cases} A_{init} * (1 - \frac{FE_{cur}}{FE_{max}}) & if \quad FE_{cur} < c * FE_{max} \\ A_{init} * (1 - c) & Others \end{cases} \tag{6.1}$$

where, A_{init} is the initial maximum amplitude of fireworks; FE_{cur} and FE_{max} represent the current and maximum number of fitness evaluations, respectively; and c is a constant for preventing the amplitude from becoming too small.

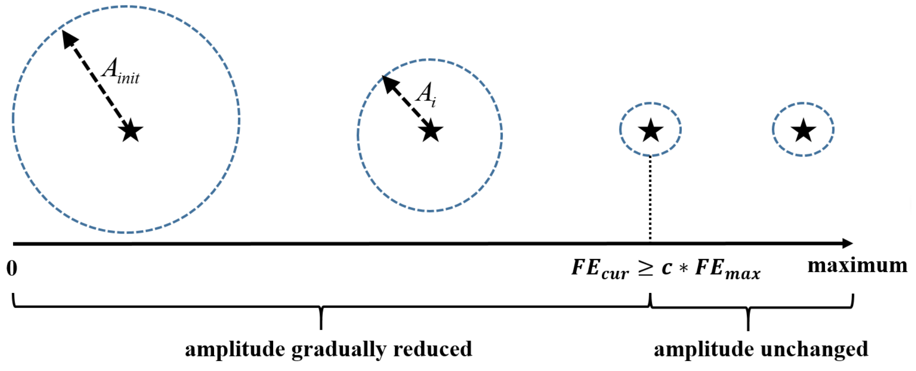


Figure 6.1: Changes in amplitude throughout the exploration period.

6.1.2 Local Optima-based Selection Strategy

This strategy [191] is to use a local optima-based selection of fireworks in the next generation instead of the distance-based selection used in the original FWA. Since the generated sparks can be considered as a local search around each firework, a set of a firework and its generated sparks can be considered a local subgroup. Then, we can obtain n local subgroups and a mutation subgroup consisting of all mutated sparks as the $(n+1)$-th subgroup. Because n new fireworks should be selected in the next generation, we merge the mutation subgroup and the subgroup of the worst firework into a new subgroup. The proposed local optima-based selection strategy takes the best firework or spark from each subgroup to form the next generation. The Figure 6.2 demonstrates this selection strategy.

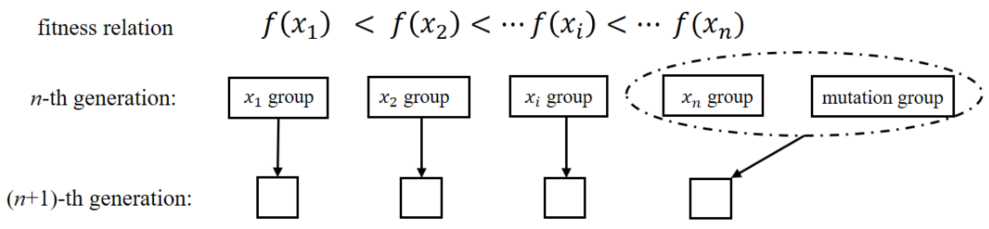


Figure 6.2: The best one in each subgroup will be selected and go to next generation.

6.1.3 Experimental Evaluations

We use 20 benchmark functions from the CEC2013 benchmark test suite [91] in our evaluations. These landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal. We test them with 3 dimensional settings: $D =$ 2, 10 and 30.

To analyze the effect of each proposed improvement, we design the following four experiments; Experiments 1, 2, 3, and 4 are, respectively, the original FWA, the original FWA + the first proposed strategy (amplitude decrease strategy), the original FWA + the second proposed strategy (selection method of the firework in the next generation), and the original FWA + both strategies. Table 6.1 shows the parameter settings of the canonical FWA. The parameter settings for Experiments 2-4 are the same as the canonical FWA except the initial amplitudes; initial amplitudes A_{init} and constant c of the Eq. (6.1) are set as 10 and 0.95, respectively.

Table 6.1: Parameter setting of original FWA.

Paramaters	Values
# of fireworks for 2-D, 10-D and 30-D search	5
# of sparks m	50
# of Gauss mutation sparks,	5
constant parameters	$a = 0.04$ $b = 0.8$
Maximum amplitude A_{max}	40
stop condition; MAX_{NFC}, for 2-D, 10-D, and 30-D search	4,000, 40,000, 100,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

We evaluate convergence along the number of fitness calls instead of generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces. We apply the Friedman test and Holm's multiple comparison to the fitness values at the stop condition, i.e. maximum number of fitness calculations, for each benchmark function to check for significant difference among the methods. Table 6.2 shows the result of these statistical tests.

Table 6.2: Statistical test result of the Friedman test and Holm's multiple comparison for average fitness values of 30 trial runs of 4 methods. $A \gg B$ and $A > B$ mean that A is significant better than B with significant levels of 1% and 5%, respectively. $A \approx B$ means that there is no significant difference between A and B. Numbers in the table represent that 1: original FWA, 2: original FWA + proposed strategy 1, 3: original FWA + proposed strategy 2, and 4: original FWA + proposed strategies 1 and 2.

f_1	2-D	10-D	30-D
f_1	$4 \approx 2 \gg 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$
f_2	$4 \approx 3 \approx 2 > 1$	$4 > 2 \gg 3 \approx 1$	$4 > 2 \gg 3 \gg 1$
f_3	$3 \approx 2 \gg 1 \gg 4$	$4 \approx 2 \approx 3 \gg 1$	$4 \approx 2 > 3 \gg 1$
f_4	$3 \approx 2 \approx 4 \approx 1$	$2 \gg 1 \approx 4 \approx 3$	$2 > 4 \gg 3 > 1$
f_5	$4 \approx 2 \gg 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$
f_6	$4 \approx 2 \approx 3 \approx 1$	$4 > 3 \approx 2 \gg 1$	$2 \approx 3 \approx 4 \gg 1$
f_7	$3 \gg 2 \approx 4 \gg 1$	$1 \approx 2 \approx 4 \approx 3$	$4 \approx 1 \approx 3 \approx 2$
f_8	$4 > 3 \approx 1 > 2$	$4 \approx 3 \approx 2 \gg 1$	$4 \approx 2 > 1 \approx 3$
f_9	$4 \gg 3 \gg 2 \gg 1$	$3 \approx 4 \approx 1 \approx 2$	$3 \approx 4 \approx 1 \approx 2$
f_{10}	$4 \approx 2 > 3 \gg 1$	$4 > 2 \gg 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$
f_{11}	$3 \approx 4 \approx 2 \gg 1$	$3 \gg 1 \approx 4 \gg 2$	$3 \approx 1 \approx 4 \gg 2$
f_{12}	$3 \approx 2 \approx 4 \approx 1$	$1 \approx 3 \approx 2 \approx 4$	$1 \approx 4 \approx 3 \approx 2$
f_{13}	$3 \approx 2 \approx 1 \approx 4$	$3 \approx 1 \approx 2 \gg 4$	$1 \approx 3 \approx 2 \approx 4$
f_{14}	$3 \approx 1 \approx 2 \approx 4$	$3 \gg 1 \approx 4 \gg 2$	$3 > 4 \approx 1 \gg 2$
f_{15}	$1 \approx 3 \approx 2 \approx 4$	$4 \approx 2 \approx 3 \approx 1$	$4 \approx 2 \approx 3 \gg 1$
f_{16}	$4 \approx 2 \gg 3 \approx 1$	$4 \approx 2 \gg 3 > 1$	$2 \approx 4 \gg 3 \gg 1$
f_{17}	$4 \approx 2 \approx 3 \approx 1$	$4 \gg 2 \gg 3 \gg 1$	$4 \gg 2 > 3 \gg 1$
f_{18}	$4 \approx 2 \approx 3 \approx 1$	$2 \approx 4 \approx 1 \approx 3$	$2 > 4 \gg 1 \approx 3$
f_{19}	$2 \approx 4 \approx 3 \gg 1$	$4 > 2 \approx 3 \gg 1$	$4 \gg 2 \gg 3 \gg 1$
f_{20}	$2 \approx 4 \approx 3 \gg 1$	$3 \approx 1 \approx 4 \approx 2$	$1 \approx 3 \approx 2 \approx 4$

6.1.4 Discussions

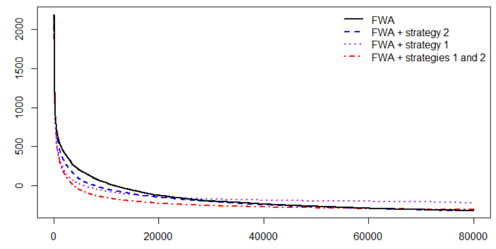
We begin our discussion with an explanation of the superiority of our proposed strategies. In the original FWA, better fireworks can obtain more resources within a small range, thus undertaking responsibility for exploitation. Exploration is achieved by worse fireworks obtaining less resources in a larger range through the whole search period. However, exploration should be a task performed primarily in the early stages of search, while exploitation should be gradually emphasized along with the convergence of the population. So the first proposed strategy uses a decrement strategy to make all fireworks responsible for exploration in the early generations, with this exploration ability becoming gradually weaker as the exploitation ability becomes gradually stronger to achieve a good balance between exploration and exploitation.

We simply use the number of fitness evaluations to control the amplitude of fireworks in this paper, but this is not the unique realization of the proposed strategy

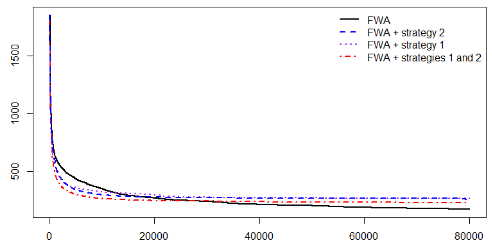
1; there must be other realizations which would allow us to improve its performance even more. For example, the amplitude can be adjusted adaptively according to optimization tasks, not just based on the number of fitness evaluations.

The distance-based selection used in the original FWA aims to preserve the diversity of fireworks, but there are still some shortcomings. This selection strategy gives higher selection probabilities to individuals located far away from other individuals. However, there is no guarantee that the fireworks selected by this original strategy have better fitness in the next generation than those in the current generation except the best individual. Further, there is also no guarantee that individuals coming from each subgroup will be selected fireworks in the next generation. If no individual from a certain subgroup is selected in the next generation, the area will not be explored in the next generation and the diversity may be lost.

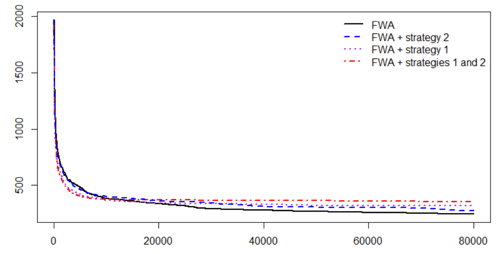
The second proposed strategy can overcome these shortcomings and ensure each local optimum individual can remain in the next generation to maximize and the preserve the population diversity. This strategy may develop to become a new niche method for finding multi local or global optima at one time run.



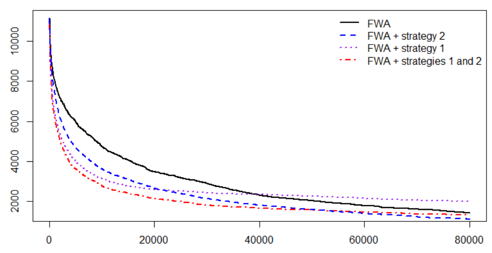
(a) f_{11}



(b) f_{12}



(c) f_{13}



(d) f_{14}

Figure 6.3: Convergence curves of the original FWA, the original FWA + proposed strategy 1, the original FWA + proposed strategy 2, and the original FWA + proposed strategies 1 and 2 for 30-D $f_{11} - f_{14}$, respectively.

Next, we discuss the effectiveness of our proposed strategies. To analyze their performances, Friedman test and Holm's multiple comparison test were applied at the stop condition in three different dimensions. The two strategies do not add additional fitness computation cost. Nevertheless, the statistical results in the Table 6.2 show that either of the two proposed strategies can improve the performance of the original FWA, and their combination can further improve performance in almost all evaluation cases.

Although combining two proposed strategies 1 and 2 with original FWA works well, it did not show clear performance for f_{11} - f_{14} in the Table 6.2. Fig. 6.3 shows the average convergence curves of 4 methods for these 30-dimensional benchmark functions. These improved strategies for Rastrigin's function and Schwefel's function showed better performance in the early searching stages, while it could not keep their better performance in the later period and even became worse than the original FWA. It may be due to their many local optima; the local optima-based selection can maximize the diversity of the population, but it may reduce the convergence speed. We need further analysis of this result to understand the real reason and develop its solution.

6.2 Scouting Strategy for Promising Directions

6.2.1 Scouting Strategy

The inspiration of this strategy comes from different explosions of fireworks in the real world. The fireworks algorithm groups focus on generating sparks for each firework in an adaptive fixed range so far. In fact, there are a variety of ways to explode, such as continuous explosion or directional explosion etc. As a first attempt, we introduce a new explosion to generate sparks, named scouting strategy [198]. And Fig. 6.4 demonstrates the gradual generation of sparks with new explosion mechanism.

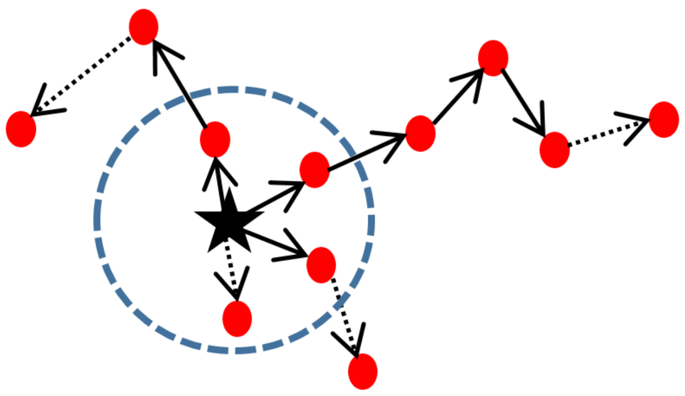


Figure 6.4: Scouting strategy to generate sparks. Black five-pointed star means a firewirk and red solid points represent generated sparks. Solid arrows indicate that current direction is good and will continue to explode, while dotted arrows indicate the end of the explosion in current direction.

Canonical FWA (1) determines the number of spark individuals and their searching area size around each firework individual, (2) generates sparks randomly in the area at once, and (3) generates a mutant spark individual(s), and (4) determines new fireworks in the next generation using information of each firework and its sparks. Since sparks are generated at once in the step (2), their information cannot be fully used.

To improve this point, our scouting strategy generates spark individuals one by one by checking their fitness. This trace looks like a scout soldier finding a better direction. When the n-th spark individual become worse than the $(n-1)$-th spark, the continuous scouting stops and new scouting starts from the initial point, the coordinates of their firework. Since the number of sparks continuous generated or the scouting depth is an index of better local search direction, we can use it that cannot be obtain in the step (2) of the canonical FWA. Due to this mechanism, a search area size of spark individuals is not limited to the fixed size in the step (1) but can extend to a better direction using the same number of sparks.

6.2.2 Filtering Strategy

This is to strengthen the competition among individuals. All spark individuals of conventional FWA have an opportunity to be selected as individuals in the next generation regardless their superiority to their parents, i.e. firework individuals. It may lead to repeatedly explore in poor areas and result in waste of searching resources.

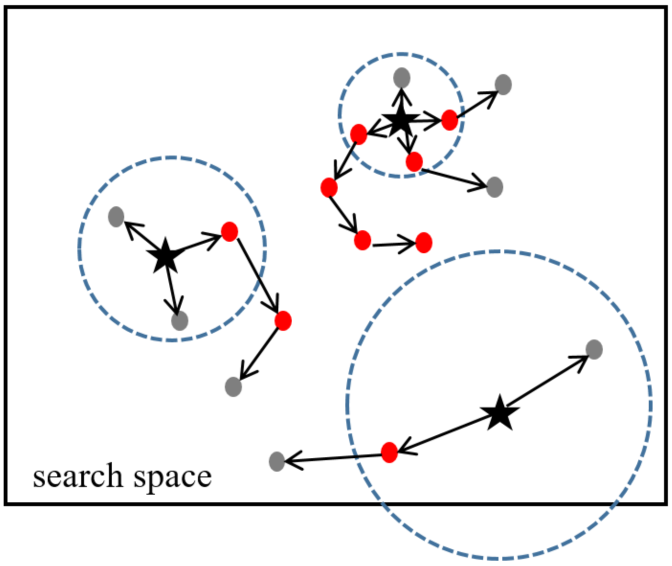


Figure 6.5: Filtering worse generated sparks. Black five-pointed star means firewirks and solid points represent generated sparks, where gray points indicate the sparks to be filtered and red points will be retained. The arrows indicate the direction of the tracking explosion.

Filtering strategy uses the fitness of a firework individual as a criterion for filtering its poor sparks and keeping its good sparks. After then, reserved sparks, mutation sparks and current fireworks are mixed together and used to be selected for the next generation. This strategy can increase the probability of searching in potential areas and accelerate FWA evolution to promising directions. Fig. 6.5 demonstrates this filtering strategy.

6.2.3 Experimental Evaluations

We use 28 benchmark functions from the CEC2013 benchmark test suite [91] in our evaluation experiments. These landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal. We test each benchmark function with 3 dimensional settings: $D = 2$, 10 and 30, respectively. The EFWA experimental parameters are set as in Table 6.3.

Table 6.3: EFWA algorithm parameter setting.

# of fireworks for 2-D, 10-D, and 30-D search	5
# of total sparks m	60
# of Gaussian mutation sparks	5
Maximum amplitude A_{max}	40
constant parameters	$a = 0.04$ $b = 0.8$
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1,000, 10,000, 40,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

For fair evaluations, we evaluate convergence along the number of fitness calls instead of generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces and apply the Wilcoxon signed-rank test on the fitness values at the maximal number of fitness calculations, MAX_{NFC}, to check the significance of the proposal and conventional EFWA. Tables 6.4 show the statistical test result.

6.2.4 Discussions

We propose a scouting strategy to track the current promising directions to dig deeper potential areas instead of randomly exploring within a fixed range assigned to each firework individual, which means that the better area is, the deeper and more careful exploitation is conducted. The proposals do not change the number of resources allocated to each firework individual, i.e. there is no extra fitness consumptions. Generated sparks also have an opportunity to produce new sparks to increase population diversity. Although we simply used single point tracking to explore the local information as our first attempt, we also can use multi-point tracking, generating multiple points rather than a point, to reduce the risk of single point tracking, i.e. falling into local optima.

The main contribution of the second strategy is to increase the probability of exploring potential areas and avoid ineffective searches. In the conventional EFWA, even though some sparks are worse than their parent (firework), they have a chance to remain in the next generation. If such sparks are selected, they may search in less potential areas, waste resources, which results slow convergence. The second strategy filters poor sparks and ensures to make the whole population evolve toward the optima areas.

Table 6.4: Wilcoxon signed-rank test results for EFWA vs. Proposal (EFWA + proposed strategies) at the stop condition. $\gg$ and $>$ mean that the left is better than the right with significance levels of 1%, 5% respectively, and $\approx$ means there is no significance.

Func.	2-D	10-D	30-D
F_1	Proposal $\gg$ EFWA	EFWA $>$ Proposal	EFWA $\gg$ Proposal
F_2	Proposal $\approx$ EFWA	Proposal $>$ EFWA	Proposal $\gg$ EFWA
F_3	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_4	Proposal $>$ EFWA	Proposal $>$ EFWA	Proposal $>$ EFWA
F_5	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA	Proposal $\approx$ EFWA
F_6	Proposal $\approx$ EFWA	Proposal $>$ EFWA	Proposal $\gg$ EFWA
F_7	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_8	EFWA $\approx$ Proposal	Proposal $\approx$ EFWA	Proposal $\approx$ EFWA
F_9	Proposal $>$ EFWA	Proposal $\gg$ EFWA	Proposal $\approx$ EFWA
F_{10}	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA	Proposal $\approx$ EFWA
F_{11}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{12}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{13}	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{14}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{15}	Proposal $\approx$ EFWA	EFWA $\approx$ Proposal	EFWA $\approx$ Proposal
F_{16}	Proposal $\approx$ EFWA	EFWA $\gg$ Proposal	EFWA $\gg$ Proposal
F_{17}	Proposal $>$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{18}	Proposal $>$ EFWA	Proposal $>$ EFWA	Proposal $>$ EFWA
F_{19}	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{20}	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	EFWA $\gg$ Proposal
F_{21}	EFWA $\approx$ Proposal	EFWA $\approx$ Proposal	Proposal $\approx$ EFWA
F_{22}	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{23}	Proposal $\approx$ EFWA	Proposal $\approx$ EFWA	EFWA $\approx$ Proposal
F_{24}	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{25}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{26}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA
F_{27}	EFWA $\approx$ Proposal	Proposal $\gg$ EFWA	Proposal $>$ EFWA
F_{28}	Proposal $\approx$ EFWA	Proposal $\gg$ EFWA	Proposal $\gg$ EFWA

Applicability of our proposed strategies is also a feature. They are appicable to not only EFWA used in this paper but also any other variants of FWA. Furthermore, these two proposals can be combined with various EC algorithm individually or in combination.

Finally, to analyze their performances, Wilcoxon signed-rank test was applied to fitness values at the stop conditions and checked significance between EFWA and (EFWA + proposed strategies). The test results in Table 3.2 shows that the proposed strategies can improve the performance of the conventional EFWA significantly in most cases, and the proposed strategies resulted faster convergence. This effect was obvious for complex task, i.e. higher dimensional tasks. It may be because the total number of sparks is not sufficient for the complex high-dimensional problems and using tracking explosions instead of explosion in a fixed range can learn more quickly about local fitness information.

6.3 Multi-layer Explosion-Based FWA

6.3.1 Multi-layer Explosion Strategy

With the development of production techniques, various exquisite fireworks explosion shapes can be customized, commonly there, heart-shaped explosion, multi-layer explosion and specific area explosion, etc. Inspired by various explosion ways and shapes, we firstly introduce a different explosion model, multi-layer explosion [195] to enhance the use of the local fitness landscape, while conventional FWA generates spark individuals around a firework individual at once. Figure 6.6 illustrates the generation process of sparks using our proposed strategy. As our first attempt, we set the number of layers to 2 in this disseration thought. It can be set to any positive integer theoretically.

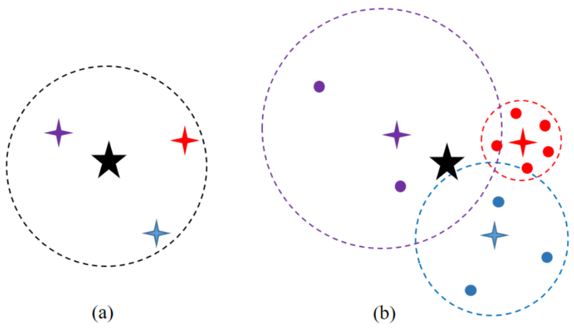


Figure 6.6: The general framework of our proposed multi-layers explosion strategy, where a two-layer explosion is shown as an example. (a) and (b) are the first and the second explosions, respectively, where black pentagrams represent firework individuals, four-pointed stars and solid dots of various colors represent generated sparks in the first layer and the second layer, respectively. A dashed circle represents the search radius of a firework and spark individuals.

First Layer Explosion

The explosion of a firework is determined by two factors, *number of sparks* and *amplitude of explosion*. They individually determine how many spark individuals can be generated by a firework and its search radius (amplitude). In the first layer, each firework determines its search radius adaptively allocated according to its fitness, which is the same as conventional FWA.

However, we treat each firework equally in the first layer, and all fireworks generate the same number of a few number of spark individuals to investigate its surrounding fitness landscape within its own search radius. These generated spark individuals in the first layer are evaluated with an objective function and used to determine the explosive shape of the second layer and the allocation of spark individuals in the second layer. The number of generated spark individuals for each firework in the first layer is decided adaptively according to its fitness.

Subsequent Layer Explosion

Let's define symbols temporally to explain the main idea of this paper, multi-layer explosions. N is the number of firework individuals; f_i is the i-th firework individual; m is the total number of sparks; l is the maximum number of explosion layers; m_i is the total number of sparks in all layer explosions under the f_i; $m_i^{(k)}$ is the number of sparks in the k-th explosion layer; $s_{i,j}^{(k)}$ is the j-th spark individual ($j = 1 \sim m_i^{(k)}$) generated in the k-th layer explosion under the f_i. Relationships among them are $m = \sum_{i=1}^{N} m_i$ and $m_i = \sum_{k=1}^{l} m_i^{(k)}$.

The i-th subgroup is formed by the i-th firework individuals, f_i, and all its sparks $s_{i,j}^{(k)}$; see the Figure 6.6. Since search parameters in each explosion layer are independently decided in each subgroup, we explain the multi-explosion process at the i-th subgroup in the below.

The first key problem is how to distribute the remaining number of sparks, $(m - N * m_i^{(1)})$, to the remained explosion layers. The total number of sparks in subsequent explosion layers is $m_i - m_i^{(1)} = \sum_{k=2}^{l} m_i^{(k)}$. We simply distribute an equal number of sparks to each layer, $(m_i - m_i^{(1)})/(l-1)$. After all firework individuals complete their first explosions, their second explosions are triggered by not the firework individuals but the their generated spark individuals, $s_{i,j}^{(1)}$.

The second key problem is how to decide the search radius around $s_{i,j}^{(k)}$ and how to divide $(m_i - m_i)/(l-1)$ explosion sparks to each $s_{i,j}^{(k)}$ in each layer. These two parameters are decided by the fitness of the $s_{i,j}^{(k)}$ adaptively. This layer explosion is repeated until explosions repeat l times.

Algorithm 10 The general framework of the proposed multi-layer explosion strategy. See the definition of symbols in the sub-Section *Subsequent Layer Explosion.*

```
1: for $i = 0; i < n; i++$ do
2:   Decide the number of generated sparks, $m_i^{(1)}$, in the first layer for each firework.
3:   Decide a search radius around the $i$-th firework according to its fitness.
4:   Conduct the first layer explosion for each firework.
5:   while The number of explosions does not reach a pre-defined maximum layer do
6:     for $j = 0; j < m_i^{(k)}; j++$ do
7:       Decide the number of sparks generated by the $j$-th spark in the previous layer.
8:       Decide a search radius of around the $j$-th spark in the previous layer.
9:     end for
10:    Generate the next explosion sparks for each spark in the previous layer.
11:  end while
12: end for
13: end of explosion.
```

Our proposed strategy divides sparks in multi-layer explosion, and sequential explosions expand searching areas to better directions gradually layer by layer according to the fitness of sparks in each layer explosion, while the fitness of a firework individual decides all sparks at once. It is important to note that our proposed explosion strategy just change the layer of the explosions without changing any other operations, i.e. generation of spark individuals and mutation operation. Algorithm 16 shows the flow of our proposed explosion strategy. When it is combined with other versions of the FWA, only their corresponding explosion operation is replaced.

6.3.2 Experimental Evaluations

We use 28 functions from the CEC2013 test suite [91] to evaluate the performance of our proposal. The test suite is devoted to the approaches, algorithms, and techniques for solving real parameter single objective optimization.

EFWA which performance is better than original FWA as a baseline algorithm. We integrate our proposed explosion strategy into EFWA and compare it with original EFWA and several other state-of-the-art EC algorithms. The table 6.5 shows the parameter settings of EFWA used in our experiments and the table 6.6 shows the parameter settings of PSO used in our experiments.

Table 6.5: Parameter setting of EFWA.

Parameters	Values
# of fireworks for 2-D, 10-D and 30-D search	5
# of sparks m	60
# of Gauss mutation sparks,	5
constant parameters	$a = 0.04$ $b = 0.8$
Maximum amplitude A_{max}	40
stop condition; max. # of fitness evaluations, MAX_{NFC}, for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 6.6: PSO algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	70
inertia factor w	1
constant c_1 and c_2	1.49445, 1.49445
max. and min. speed V_{max} and V_{min}	2.0, −2.0
stop condition; max. # of fitness evaluations, MAX_{NFC}, for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

For fair evaluations, we evaluate convergence along the number of fitness calls rather than generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces, D = 2 (2-D), 10 (10-D), and 30 (30-D), respectively.

To evaluate the effectiveness of our proposed explosion strategy, we design two sets of control experiments. In the first experiment, we only compare the conventional EFWA and (EFWA + our proposed strategy), and the Wilcoxon signed-rank test at the maximum number of fitness calculations is used to test significant difference. In the second experiment, we compare (EFWA + our proposed strategy) with PSO and guided FWA that is one of the most competitive variation of FWA. The Kruskal-Wallis test and Holm's multiple comparison test are applied to check whether there is a difference among these three algorithms at the stop condition. Finally, the Tables 6.7 and 6.8 show the statistical test results of the first and second controlled experiments, respectively.

Note that we do not use a complete replication guided FWA published in [89] but instead integrate the concept of guiding spark into EFWA with the following two modifications: (1) We do not calculate a guided spark for all firework individuals but only the best firework individual, and (2) all sparks generated by the best firework are divided into two groups, better than the firework and worse than the firework. All sparks in both groups are involved in calculating guided spark rather than sparks in the front of the two groups. Our proposed strategy and other efficient strategies for FWA can be compared based on the same baseline algorithm, which can make the experiment fairer.

Besides, we set the number of generated spark individuals in the first layer to 3, 4 and 5 to investigate its influence on the performance, and other parameter settings is the same with the Table 6.1 completely. The Friedman test and Holm's multiple comparison test are applied to check significant difference among them at the stop condition. The Table 6.9 presents the statistical tests result of different parameter settings of generated sparks in the first layer.

6.3.3 Discussions

Analysis of compositions of our proposal

We begin our discussion on the superiority of our proposed explosion strategy. Original FWA and several its powerful variant versions, have paid little attention to the use of the fitness landscape information to generate spark individuals efficiently and reasonably. Besides, they only use a small amount of firework individuals to explore fitness landscape and generate a large number of spark individuals. Although it can achieve a fine local search around a firework individual, many generated spark individuals are used in only a selection operation and then destroyed. It results that many spark individuals are not fully used in fact, and generating a large number of sparks from only a few fireworks is risky. Thus, our proposal can overcome the above deficiencies without adding any additional fitness calculations. With the same number of fitness calculations, our multi-layer explosion strategy can explore and use the features of the local fitness landscape more effectively. In the first layer, each firework individual generates several spark individuals to further explore the local fitness landscape carefully, while conventional FWA only uses a firework individual to achieve it. The objective of the first explosion is to enhance the understanding of local fitness landscape with multiple tentative individuals (firework individuals and spark individuals in the first layer). Subsequently, the following layer explosion

Table 6.7: Statistical test results of the Wilcoxon signed-rank test for average fitness values of 30 trial runs of the proposal (EFWA + our proposed mechanism) and EFWA at the stop condition, MAX_{NFC}. $A \gg B$ and $A > B$ mean that A is significant better than B with significant levels of 1% and 5%, respectively. $A \approx B$ means that although A is better than B, there is no significant difference between them.

Func.	2-D	10-D	30-D
f_1	proposal $\gg$ EFWA	EFWA $\gg$ proposal	EFWA $\gg$ proposal
f_2	proposal $\approx$ EFWA	proposal $\approx$ EFWA	EFWA $>$ proposal
f_3	proposal $\approx$ EFWA	proposal $\gg$ EFWA	proposal $\approx$ EFWA
f_4	proposal $>$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_5	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\approx$ EFWA
f_6	proposal $\gg$ EFWA	proposal $\approx$ EFWA	proposal $\approx$ EFWA
f_7	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $>$ EFWA
f_8	proposal $\gg$ EFWA	proposal $\approx$ EFWA	proposal $>$ EFWA
f_9	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{10}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	EFWA $\gg$ proposal
f_{11}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{12}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{13}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{14}	proposal $\approx$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{15}	proposal $\approx$ EFWA	proposal $>$ EFWA	proposal $\approx$ EFWA
f_{16}	EFWA $\approx$ proposal	EFWA $\approx$ proposal	EFWA $\approx$ proposal
f_{17}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{18}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{19}	proposal $\gg$ EFWA	proposal $>$ EFWA	proposal $\gg$ EFWA
f_{20}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\approx$ EFWA
f_{21}	proposal $\approx$ EFWA	EFWA $\gg$ proposal	proposal $\approx$ EFWA
f_{22}	proposal $>$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{23}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $>$ EFWA
f_{24}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{25}	proposal $\gg$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{26}	proposal $\approx$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{27}	proposal $>$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA
f_{28}	proposal $>$ EFWA	proposal $\gg$ EFWA	proposal $\gg$ EFWA

is adaptively conducted based on the results of the previous explosion. Meanwhile, the center of the explosion is transferred from firework individuals to spark individuals generated in the previous layer, which can increase the diversity of generated spark individuals and achieve a more refined local search. Thus, the objective of following explosion is to be more reasonable to generate diverse and potential spark individuals based on the characteristic of local fitness landscape. As a summary, the

Table 6.8: Statistical test results of the Kruskal-Wallis test and Holm's multiple comparison test for average fitness values of 30 trial runs of the proposal (EFWA + our proposed mechanism), PSO and guided EFWA (GEFWA) at the stop condition, MAX_{NFC}. $A \gg B$, $A > B$ and $A \approx B$ represent the same meaning as the symbols in the Table 6.7.

Func.	2-D	10-D	30-D
f_1	proposal $\gg$ PSO $\gg$ GEFWA	GEFWA $\gg$ proposal $\gg$ PSO	GEFWA $\gg$ proposal $\gg$ PSO
f_2	proposal $\approx$ GEFWA $>$ PSO	PSO $\gg$ proposal $\approx$ GEFWA	PSO $\approx$ GEFWA $\gg$ proposal
f_3	proposal $\gg$ PSO $\approx$ GEFWA	PSO $\approx$ proposal $\approx$ GEFWA	PSO $\gg$ proposal $\approx$ GEFWA
f_4	proposal $\approx$ GEFWA $\gg$ PSO	PSO $\gg$ proposal $\gg$ GEFWA	proposal $\gg$ PSO $>$ GEFWA
f_5	proposal $\gg$ PSO $\approx$ GEFWA	PSO $\gg$ proposal $\gg$ GEFWA	proposal $\approx$ GEFWA $\gg$ PSO
f_6	PSO $\approx$ proposal $\gg$ GEFWA	PSO $\approx$ proposal $\approx$ GEFWA	PSO $\approx$ proposal $\approx$ GEFWA
f_7	proposal $>$ PSO $>$ GEFWA	PSO $\gg$ proposal $>$ GEFWA	PSO $\gg$ proposal $\gg$ GEFWA
f_8	proposal $\approx$ PSO $\approx$ GEFWA	proposal $\approx$ PSO $\approx$ GEFWA	proposal $\approx$ GEFWA $\approx$ PSO
f_9	proposal $\approx$ PSO $\approx$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA
f_{10}	proposal $\gg$ GEFWA $\approx$ PSO	PSO $\gg$ proposal $\gg$ GEFWA	GEFWA $\gg$ proposal $\gg$ PSO
f_{11}	proposal $\gg$ PSO $\approx$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA
f_{12}	proposal $\approx$ PSO $\approx$ GEFWA	PSO $\gg$ proposal $\gg$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA
f_{13}	proposal $\gg$ PSO $\approx$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA
f_{14}	PSO $\approx$ proposal $\approx$ GEFWA	proposal $\gg$ PSO $\approx$ GEFWA	proposal $\gg$ GEFWA $>$ PSO
f_{15}	PSO $\approx$ proposal $\approx$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA	proposal $>$ GEFWA $>$ PSO
f_{16}	GEFWA $>$ proposal $\approx$ PSO	GEFWA $\approx$ proposal $\gg$ PSO	GEFWA $\approx$ proposal $\gg$ PSO
f_{17}	proposal $\gg$ PSO $\approx$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA
f_{18}	proposal $\approx$ PSO $\approx$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA	proposal $\gg$ GEFWA $>$ PSO
f_{19}	proposal $\approx$ GEFWA $\approx$ PSO	proposal $\gg$ PSO $\approx$ GEFWA	PSO $\approx$ proposal $>$ GEFWA
f_{20}	proposal $\gg$ PSO $\approx$ GEFWA	PSO $\approx$ proposal $>$ GEFWA	proposal $\approx$ GEFWA $\gg$ PSO
f_{21}	PSO $\approx$ proposal $\approx$ GEFWA	GEFWA $\gg$ proposal $\gg$ PSO	proposal $\approx$ GEFWA $\gg$ PSO
f_{22}	PSO $\approx$ proposal $\gg$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA	proposal $\gg$ PSO $\approx$ GEFWA
f_{23}	PSO $\approx$ proposal $\approx$ GEFWA	proposal $\approx$ PSO $\approx$ GEFWA	proposal $\approx$ PSO $\approx$ GEFWA
f_{24}	proposal $\approx$ PSO $\approx$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA
f_{25}	PSO $\approx$ proposal $\gg$ GEFWA	PSO $\approx$ proposal $\gg$ GEFWA	proposal $\gg$ PSO $\approx$ GEFWA
f_{26}	PSO $\approx$ proposal $>$ GEFWA	PSO $\approx$ proposal $\approx$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA
f_{27}	PSO $>$ proposal $>$ GEFWA	PSO $\approx$ proposal $\approx$ GEFWA	PSO $>$ proposal $\gg$ GEFWA
f_{28}	PSO $\approx$ proposal $>$ GEFWA	proposal $\approx$ PSO $\gg$ GEFWA	proposal $\gg$ PSO $\gg$ GEFWA

proposed strategy can achieve more careful local search automatically based on the characteristics of local fitness landscape without adding any fitness cost consumption.

Secondly, we want to discuss the potential of our proposed explosion strategy. Although we evaluated the simplest two-layer explosion model in this paper to improve the understanding of local fitness landscape, the proposed explosion mechanism can be extended to any layers to explore local fitness information more accurately. Note that the conventional FWA is a special case of our proposal as a single-layer explosion. Thus, our proposed explosion mechanism is flexible and can be adjusted to any number of layers as needed. It becomes a potential topic to develop an adaptive version determining the number of explosion layers according to the characteristics of optimization problems and computational cost.

Not only conventional FWA but also our proposed explosion strategy can be easily combined with other variations of FWA with few modifications to their original framework. In this paper, we select EFWA as a baseline algorithm. The main objective of this paper is to use the proposed multi-layer explosion strategy to speed

Table 6.9: Statistical test results of the Friedman test and Holm's multiple comparison test for average fitness values of 30 trial runs of different parameter settings of generated sparks in the first layer at the stop condition, MAX_{NFC}. FN3, FN4 and FN5 mean the number of generated spark individuals in the first layer is 3, 4 and 5 for each firework individual, respectively. $A \gg B$, $A > B$ and $A \approx B$ represent the same meaning as the symbols in the Table 6.7.

$Func.$	2-D	10-D	30-D
f_1	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN5 $\approx$ FN4 $\approx$ FN3
f_2	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_3	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_4	FN3 $\approx$ FN4 $\approx$ FN5	FN4 $\approx$ FN3 $\approx$ FN5	FN4 $\approx$ FN3 $>$ FN5
f_5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\gg$ FN5	FN3 $\gg$ FN4 $\approx$ FN5
f_6	FN4 $\approx$ FN3 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN5 $\approx$ FN4
f_7	FN3 $\approx$ FN4 $>$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN5 $>$ FN3 $\approx$ FN4
f_8	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN4 $\approx$ FN5
f_9	FN4 $\approx$ FN5 $\approx$ FN3	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN4 $\approx$ FN5
f_{10}	FN5 $\approx$ FN4 $\approx$ FN3	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\gg$ FN5
f_{11}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\gg$ FN4 $\approx$ FN5	FN3 $\gg$ FN4 $>$ FN5
f_{12}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $>$ FN4 $\approx$ FN5
f_{13}	FN3 $>$ FN4 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN5 $\approx$ FN4
f_{14}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN4 $\approx$ FN5
f_{15}	FN5 $\approx$ FN4 $\approx$ FN3	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_{16}	FN4 $\approx$ FN3 $\approx$ FN5	FN3 $\approx$ FN4 $>$ FN5	FN5 $\approx$ FN4 $\approx$ FN3
f_{17}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $>$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\gg$ FN5
f_{18}	FN4 $\approx$ FN3 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4	FN4 $\approx$ FN3 $\approx$ FN5
f_{19}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_{20}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN5 $\approx$ FN3 $\approx$ FN4
f_{21}	FN4 $\approx$ FN3 $\approx$ FN5	FN4 $\approx$ FN3 $\approx$ FN5	FN4 $\approx$ FN5 $\approx$ FN3
f_{22}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_{23}	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN4 $\approx$ FN5	FN4 $\approx$ FN3 $\approx$ FN5
f_{24}	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5
f_{25}	FN4 $\approx$ FN3 $\approx$ FN5	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4
f_{26}	FN3 $\approx$ FN4 $\approx$ FN5	FN5 $\approx$ FN3 $\approx$ FN4	FN3 $\approx$ FN4 $\approx$ FN5
f_{27}	FN3 $\approx$ FN5 $\approx$ FN4	FN3 $\approx$ FN5 $\approx$ FN4	FN4 $\approx$ FN3 $\approx$ FN5
f_{28}	FN4 $\approx$ FN5 $\approx$ FN3	FN3 $\approx$ FN4 $\approx$ FN5	FN3 $\approx$ FN5 $\approx$ FN4

up any FWA rather than improve a particular version, e.g. dynFWA and adaptive FWA. Surely, the proposed strategy can replace the explosion operations of these future versions easily, and we can infer that their performance is further improved based on our experimental results. Thus, we can say that our proposal is widely applicable and easy to use.

The third discussion is how to improve the cost-performance of our proposal. Although our proposal strengthens the acquisition of the local fitness landscape information, many generated spark individuals still are not fully used. As a possible attempt, we may identify more potential directions based on the shape of the multiple explosions and assigned number of spark individuals and use these pieces of information to guide evolution more effectively. Besides, past spark individuals can also be recorded to extract more accurate fitness landscape information, for example, using past information to make an approximate fitness model. Thus, it is also a topic worthy of continued research to improve the rate of used generated spark individuals.

To analyze the performance of our proposal, two controlled experiments were designed. For the first controlled experiment, we apply the Wilcoxon signed-rank test between conventional EFWA and conventional EFWA with our proposed strategy at the stop condition. The statistical results shown in the Table 6.7 confirmed that our proposed explosion strategy could improve the performance of FWA significantly, especially for complex cases (multimodal optimization and combinatorial optimization). For high-dimensional F_1 and F_2 unimodal functions, our proposal is worse than conventional EFWA. Perhaps it was because the proposal makes spark individuals explore local fitness landscape fully and thus lead to them distributed widely. It may cause the slow convergence for unimodal optimization, while it is useful to avoid falling into the local minima in multimodal optimization. For F_{16}, our proposal did not show accelerated convergence. It is necessary to further investigate to apply our proposed explosion strategy well.

The second controlled experiment compares our proposal with PSO and a powerful variant of FWA, guided FWA. The Kruskal-Wallis test and Holm's multiple comparison test are used to check significant difference among three algorithms at the stop condition. The statistical results shown in the Table 6.8 indicate that our proposal performs well in most cases and always have better performance than the guided EFWA. To ensure fairness of comparison, we apply our proposal and guiding spark strategy to the same baseline algorithm. The results reveal that our proposed strategy has stronger performance, and we can infer that our proposed strategy can be combined with not only the guided FWA but also any other variations of FWA to further enhance their performance. This is possible because the guiding spark strategy is only predicting a potential direction in a local area, while our proposed strategy can explore the local fitness landscape more comprehensively. Meanwhile, the proposed multi-layer explosion strategy uses sparks in the previous layer to generate new potential sparks to improve diversity. Thus, we can say that our proposal is promising thanks to its comprehensive exploration with local information.

We also compared our proposal with PSO. The statistical results showed that our proposal was better than PSO in low dimensions except f_{27}, but it became worse on some functions, e.g. f_3 and f_7 in the high dimension. It may be because the number of fireworks is insufficient for the increased dimensions, and PSO uses not only the local optimum of individuals in past generations but also the global best information. It gives us a new inspiration; strengthening the communication among fireworks may further increase performance of our proposal. One of our future works is to investigate the relationship between population size and dimensions and increase collaboration among individuals.

Analysis of parameter settings of proposal

To investigate the effect of the number of generated spark individuals in the first layer on the performance of our proposed strategy, we add an experiment and set different number of generated spark individuals in the first layer using the same parameter settings with the Table 6.5. We apply the Friedman test and Holm's multiple comparison test to check significant difference at the stop condition. The statistical results reproduced in the Table 6.9 show that the number of generated sparks does not have a large impact on the performance of our proposal. Although the difference is small, the smaller the number of sparks in the first layer, the higher the precision of the final result. When the computational cost is limited, we recommend not to spend too much resources in the first layer, and it depends on the optimization problem and computational cost.

Next, we want to discuss parameter settings. Although it has been discussed for long years, there is no unified methods to guide it. We often do not have sufficient its a priori knowledge for practical applications and a variety of optimization problems, and it is difficult to decide appropriate parameter values. As the first attempt, we set all parameters in this paper based on previous relevant literatures and experiences. We can note that the key point affecting performances is the distribution number of sparks in different layers. When the number of sparks in the first layer is small, sufficient information on a local fitness landscape cannot be obtained and sufficient resources for searching cannot be used in the following layers. Thus, it is important to balance the number of sparks in different layers and develop an adaptive allocation version.

Finally, when our proposed strategy is applied to different variants of FWA, we need to consider their characteristics and select appropriate parameter values to achieve the best performance. In summary, there is no established method to guide parameter settings, but we may be able to give rough recommended parameter settings through a large number of trial run experiments in our future work.

Potential and future topics

Inspired by the various explosion modes of real fireworks, we propose a multi-layer explosion strategy to enhance the understanding of the local fitness landscape. Many other different explosions can be developed to enhance different capabilities of FWA. For example, tree-shaped or fan-shaped explosions can be developed to track potential directions instead of searching randomly or equally in a space. As many spark individuals are generated by a firework individual in conventional FWA, it is also worth to develop different strategies to generate spark individuals to increase diversity and reduce too intensive risk. Overall, there is still much room to further improve the performance of FWA by introducing novel mechanisms.

It is an attractive topic to tune parameters adaptively to maintain the strong performance of FWA at all times. We can also develop adaptive versions from the following three aspects: (1) adaptive decision of the number of explosion layers without limiting to two layers, (2) adaptive decision of the distribution of spark individuals at different levels, and (3) adaptive decision of a search radius on each dimension.

6.4 Chapter Summary

In this chapter, we propose new search operations for FWA to enhance its performace and new explosion patterns, i.e. scouting strategy and multi-layer explosion strategy, to improve search efficiency. The controlled experiments confirmed that these proposed strategies can improve the FWA performance significantly.

In future work, we will further study these strategies and make full use of local information to obtain better performance, further analyse the applicability of our proposal and apply them to real world applications as our future works.

Chapter 7

Niche Strategies for Fireworks Algorithm

7.1 Distance-based Exclusive Strategy

FWA extracts the characteristics of the explosion of real fireworks and simulates these explosion patterns repeatedly to ensure convergence. Firework individuals take on different search capabilities including exploitation and exploration according to their fitness. A better firework individual with higher fitness generates more sparks within a narrow range, while a poor firework individual generates a fewer spark within a wide range. A firework individual and its generated spark individuals locate in a similar area and can be grouped for the same niche area. Inspired by the explosion search, each firework individual would search in a niche area, and no extra operations are required to divide the generated sparks individuals into other niche areas. The core idea of our proposal is to generate individuals into each niche area and make them search independently without overlapping [196]. Finally, each firework individual locates in a different local optimum area. Here is the implementation of our proposal in detail.

The key issue is how to avoid overlapping search among firework individuals. To address this issue, we propose a distance-based exclusive strategy to ensure that there is no overlapping search among firework individuals. Since each firework individual has its own explosion radius assigned adaptively according to its fitness, it can be used to detect overlap easily without introducing any new parameters. We determine that there is overlap search between the two firework individuals if the Euclidean distance between the two individuals is less than the sum of the search radii of these two individuals.

We check the distances between each firework individual and other fireworks which fitness is better than that of the reference firework and mark the poor firework individual if their searching radius overlaps. Until all firework individuals are sequentially detected, we delete the marked firework individuals and generate new firework individuals to replace them. These generated new firework individuals participate in the next round of explosion operations rather than deleted firework individuals. Note that we use synchronous update detection in this paper. Asynchronous detection is also acceptable and may be more reasonable, where a marked firework individual is immediately replaced by a new generated one. In fact, there are many methods to generate new firework individuals. We employ the opposite-based

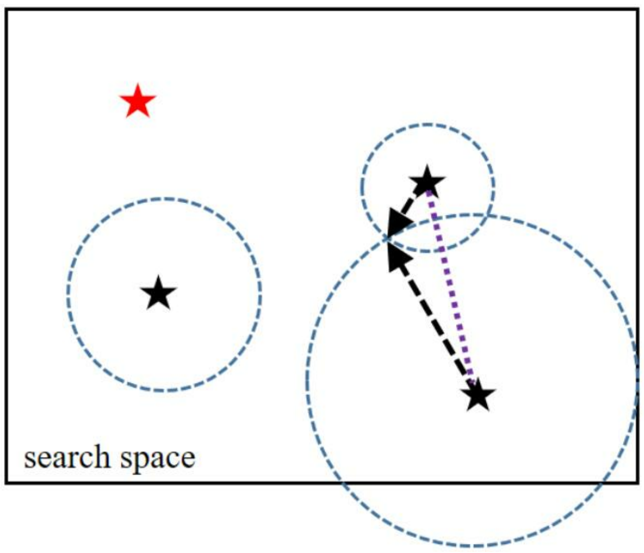


Figure 7.1: A demonstration of proposed distance-based exclusive strategy. Once the explosion radius of two firework individuals overlaps, i.e. the sum of black dotted line segments is longer than the purple dotted line segment, the poor firework individual is removed from the local area. Then, a new seaching firework individual (the red five pointed star) is generated to take over an explosion operation of the poor firework individual according to the opposite-based generation strategy.

generation strategy shown by the Eq. (7.1) to generate new firework individuals assigned to unexplored areas as much as possible.

$$x^j_{opposite} = X^j_{max} + X^j_{min} - x^j_i \tag{7.1}$$

where i and j mean the i-th firework individual and its j-th dimension, respectively. X^j_{max} and X^j_{min} are the upper and lower bound of the j-th dimension. Thus, x^j_i and $x^j_{opposite}$ are symmetrical with respect to the center of a search space.

The next point to be solved is how to search multiple niche areas. Although random selection and distance-based selection are widely used in the original FWA and several its variants, these selection methods may lose the diversity and the ability to search multiple niche areas. To overcome these limitations, local optima-based selection strategy [192] is used to ensure each local optimum firework individual kept in the next generation and maximize the diversity for searching more local optima areas. Note that the mutation operation is not used in our proposal, which it means all firework individuals are independent and the optimal individual (spark or current firework) in each niche area is copied to the next generation to form the next population. The Algorithm 11 shows the flow of our proposal combining with FWA.

7.2 Experimental Evaluations

We use eight benchmark functions with different dimensions from the CEC2015 multi-niche benchmark test suite [128] in our experimental evaluations; they are designed for real parameter bound constrained single-objective optimization. The

Algorithm 11 The framework of our proposal combining with FWA. Steps 4-10 are our proposed strategy.

```
1: Initialize $n$ firework individuals randomly.
2: Evaluate the fitness of each firework.
3: while a termination condition is not satisfied do
4:     for $i = 0; i < n; i++$ do
5:         Check whether the explosion radius is overlapped among firework individuals.
6:         if If there are overlaps then
7:             Mark the poor firework individuals.
8:         end if
9:     end for
10:    Replace the marked fireworks individuals with new individuals generated by the opposite-based generation strategy.
11:    Generate explosion sparks for each firework.
12:    Use Gauss mutation to obtain Gauss sparks (optional).
13:    if sparks are generated outside search area then
14:        use a mapping rule for bringing back to the area.
15:    end if
16:    Evaluate the fitness of each generated sparks.
17:    Use local optima-based selection strategy to select $n$ new fireworks for the next generation.
18: end while
19: end of program.
```

Table 7.2 shows their types, characteristics, variable ranges, and the number of global / local optima. These landscape characteristics include shifted, rotated, combination and multi-modal. The EFWA is used as the test baseline algorithm and combine it with our proposed strategy. The parameter settings used in our experiments are described in the Table 7.1.

Table 7.1: Parameter setting of EFWA.

Parameters	Values
# of fireworks for any dimension search	10
# of sparks m	50
# of Gauss mutation sparks	5
constant parameters	$a = 0.04$ $b = 0.8$
Maximum amplitude A_{max}	20
Dimensions D	2, 3, 4, 5, 6, 8, 10, 16 and 20
Max. # of fitness evaluations	20,000D
# of trial runs	30

To evaluate the performance of our proposal, we not only compare it with the conventional EFWA, but also with a well-known niching method, fitness sharing. The basic idea of fitness sharing is to restrict and share resources (e.g. fitness value) at each niche for decreasing redundancy in the population. Since the search range in all dimensions is limited to [-100,100], we set the niche radius at a fixed value 5 for fitness sharing in our experiments. It means that if a distance between two firework individuals is less than the niche radius, it is regarded that they are in the same area and thus need to share fitness. Fitness sharing is used to reassign fitness for firework individuals before the explosion operation is performed. Then, the reassigned fitness of firework individuals is used to determine the number of generated sparks and their explosion radius. The above description demonstrates how to integrate fitness sharing into the conventional EFWA.

Table 7.2: Benchmark Function.

No.	Functions	Dimension	# of global/local
F_1	Shifted and Rotated Expanded Two-Peak Trap	5	1/15
		10	1/55
		20	1/210
F_2	Shifted and Rotated Expanded Five-Uneven-Peak Trap	2	4/21
		5	32/0
		8	256/0
F_3	Shifted and Rotated Expanded Equal Minima	2	25/0
		3	125/0
		4	625/0
F_4	Shifted and Rotated Expanded Decreasing Minima	5	1/15
		10	1/55
		20	1/210
F_5	Shifted and Rotated Expanded Uneven Minima	2	25/0
		3	125/0
		4	625/0
F_6	Shifted and Rotated Expanded Himmelblau's Function	4	16/0
		6	64/0
		8	256/0
F_7	Shifted and Rotated Expanded Six-Hump Camel Back	6	8/0
		10	32/0
		16	256/0
F_8	Shifted and Rotated Modified Vincent Function	2	36/0
		3	216/0
		4	1296/0
All Search Range: [-100,100]			

We evaluate convergence against the number of fitness calls rather than generations for fair evaluations. We test each benchmark function with 30 trial runs in different dimensional spaces. The fitness accuracy error is set to 10^{-4}, which means that if the error between a found optimum and a true optimum is less than this value, then the found optimum is considered to converge to the true optimum. The maximum number of fitness calculations is set to $20,000D$, where D is the spatial dimension.

To analyze the performance among these algorithms, we record the average number of found optima after the 30 trial runs, as well as the maximum and minimum number of found optima at the stop condition, e.g. the maximum number of fitness calculations. The results are shown in the Table 7.3. We apply Friedman test and Holm's multiple comparison for average number of found optima at the stop condition of three methods to check the significance difference among them. The Tables 7.4 shows the statistical test result of average number of the found optima.

Table 7.3: We record the mean of 30 trial runs, as well as the maximum and minimum number of found optima at the stop condition. EFWA: enhanced FWA; EFWAWithFS: apply fitness sharing to EFWA; Poposal: apply our proposed strategy to EFWA.

No.	Dimension	EFWA			EFWAWithFS			Proposal		
		Mean	Min.	Max.	Mean	Min.	Max.	Mean	Min.	Max.
F1	5	0	0	0	0.033333	0	1	0	0	0
	10	0	0	0	0	0	0	0	0	0
	20	0	0	0	0	0	0	0	0	0
F2	2	1	1	1	1	1	1	2.166667	1	4
	5	0.833333	0	1	0.933333	0	1	4.933333	2	9
	8	0.3	0	1	0.366667	0	1	1.933333	0	4
F3	2	1	1	1	1	1	1	7.133333	4	9
	3	1	1	1	1	1	1	3.333333	1	7
	4	1	1	1	0.9	0	1	1.4	0	5
F4	5	0.033333	0	1	0	0	0	0	0	0
	10	0	0	0	0	0	0	0	0	0
	20	0	0	0	0	0	0	0	0	0
F5	2	1	1	1	1	1	1	6.833333	4	9
	3	1	1	1	1	1	1	5.966667	3	8
	4	1	1	1	0.933333	0	1	1.833333	0	6
F6	4	1	1	1	1	1	1	8.033333	6	9
	6	1	1	1	1	1	1	9	9	9
	8	1	1	1	1	1	1	9	9	9
F7	6	0.266667	0	1	0.233333	0	1	0.233333	0	1
	10	0.033333	0	1	0.033333	0	1	0	0	0
	16	0	0	0	0	0	0	0	0	0
F8	2	1.8	1	6	1.833333	1	7	6.4	3	9
	3	1.633333	1	5	1.333333	1	3	8.766667	8	9
	4	1.4	1	4	1.133333	1	3	8.4	6	9

7.3 Discussions

We begin our discussion on an explanation of the superiority of our proposed strategy. Our proposal uses the explosion pattern of FWA fully to make each firework individual stay in a different niche area. The independent selection mechanism can keep the diversity of the population and ensure that multiple niche areas are searched well.

In comparison to other existing niching methods, the proposed strategy does not introduce any new parameters to divide individuals into different niche areas because each local explosion operation can be considered a niche. The proposed strategy needs extra fitness calculations to generate new firework individuals when

Table 7.4: Statistical test results of the Friedman test and Holm's multiple comparison for average number of found optima among three algorithms at the stop condition. $A \gg B$ and $A > B$ mean that A is significantly better than B with significant levels of 1% and 5%, respectively. $A \approx B$ means that there is no significant difference between them although A is better than B. The abbreviation used in this table is the same with those used in the Table 7.3.

No.	Dimension	Statistical test results
F1	5	EFWAWithFS $\approx$ Proposal $\approx$ EFWA
	10	EFWAWithFS $\approx$ Proposal $\approx$ EFWA
	20	EFWAWithFS $\approx$ Proposal $\approx$ EFWA
F2	2	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
	5	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
	8	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
F3	2	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
	3	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
	4	Proposal $\approx$ EFWA $\approx$ EFWAWithFS
F4	5	EFWA $\approx$ Proposal $\approx$ EFWAWithFS
	10	EFWA $\approx$ Proposal $\approx$ EFWAWithFS
	20	EFWA $\approx$ Proposal $\approx$ EFWAWithFS
F5	2	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
	3	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
	4	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
F6	4	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
	6	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
	8	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
F7	6	EFWA $\approx$ EFWAWithFS $\approx$ Proposal
	10	EFWA $\approx$ EFWAWithFS $\approx$ Proposal
	16	EFWA $\approx$ EFWAWithFS $\approx$ Proposal
F8	2	Proposal $\gg$ EFWAWithFS $\approx$ EFWA
	3	Proposal $\gg$ EFWA $\approx$ EFWAWithFS
	4	Proposal $\gg$ EFWA $\approx$ EFWAWithFS

search areas are overlapped. However, it can avoid multiple firework individuals searching in the same local area repeatedly, and the possibility that new generated firework individuals locate in an unexplored area becomes high. Besides, our proposal always keeps the optimal area unreplaced, and the poorer the area is, the higher the possibility that the area is cleared is. Thus, this strategy can avoid the wandering search in the same area and maintain multiple niches with low increased cost. We can say that it is a *low risk, high return* strategy.

Secondly, we want to discuss on the potential and usability of our proposal. The proposed strategy can be combined with any other FWA variations easily to track multiple different niches without changing their main framework; it is not limited to EFWA used in our experiments. It can be further extended to other EC algorithms. For example, each particle in PSO has a maximum evolutionary step (maximum speed) that can be used to detect whether the particles located in the same local area. Subsequently, worse particles in the same local area are replaced with newly generated particles to keep multiple niche search. It is only a preliminary idea, and a more perfect framework still needs to be discussed. The opposite-based generation strategy is adopted to generate new firework individuals in this paper, but other generation strategies are also acceptable, e.g. random generation, Forex trade strategy generation, and others. Thus, how to generate firework individuals appropriately at different search stages can be developed as a new potential topic. Overall, we can say that the proposed strategy has a strong plasticity and promising.

To analyze the performance of our proposal, the Friedman test and Holm's multiple comparison are applied to check the significant difference at the stop condition. From the results of the statistical tests, we have found that our proposal does not weaken the performance of FWA on all benchmark functions, and satisfactory performance was obtained on F_2, F_3, F_5, F_6 and F_8. On the other hand, fitness sharing does not play any role in finding multiple optima. It may be because it cannot keep multiple niche search with convergence and produces firework individuals to converge to the same local optimum area although it can reduce redundancy. It also implies that our proposal can keep diversity well. Both, F_1 and F_4 have only one global optimum and multiple local optima. None of three algorithms converge to the global optimal area of these two benchmark functions, which may be caused by the fact that the global optimum is hard to find. Our proposal had no effect on F_7. Its specific reasons are still unclear, and we will investigate in our future work, which may help us to design more reasonable strategies to optimize multimodal problems.

Some open topics and possible improvements are given. Thanks to the explosion search of FWA, it can be developed easily for multimodal optimization problems. But, it limits the population size not to be set too large because abundant resources (fitness evaluations) are used to generate spark individuals for local search. The local optima-based selection strategy may facilitate the implementation of parallel computing, though it is not conducive to exchange information among firework individuals and not easy to jump out of local optima. Things have pro and con, and it is necessary to further consider how to balance these conflicts to pursue maximum performance. Thus, our proposal can ensure each firework individual may locate an optimum, that determines that it is not suitable for optimization problems with massive optima.

Finally, we want to discuss on parameter settings. Since there are no mature methods to guide the parameter settings, all parameters used in experiments are set based on our experiences. Actually, adaptive detection mechanisms are potential approaches rather than a fixed strategy according to the convergence progresses or the characteristics of a fitness landscape. Thus, developing an adaptive version has become a challenge in our future works.

7.4 Chapter Summary

We introduced a distance-based exclusive strategy to FWA to solve multimodal tasks. Each individual can stay in different niche area and can converge to different optima. The experiments have confirmed that our proposed strategy is effective and can ensure that FWA locates in multiple different optima areas.

In our future work, we will consider the fitness characteristics of optimization problems and use the collected feedback in EC process to develop a more intelligent version to improve performance of our proposal.

Chapter 8

Competitive Strategies for Differential Evolution

8.1 Improvable Issue of Differential Evolution

In general, better individuals are expected to locate closer to the global optimum than poorer ones and have higher possibility of generating better offspring. Worse individuals may search in hopeless areas or far away from the global optimum and need more computational resources to approach to the global optimum than better individuals. However, canonical DE and several DE variants have a fair policy, and provide an equal opportunity to all parent individuals. This make each of them generate only one offspring individual. We thought that this fair policy may be the point where we can improve DE.

DE compares a parent with its offspring generated in the same local area, and a winner is put in the next generation. This hill climb selection strategy can ensure outstanding offspring replaces its parent so that the whole population evolves steadily towards better areas. However, there must be the case that the replaced parents are better than other individuals in the current population. If poor offspring is generated by a poor parent and is not replaced, computational resources is used for exploring in the same local area until better offspring is generated. This reduces convergence speed and hinders population distribution to converge. This is not a good for especially large-scale optimizations or high computational cost problems. We thought that developing methods for handling poor individuals should be a point where we can improve DE.

Some literatures suggest that increasing competition pressure among individuals is beneficial to accelerating convergence [151, 146]. Thus, we propose two competitive strategies, *competitive generation strategy* and *competitive selection strategy*, to increase the competitions among individuals and aim to enhance speed of eliminating inferior individuals and increase the possibility of generating more outstanding offsprings [189].

8.2 Two Competitive Strategies

Competitive Generation Strategy

We propose a competitive mechanism that gives better parent individuals higher opportunities to generate more offspring rather than only one offspring; A parent of canonical DE genetates one offspring individual [152, 123, 124]. We firstly mark the worst $a\%$ individuals and compare each of them with a randomly selected individual from the current population. If the latter is better than former, the randomly selected parent can have one more opportunity to generate an offspring and the marked poor parent lose the opportunity to do that. Otherwise, we do nothing and follow the canonical DE processing. This strategy guarantees not to change the population size but allow better individuals to generate more offspring and suppress poor individuals to generate their offspring. The Fig. 8.1 demonstrates our proposed strategy.

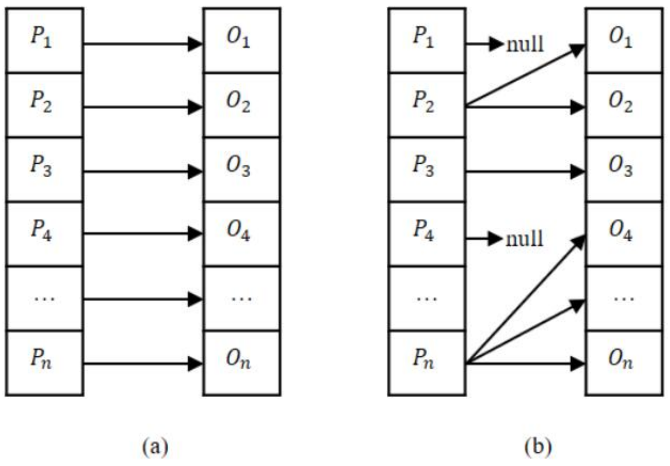


Figure 8.1: Canonical DE makes each parent, p_i, generate only one offspring, o_i. Proposed competitive generation strategy makes parents compete each other; some parents can generate multiple offspring, but others may generate no offspring.

Competitive Selection Strategy

This competitive strategy handles the worst individual in a potential hopeless searching area. The competitive selection strategy is a proposal to compare a newly generated offspring, i.e. a trial vector, with the worst individual in the current population and a winner is put in the next generation. The canonical DE compares it with its parent, i.e. a target vector [152, 123, 124]. This strategy ensures that better individuals can survive and poorer individuals are eliminated, which is expected to improve search performance. Algorithm 12 shows the flow of DE with our two proposed strategies.

Algorithm 12 DE framework with our two proposed strategies. Step 4 describes the first strategy, and Steps 10-14 describes the second strategy.

```
1: Initialize population randomly.
2: Evaluate the population.
3: while a termination condition is not satisfied do
4:     Determine the number of generated offspring for each parent individual using
       the first strategy (the total generated offspring is unchanged).
5:     for $i = 1 \ldots$ population size $m$ do
6:         while an individual has the opportunity to generate offspring do
7:             Choose two individuals, $\boldsymbol{x1}_i$ and $\boldsymbol{x2}_i$, randomly and one base vector,
               $\boldsymbol{base}_i$ from the best individual (DE/best) or randomly (DE/rand).
8:             Calculate a mutant vector as $\boldsymbol{mutant}_i = \boldsymbol{base}_i + F * (\boldsymbol{x1}_i - \boldsymbol{x2}_i)$, where
               $F$ is a scale factor.
9:             Cross the mutant vector and a target vector and generate a trial vector.
10:            if the generated offspring is better than the worst individual then
11:                Replace the worst one with the offspring.
12:            else
13:                Keep the population unchanged.
14:            end if
15:        end while
16:    end for
17: end while
18: end of program.
```

8.3 Experimental Evaluations

We implement two proposed competitive strategies into conventional DE and evaluate their effectiveness using 20 benchmark functions from the CEC2013 benchmark test suite [91] with three dimensional settings, 2-D, 10-D, and 30-D. Their landscape characteristics include shifted, rotated, global on bounds, uni-modal and multi-modal. The DE parameter settings are shown in the Table 8.1.

Table 8.1: DE algorithm parameter setting.

population size for 2-D, 10-D, and 30-D search	100
scale factor F	0.7
crossover rate	0.9
DE operations	DE/rand/1/bin
stop criterion; max. # of fitness evaluations for 2-D, 10-D, and 30-D search	1,000, 10,000, 40,000
dimension of benchmark functions, D	2, 10, and 30
# of trial runs	30

We compare four DE variants to evaluate two proposed strategies: canonical DE, (canonical DE + competitive generation strategy), (canonical DE + competitive selection strategy), and (canonical DE + both strategies). We run these four DE variants using 20 benchmark functions $\times$ 3 different dimensions $\times$ 30 trial run and compare their fitness values at the stop condition, i.e., the maximum number of fitness evaluations. We apply the Friedman test and Holm's multiple comparison test to these fitness values for each benchmark function to check for significant difference among the averages of four DE variants. The Table 8.2 shows the result of these statistical tests.

Table 8.2: Statistical test results of the Friedman test and Holm's multiple comparison for average fitness values of 30 trial runs of 4 methods. $A \gg B$ and $A > B$ mean that A is significant better than B with significant levels of 1% and 5%, respectively. $A \approx B$ means that there is no significant difference between them. Numbers in the table represent that 1: canonical DE, 2: DE + proposed strategy 1, 3: DE + proposed strategy 2, and 4: DE + proposed strategies 1 and 2.

	2-D	10-D	30-D
f_1	$3 \approx 4 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
f_2	$3 \approx 4 \gg 1 > 2$	$3 \approx 4 \gg 1 \approx 2$	$4 \approx 3 \gg 2 \approx 1$
f_3	$3 \approx 4 \gg 1 \approx 2$	$3 \approx 4 \gg 1 \approx 2$	$3 \approx 4 \gg 1 \approx 2$
f_4	$3 \approx 4 \gg 1 > 2$	$3 \approx 4 \gg 1 \approx 2$	$3 \approx 2 \approx 4 \approx 1$
f_5	$4 \approx 3 \gg 1 > 2$	$4 \approx 3 \gg 1 > 2$	$4 \approx 3 \gg 1 > 2$
f_6	$3 \approx 4 \approx 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
f_7	$4 \approx 3 \gg 1 \approx 2$	$3 \approx 4 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
f_8	$1 \approx 4 \approx 2 \approx 3$	$2 \approx 1 \approx 3 \approx 4$	$3 \approx 4 \approx 1 \approx 2$
f_9	$4 \approx 3 \approx 2 \approx 1$	$4 \approx 2 \approx 1 \approx 3$	$3 \approx 4 \approx 1 \approx 2$
f_{10}	$2 \gg 4 \approx 3 \gg 1$	$3 \approx 4 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
f_{11}	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 3 \approx 1 \approx 2$	$3 \approx 4 \approx 1 \approx 2$
f_{12}	$3 \approx 4 \approx 1 \approx 2$	$4 \approx 3 > 1 \approx 2$	$3 \approx 4 \approx 2 \approx 1$
f_{13}	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 3 \gg 2 \approx 1$
f_{14}	$4 \approx 3 \approx 2 \approx 1$	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 1 \approx 2 \approx 3$
f_{15}	$4 \approx 1 \approx 3 \approx 2$	$3 \approx 1 \approx 4 \approx 2$	$3 \approx 1 \approx 2 \approx 4$
f_{16}	$2 \approx 3 \approx 4 \approx 1$	$3 \approx 2 \approx 4 > 1$	$3 \approx 2 \approx 1 \approx 4$
f_{17}	$4 \approx 2 \approx 3 \approx 1$	$4 \approx 3 \gg 1 > 2$	$3 \approx 4 \gg 1 \approx 2$
f_{18}	$3 \approx 4 \approx 2 \approx 1$	$4 \approx 3 \gg 1 \approx 2$	$3 \approx 4 \gg 1 \approx 2$
f_{19}	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
f_{20}	$4 \approx 3 \approx 2 \approx 1$	$3 \approx 4 > 1 \approx 2$	$3 \approx 4 \approx 2 \approx 1$

8.4 Discussions

Analysis of Additional Calculations Cost

Both of two proposed strategies do not increase any extra fitness calculations. The first strategy introduces a new parameter, a%, to control the proportion of individuals participating in the competition. When the a is set to 0, it becomes the conventional DE method. On the contrary, when a is set to 100, all individuals participate in competition. The second strategy changes only the comparison target from a parent to the worst individual in the current population and does not increase computational cost.

We discuss a parameter a% in the competitive generation strategy. If the fitness of individuals is ranked in the worst a%, they may lose a chance of generating offspring with $1 - a\%$ probability. The average probability that all participating individuals generate no offspring is $\sum_{i=0}^{a\%*PS}(PS - a\% * PS - i)/PS$, where PS means the population size. The poorer the fitness of individuals is, the easier it is to lose an opportunity of generating offspring.

Discussion on Competitive Generation Strategy

The first competitive generation strategy gives multiple chances of generating offspring to better individuals, which means that some poor individuals may become unable to generate offspring due to a fixed population size. This feature causes a new problem, i.e., worse individuals that lose the chance are not replaced because of no generated offspring. Along with generations, potential individuals become better and better, but it becomes more and more difficult for these worse individuals to evolve and they remain as they are.

Although it ensures the increase of better individuals, it also ensures the increase of worse individuals remaining without evolution. This problem is solved by combining the second competitive selection strategy in the next subsection. The bigger value of the parameter, $a\%$, increases the number of the mentioned poor individuals, but its smaller value reduces the speed of generating better individuals. How to balance this confliction is a crucial issue. We can include that applying the first competitive strategy at every k-th generation instead of every generation to keep the chance that poor individuals evolve; introducing a dynamic parameter, $a\%$, to adjust the proportion of individuals participating in the competition according to generations.

Discussion on Competitive Selection Strategy

The second competitive selection strategy is to accelerate eliminating worse individuals by comparing generated offspring with the worst individual. Regardless the superiority of offspring to its parent, the offspring has an opportunity to survive in the next generation when it is better than the worst individual. If it replaces the worst one and the worst one has not yet generated its own offspring, it inherits the opportunity of the worst individual and generates a new offspring. This strategy can accelerate individual evolution towards to the global optimum. However, it also increases the risk of premature convergence, because both parent individual and its offspring in a local area remain, when the offspring is better than the worst individual. This may reduce population diversity. One of ideas to alleviate competitive overpressure and premature convergence is to compare offspring with a randomly selected individual from the worst $p\%$ individuals rather than only the worst individual.

From the experimental results, we can conclude that the proposed strategies, especially the second strategy, significantly improve optimization performance of DE . The proposed strategies have not shown any performance deterioration. We can observe that these strategies make DE convergence faster in early generations.

We apply the Friedman test and the Holm multiple comparison test among canonical DE and the combinations of proposed strategies with DE. Although DE with proposed competitive strategies works well for majority of benchmark functions, they do not show significant difference for f_8 (Rotated Ackley's function), f_9 (Rotated Weierstrass function), f_{14} (Schwefel's function), and f_{15} (Rotated Schwefel's function). It may be the reason that these functions have many local optima. We need further analysis of these results to investigate the exact reasons well, and develop more suitable competition strategy for DE optimization.

8.5 Chapter Summary

We proposed two strategies, the competitive generation strategy and the selection strategy, for the DE algorithm to increase competition among individuals and accelerate its convergence. The controlled experiments confirmed that proper introduction of the competition mechanism can improve the optimization performance of canonical DE. Concretely speaking, the competitive selection strategy is effective, which eliminates the worst individual and generates its alternative near a better individual. However, the acceleration performance of the competitive generation strategy that gives potential individuals more opportunities to generate offspring is not obvious.

In future work, we will further study the proposed competitive strategies and try to obtain a balance between competition and cooperation. Consequently, how to rationally eliminate poor individuals and maintain the diversity of population is also an open topic for further investigation. We will discuss and analyse these topics.

Chapter 9

Proposal of a New Algorithm: Vegetation Evolution

9.1 Vegetation Evolution

9.1.1 Motivations

Many varied and colorful species have emerged on the earth after a long period of biological evolution and changes in the natural environment, while those species that could not adapt have been eliminated. In the EC community, a great number of practitioners have found inspiration in the natural selection and the behaviour of animals [12, 112]. However, few people have paid attention to the evolution and growth of vegetation and found inspiration in it [183]. Actually, of all living things on the earth, a great part is made up of not humans, but of vegetation. Humans, together with all other animals, accounted for only a small portion of life. Vegetation has evolved many effective and intelligent survival mechanisms to adapt to various environments, which include the valuable innovation of perennial evolution. Consequently, there is a large potential for developing new EC algorithms by extracting and simulating the behaviors of vegetation. We try to summarize some of their mechanisms and develop a new optimization algorithm based on them.

Many plants begin their lives from seeds, just as animals grow up from fertilized eggs. Most plants take time to grow from seed to maturity, during which time they use their unique survival mechanisms to ensure their growth. Different plants have different times to reach maturity and different growth patterns. For example, some can grow into towering trees after decades, while some grow into grass after only a few months. However, the diversity of these survival mechanisms does not prevent us deriving some inspiration from them; we can simply abstract this growing stage as a local exploitation of an individual around itself. Once a plant has absorbed enough nutrients to grow and mature, they will generate new seeds to ensure the continuation of their species. Generally, a plant does not generate only one seed but hundreds of seeds. These generated seeds are dispersed everywhere and open a new round of their growth in these new environments. We can roughly generalize this propagation stage as a wide exploration.

Vegetation diffuses its offspring through four main methods [142]. (1) Water dispersal: plants growing in water or on the water use this method to disperse their offspring. For example, coconuts generally grow near the sea, and their matured fruit including seeds will fall into the sea and drift with the currents. When they drift to shore, they slowly germinate and grow there to become new coconut trees if the environment they have arrived at is suitable. (2) Wind dispersal: the wind is everywhere and scattered seeds, usually small and light, can be suspended in the air and blown to distant places by the wind. With some seeds, the surface has hair, wings or other mechanisms which are suitable for being carried by the wind. This dispersal can be seen everywhere in our daily life, such as with the dandelion, hyacinth and nodding burnweed. (3) Autologous dispersal: plants use their own power to disperse their seeds without resorting to external forces. Impatiens are ejected by the elastic force produced by the bending of the pericarp when their seeds mature. In another case, when the fruit of ecballium elaterium is ripe, a strong internal pressure is generated and its seeds are sprayed. (4) Animal carrying: here, dispersal primarily depends on the mobility of animals to carry the seeds to other places. Bidengrass, Xanthium and others have hooks on their seed coats which can latch onto animal fur. Some plants attract animals with their bright colors to be fed on. They spread with animal feces and spread in all directions because they cannot be digested. Squirrels stored pine nuts and also help pine trees to spread their seeds. There are some other ways by which seeds are propagated which we will not expand upon here.

Although vegetation has evolved these complex mechanisms for survival, we can obtain some pieces of inspiration and simplify the mechanisms to develop a new optimization algorithm. By studying the above growth and propagation patterns, we can roughly know that the growing periods for plants can be divided into two. First, plants need to absorb nutrients to grow in places where they take roots. This process can be seen as an exploitation in a local area. Next, the mature plants use various forces to disperse their seeds widely. This process can be viewed as an exploration over a wide range. Together these processes can form an optimization framework for a new algorithm which we call vegetation evolution (VEGE) [194].

9.1.2 Frameworks of Vegetation Evolution

Inspired by the evolution of vegetation, we develop a simplified model and realize a new evolutionary framework for optimization. Here, we focus on how we transformed these concrete existing survival mechanisms into the new evolutionary algorithm, VEGE [194].

In VEGE, each seed is abstracted as an individual. Each individual behaves differently during its two different growing periods, the *growth period* and the *mature period*, to realise different search capabilities. When an individual grows to maturity, it generates multiple new seed individuals rather than just one seed. Each individual in the current population generates the same number of multiple seeds, forming a temporary seed population. Finally, individuals in the next generation are selected from a mixed population consisting of the current population and the temporary seed population. The above process is repeated until the termination condition is satisfied, and VEGE outputs eventually find the optimal solution. Fig. 9.1 illustrates the general process of our proposed VEGE algorithm. Here, we present the implementation of the algorithm in detail.

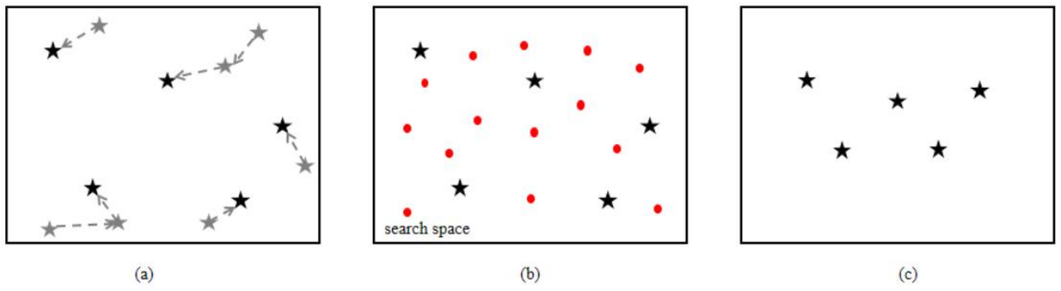


Figure 9.1: The search process of our proposed VEGE algorithm. (a) The initial population is randomly generated, and experiences a growing period until the individuals enter their mature period. Dotted arrows indicate the growth directions of individuals within a local area. (b) All individuals enter a mature period, and each individual generates multiple seed individuals. Red circles indicate generated seed individuals and all of them form a temporary seed population. (c) Individuals in the new generation are selected from all individuals in step (b). All individuals in the new generation experience a growing period again, and steps (b) and (c) are iterated until a termination condition is satisfied.

Growth period

When a plant roots in a new environment, it absorbs nutrients to maximize its healthy growth by extending the roots and opening branches. Inspired by this observation, we model this process as local search to simulate plant growth and evolve into potential directions. All individuals during this growth period are made responsible for exploitation by emphasizing competition among individuals.

The key problem is: how can we achieve the local search for an individual? Although different real-life plants have different mechanisms to control their growth, we roughly extract two factors to control local search: growth direction and growth length. Once these two factors are determined, the morphology of plant growth is also generally determined.

Corresponding to our proposed algorithm, we employ a new parameter, *growth radius*, to control the local search. As with the first attempt, we use a stochastic strategy to generate offspring randomly within the growth radius. Thus, the direction from an individual to its randomly generated offspring can represent a growth direction and the distance between them implies a growth length. When a generated offspring is better than its parent, we can say that there is a potential to grow along this direction. In this case we can consider it to have been a successful local search, and we replace the offspring with its parent. Otherwise, the growth direction is poor, and the parent individual does not move to its offspring and is kept the same. This growth mode is also very common in nature. Some plants do not grow constantly, but instead sometimes choose to stop growing so that they can endure harsh environments - or, conversely, they grow more vigorously. In this paper, we create a simple model of real plant growth for the local search of our proposed VEGE algorithm. Since there are many more complex mechanisms involved in the control of growth in the real world, it will be a challenge in the future to further develop more efficient local searches by learning more about the novel mechanisms

Algorithm 13 The general framework for an individual searching during the growth period. GC, GR and x_i mean the growth cycle, the growth radius and the i-th individual in the current population, respectively.

```
1: $count1 = 1$.
2: while $count1 \leq GC$ do
3:     An individual generates an offspring individual for exploitation, $x_{next}$, using
       $x_{next} = x_i + GR * rand(-1, 1)$ in each dimension.
4:     if the $x_{next}$ is better than the $x_i$ then
5:         The generated $x_{next}$ individual replaces its parent $x_i$.
6:     else
7:         The $x_i$ is kept as a parent individual.
8:     end if
9:     $count1 = count1 + 1$.
10: end while
```

employed by vegetation.

The next problem to be solved is how to control the amount of growth. In general, not all plants can keep growing in the real world without limit; some will stop growing when they reach certain conditions. Additionally, the cost of fitness calculations is also limited in EC algorithms when we apply them to solve realistic problems. Thus, another new parameter, *growth cycle*, is introduced to control the maximum amount of growth to something reasonable. In our proposed algorithm, we count the amount of growth not only when generated offspring is better than its parent but also when the opposite is true (see the 9th line in Algorithm 13). Thus, these two parameters can control the growth period of individuals. Once the number of growth cycles reaches the predefined maximum number, an individual enters its mature period. Certainly, these two parameters can also be changed adaptively. However, we set these two parameters as constants in our later experiments. Algorithm 13 summarizes the general framework for an individual behavior in a growth period.

Mature period

When plants mature in nature, they usually generate many seeds rather than just one seed to ensure that some seeds which are suitable for survival can undertake the task of further increasing the population. Not all generated seeds will succeed in growing as new plants, and some will be eliminated and then die. Inspired by this reproductive phenomenon, we adopt a one-to-many generation mechanism to increase population diversity: each individual which has entered its mature period generates multiple seeds regardless of its fitness to achieve exploration in a wide area through interspecies cooperation.

The key problem is how to generate many seed individuals and how to disperse them widely. Although plants have a variety of methods for dispersing seeds, we extract two factors to decide the spread of seeds roughly: *dispersed direction* and *dispersed distance*. Once these two factors are determined, we can easily decide the new rooting positions for the newly generated seeds.

Algorithm 14 The general framework of an individual search during the mature period. MS is the moving scale; x_i is the i-th individual in the current population; x_1 and x_2 are two randomly selected different individuals and are different from x_i.

```
1: $count2 = 1$.
2: while $count2$ is less than the maximum number of generated seed individuals of $x_i$ do
3:     The individual generates a new seed individual, $x_{seed}$, using $x_{seed} = x_i + MS * (x_1 - x_2)$.
4:     Record the generated seed individual into a seed population for selecting the next generation.
5:     $count2 = count2 + 1$.
6: end while
```

In complex natural environments, many plants choose to work together to ensure reproduction. Corresponding to our proposed algorithm, we use the differential information between individuals to simulate the cooperation between plant individuals. To be precise, we randomly select two individuals and make a difference vector between them. The first reason we adapt a differential vector is that it is easily calculated, and the second one is that any differential vector between any two individuals forms a share of the population distribution and therefore a differential vector contains a part of this information. Thus, a differential vector can indicate the direction of seed propagation.

In addition, we employ a new parameter, *moving scaling*, to scale dispersed distance. It is a randomly generated constant because, following the analogy for the spread of seeds in nature which is often influenced by many factors such as wind strength, water flow velocity and others. The new parameter, *moving scaling*, simulates these uncertain factors to increase the random search performance of our proposed algorithm. We set up the *moving scaling* randomly generated in $[-2, 2]$ in this paper based on our experience. In the above, we have described how an individual produces seed individuals through the cooperating between multiple individuals.

A subsequent problem that needs to be addressed is how to determine the number of seeds generated by each individual. Actually, different kinds of plants generate different numbers of seeds, and even the number of generated seeds by different individuals of the same species may be different. As a first attempt, we do not simulate the various complex realities determining seed numbers but introduce only a new parameter, *number of seeds*, to control the number of seed individuals generated by each mature individual. In our proposal, we handle all individuals equally and make them generate the same number of seeds. Thus, we can embed the reproduction of real plants into our algorithms. Algorithm 16 summarizes the general framework for an individual ’ s behavior during its mature period.

The last important problem is how to select individuals for the next generation during the mature period. Since an individual generates multiple seed individuals during its mature period, the total number of generated seed individuals will exceed the population size. Thus, we must devise a reasonable method for selecting some potential individuals for the next generation. There are many famous selection methods, such as tournament selection, truncation selection, ranking selection and others. As a first attempt, we use a greedy strategy to select the most promising individuals from a mixed population consisting of the current population and the seed population. All individuals are sorted according to their fitness, and the top PS individuals are selected into the next generation. Although this selection method can always maintain the most high potential individuals, it also has some risks, such as loss of diversity, becoming trapped at a local optimum, and others. The next challenge is to develop more efficient selection methods. A feasible approach for solving this problem would be to model the more sophisticated and complex selection mechanisms employed by real plants.

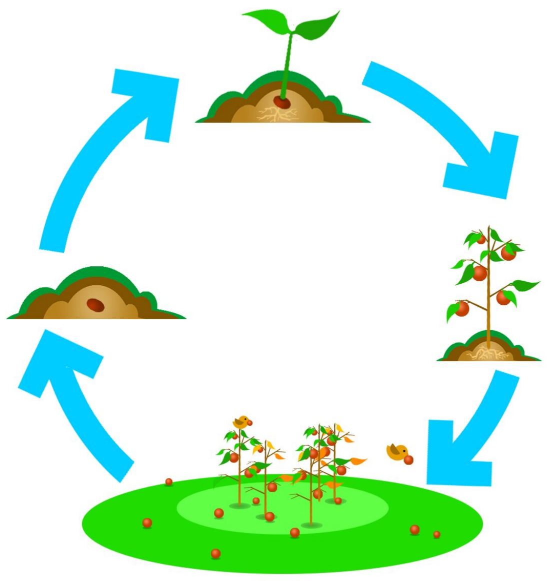


Figure 9.2: The roughly growth state of tomatoes. We subjectively divide it into two periods, a growth period and a mature period.

So far, we have roughly embed our inspirations from vegetation growth into an EC algorithm. An individual iterating a growth period and a mature period can be seen as a cycle. Immediately, the newly selected seed individuals begin a new round of the cycle from the growth period. Thus, our proposed VEGE algorithm iteratively executes this cycle towards finding the optimum. Algorithm 15 outlines the general framework of the whole of our proposed algorithm. Additionally, Fig 9.2 roughly illustrates the growth cycle of a tomato, which may help to understand our proposed algorithm. To be consistent with existing EC algorithms, we define that a cycle does not equal one generation, but includes multiple generations. It means that all individuals which are undergoing a growth period or a mature period are called a generation. Thus, an individual does not perform both a growth period and a mature period contiguously in one generation, but performs either a growth period or a mature period.

Algorithm 15 The general framework of our proposed VEGE algorithm.

```
1: Initialize the population randomly.
2: Evaluate the population.
3: if an individual is in a growth period then
4:    for $i = 1, \dots,$ population size do
5:       Perform Algorithm 1 for local search (exploitation).
6:    end for
7: else
8:    for $i = 1, \dots,$ population size do
9:       Perform Algorithm 2 for wide search (exploration).
10:   end for
11:   Mix the current population and seed population, and apply a selection strategy
      to update the current population.
12: end if
13: Output the found optima.
```

9.1.3 Experimental Evaluations

To evaluate the performance of our proposed VEGE algorithm, we compare it with three other algorithms, DE, PSO and EFWA. For this evaluation experiment, 28 functions from the CEC2013 test suite [91] are used as a test bed with three dimensional settings, 2-dimensions (2-D), 10-D, and 30-D, which is devoted to the approaches, algorithms and techniques for solving real parameter single objective optimization.

Table 9.1: VEGE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	10
growth cycle GC	6
growth radius GR	a random number in [-1,1]
total seed individuals SI for 2-D, 10-D, and 30-D search	60
moving scaling MS	a random number in [-2,2]
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 9.1 shows the VEGE parameter settings used in our experiments. Each individual evolves, alternately performing the two operational periods. Since the ranges for all variables of all benchmark functions are in [-100, 100], we set the upper and lower limits of the growth radius, GR, to 1 and -1 for a local search. The maximum moving scaling, MS, is a random number in [-2, 2] for a wide search. The Tables 9.2, 9.3 and 9.4 shows the parameter settings of DE, PSO and EFWA used in our experiments, respectively.

For fair evaluations, we evaluate convergence along the number of fitness calls rather than generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces. The Kruskal-Wallis test and Holm's multiple comparison test are applied to check whether there is a difference among all the algorithms

Table 9.2: DE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	60
scale factor F	0.8
crossover rate	0.9
DE operations	DE/rand/1/bin
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 9.3: PSO algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	60
inertia factor w	1
constant c_1 and c_2	1.49445, 1.49445
max. and min. speed V_{max} and V_{min}	2.0, −2.0
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

at the stop condition, i.e. the maximum number of fitness calculations. Table 9.5 shows the result of these statistical tests.

9.1.4 Discussions

Analysis of VEGE components

We begin our discussion with an explanation of the superiority of our proposed algorithm. The new population-based VEGE is inspired by the vegetation growth and its seed dispersal. It realizes different search capabilities by simulating the different periods of vegetation life. All individuals in the growth period are responsible for exploitation. There is no communication among them and they compete independently to search within an area for greater potential. When the number of local search reaches the maximum predefined growth cycle, the period of the population changes to the mature period. Next, the population as a whole generates many

Table 9.4: EFWA algorithm parameter settings.

# of fireworks for 2-D, 10-D and 30-D search	5
# of sparks m	50
# of Gauss mutation sparks,	5
constant parameters	$a = 0.04$ $b = 0.8$
maximum amplitude A_{max}	40
initial and final min. amplitude A_{init}, A_{final}	0.1, 0.05
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

Table 9.5: Statistical test results of the Kruskal-Wallis test and Holm's multiple comparison for the average fitness values of 30 trial runs among 4 methods at the stop condition. ($A \gg B$) and ($A > B$) mean that A is significantly better than B with significance levels of 1% and 5%, respectively. ($A \approx B$) means that there is no significant difference among them.

	2-D	10-D	30-D
F_1	$VEGE \gg DE \approx PSO \gg EFWA$	$EFWA > VEGE \gg PSO \gg DE$	$EFWA \gg VEGE \gg PSO \gg DE$
F_2	$DE \approx VEGE \gg EFWA \approx PSO$	$PSO \approx VEGE \gg DE \approx EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_3	$VEGE > DE \gg PSO \approx EFWA$	$VEGE \gg PSO > DE \approx EFWA$	$VEGE \approx PSO \gg EFWA \gg DE$
F_4	$VEGE \approx DE > EFWA \approx PSO$	$PSO \approx DE \gg VEGE \gg EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_5	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg PSO \gg DE > EFWA$	$PSO \approx EFWA \gg VEGE \gg DE$
F_6	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE > PSO \approx EFWA \gg DE$
F_7	$VEGE \gg DE \approx PSO > EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$	$VEGE \approx DE \approx PSO \gg EFWA$
F_8	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx DE \approx EFWA$
F_9	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \approx PSO \gg EFWA > DE$	$PSO \approx VEGE \gg EFWA \gg DE$
F_{10}	$VEGE \gg DE \gg PSO \approx EFWA$	$VEGE \gg PSO > EFWA \gg DE$	$EFWA \gg PSO > VEGE \gg DE$
F_{11}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$
F_{12}	$VEGE \gg DE > PSO \approx EFWA$	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$
F_{13}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$VEGE \gg DE \gg PSO > EFWA$
F_{14}	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \approx PSO \approx EFWA \gg DE$	$EFWA \approx VEGE \approx PSO \gg DE$
F_{15}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \approx PSO \gg EFWA \gg DE$	$VEGE > EFWA \approx PSO \gg DE$
F_{16}	$EFWA \gg VEGE \approx PSO \approx DE$	$VEGE \approx PSO \gg EFWA \gg DE$	$EFWA \gg VEGE \gg DE \approx PSO$
F_{17}	$VEGE \approx DE \gg PSO \approx EFWA$	$VEGE \approx PSO \gg DE \gg EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$
F_{18}	$VEGE > DE \approx PSO \approx EFWA$	$VEGE \approx PSO \gg DE \approx EFWA$	$VEGE \approx EFWA \gg DE \gg PSO$
F_{19}	$VEGE > DE \gg PSO \approx EFWA$	$VEGE \gg PSO \approx EFWA \gg DE$	$PSO \approx VEGE > EFWA \gg DE$
F_{20}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE > DE \gg EFWA \gg PSO$
F_{21}	$VEGE \gg DE \approx PSO \approx EFWA$	$VEGE \approx EFWA \gg DE \approx PSO$	$EFWA \approx VEGE \gg DE \gg PSO$
F_{22}	$VEGE \gg PSO \approx DE > EFWA$	$VEGE > PSO \approx EFWA \gg DE$	$VEGE \approx EFWA \approx PSO \gg DE$
F_{23}	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg PSO > EFWA \approx DE$	$VEGE \gg PSO \approx EFWA \gg DE$
F_{24}	$VEGE \gg DE \approx PSO > EFWA$	$VEGE \gg DE > PSO \gg EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$
F_{25}	$VEGE \approx PSO \approx DE \approx EFWA$	$VEGE \gg DE \gg PSO \gg EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$
F_{26}	$VEGE \gg PSO \approx DE \gg EFWA$	$VEGE \gg DE \approx PSO \gg EFWA$	$DE > VEGE \gg PSO > EFWA$
F_{27}	$VEGE \approx PSO > DE > EFWA$	$VEGE \gg PSO > DE \approx EFWA$	$VEGE \gg PSO \gg DE \approx EFWA$
F_{28}	$VEGE \gg PSO \approx DE \approx EFWA$	$VEGE \gg DE \gg PSO > EFWA$	$VEGE \gg DE \gg PSO \approx EFWA$

diverse seed individuals which are responsible for exploration through the cooperation among the individuals. Then, a certain number of promising individuals are selected as offspring in the next generation, and they continue to repeat the growth trajectory of their parents. The proposal emphasizes exploitation and exploration alternately to achieve a good balance, rather than focusing on one ability at a given time. This guarantees that an individual has a strong local search ability, but can use generated seed individuals to jump out from a local area to avoid premature convergence. Furthermore, the length of difference vectors changes adaptively according to the convergence of the population, and the difference vectors can keep variable distribution shape information of the population. Overall, the balance between exploitation and exploration can be maintained well, and it changes adaptively with the convergence of population.

Secondly, we would like to point out that the performance of our proposal is mainly influenced by three operations: growth, maturation and selection.

The growth operation controls the local search ability of individuals using two introduced parameters, GR and GC. This operation is designed to generate better offspring in directions which have potential. We adopt a one-to-one growth relationship to achieve the growth operation in this paper, where one-to-one growth means that an individual grows in only one direction at a time. Actually, the growth of plants in nature is diverse, and, for example, it is common in nature to grow multiple branches on a tree trunk. Corresponding to our proposed VEGE, this would mean that multiple directions would be simultaneously explored in growth. Although this would increase fitness cost consumption, it would also deepen the exploration of the local fitness landscape and avoid falling into a local optima. Thus, developing an efficient local search mechanism is the main task of this operation. We will continue to observe the behaviour of plants in nature and implement more promising mechanisms.

The maturation operation is designed for wide search to increase diversity and can ease the pressure of local search by using two introduced parameters, MS and SI. This operation generates diverse seed individuals and spreads them to explore as many unexplored areas as possible. In this paper, we have just roughly simulated the simplest implementation. Actually, plants in nature have multiple ways to produce offspring such as sexual reproduction and asexual reproduction. It is now also possible to achieve hybridization between different species through artificial means. There are many ways for plants to spread their seeds. Thus, how to generate diverse potential seeds and spread them is the main tasks of this operation. We are going to focus on developing new efficient strategies to generate seed individuals and spread seed individuals which are inspired by natural plants.

The last, but also important, operation is the selection operation. Although an individual experiences two periods (growth and maturation), many seed individuals are generated only in the mature period. Determining a rational method for selecting which individuals from the mature period should be retained in the next generation is important. In this paper, we use fitness ranking to select individuals. Although this operation can maintain faster convergence speed, it also has the risk of rapidly losing population diversity. To overcome this limitation, we intend to develop a new adaptive selection operation based on the current optimization situation in our forthcoming works. For example, diversity selection can be appropriately emphasized to avoid over-concentration of the population and stagnation. Or, different selection methods can be used according to the dynamics of population diversity. In short, how to maintain the balance between the three operations to achieve better performance is a topic that needs continuous research. Too much emphasis on one of them may weaken the performance of our proposal, by causing the population to stagnate or fall into local areas and slow convergence.

Thirdly, we discuss the potential of our proposed algorithm. We only simulated the rough survival and reproduction mechanisms of vegetation to propose an optimization framework. In fact, we have not yet thoroughly researched the many plants in nature, for which there must be many known and unknown survival mechanisms. It is a huge challenge to further enhance the performance of our proposed algorithm by introducing more novel and efficient natural mechanisms used by vegetation. Already, we can develop some variants of the currently proposed version, such as an *adaptive version*, an *independent version*, and a *hybrid version*.

Adaptive version: All individuals of our proposed algorithm perform two different periods alternately and generate some seed individuals. In nature, many plants have different growth conditions depending on their living environment. i.e. plants will grow better on fertile land. Different individuals may generate different numbers of seeds even though they are the same species. Inspired by this phenomenon, we can consider their fitness to be analogous to their growth environments. It is possible to develop an adaptive version of VEGE, where each individual has a different generative cycle to reach maturity and generates a different number of seed individuals according to its own fitness. Because an individual with higher fitness may be closer to the global optimum, such an individual should be given a longer growth period and generate more seed individuals.

Independent version: Since offspring in the next generation are selected after seed individuals from all parent individuals are collected, parallel computing cannot be applied to this process. We can design an independent selection mechanism to maintain multiple different small groups, where each group consists of an individual and all seed individuals generated by it. Further, we can introduce an exclusion mechanism to ensure that different small groups do not overlap search and allow them to locate multiple different extremes. We can apply this mechanism to multimodal optimization problems. Although each seed individual is collaboratively generated by multiple individuals, it is easy to implement parallel computing thanks to the independent selection mechanism.

Hybrid version: The proposed algorithm framework is flexible and easy to combine with other EC algorithms. Their different mechanisms or algorithms can replace a part of our VEGE strategies, and the replaced strategy(es) can be applied to the growth period and/or the mature period to obtain the advantages of different EC algorithms. According to our design philosophy, all individuals focus on their exploitation ability in a growth period, while their exploration ability is emphasized during the mature period. Thus, we can embed one or two different EC algorithms into our proposed optimization framework by setting different parameters to achieve two different search abilities. Although the overall framework of our proposal has not changed in this hybrid version, the specific implementation details may change. For example, when we integrate DE or PSO into our framework, we naturally introduce the parameters of the corresponding algorithm. So, these works need to be further refined deeply in the near future.

To analyze the performance of proposal, we apply the Kruskal-Wallis test and Holm's multiple comparison test at the stop condition and compare four EC algorithms. The statistical test results shown in table 6.2 confirm that our proposed VEGE algorithm is effective and demonstrates potential compared to the other three well-known algorithms. Besides, we can see that our proposal performs the best in all algorithms except F_{16} in 2-D where it is second best. As the dimension increases, VEGE is not ranked as the best in F_1 and F_4 in 10-D but is also not ranked the worst. When the dimension is increased to 30, the VEGE performance is further reduced, but overall, it is still the best. It is probably because the number of individuals becomes too small to generate sufficient differential vectors according to the increase of dimension. Suppose the population size is set to PS, the total number of possible differential vectors is $(PS-1)*(PS-2)$. In our experiments, the number of differential vectors that may be used in each generation is 72 because the PS is set to 10. Thus, the number of difference vectors may be sufficient for a low-dimensional space but insufficient for a high-dimensional space.

In addition, the use of greedy selection may further accelerate the loss of diversity in the population. This indicates that a small of population size is not recommended for high-dimensional problems. It is also a good choice to use outstanding individuals from past generations to generate a sufficient number of differential vectors. It is also necessary to further investigate the relationship between spatial dimensions and population size. Due to limitations of space, we cannot draw the convergence curves of all algorithms which confirm our conjecture and indicate that our proposal has faster convergence speed in the early stages in high dimensions.

Finally, we discuss the unique contribution of our proposed VEGE compared to another plant-inspired algorithm, flower pollination algorithm (FPA) [183]. Although both borrow inspiration from plants, the core optimization mechanisms are essentially different. FPA simulates the pollination process of flowers to find the global optimum. Each individual either performs global pollination operation (exploration) or performs local pollination operation (exploitation) to generate only one offspring individual throughout the search process, i.e. one-to-one parent-child relationship. Thus, FPA uses a parameter to balance exploration and exploitation well by controlling the ratio of two operations in every generation. However, VEGE simulates the growth of plants and the propagation of their seeds to find the global optimum. Each individual first tries to grow several times to find locally potential areas (exploitation), then a mature individual generates many seed individuals instead of one, i.e. one-to-many parent-child relationship, to achieve a global search (exploration). Thus, VEGE repeatedly switches these two capabilities along with the convergence process. In short, these two EC algorithms are great different in terms of implementation details and core ideas.

Analysis of VEGE parameter settings

Like other EC algorithms, we also use a fixed population size, PS, to represent a set of all current individuals, where each individual is a candidate solution. Due to the use of differential vectors, the population size should not be less than 3 in our proposed optimization framework.

In addition, we have introduced four new special parameters. Two of the four parameters are random numbers, and only GC and MS need to be set carefully according to the characteristics of the optimization problems. In the following, we describe their goals and roles respectively.

Growth Cycle (GC): Controls the number of local searches. Individuals enter their mature period after they perform local searches. The larger the GC is, the greater the ability for an individual to explore locally. However, too large a GC makes the population fall into local areas easily, whereas populations with too small GC may not be able to search deeply. Thus, appropriate parameter values help the VEGE algorithm to maintain good performance. In our future work, we will investigate the relationship between the settings and convergence periods and summarize the relationship with optimization problem characteristics, e.g. how to set the size of the growth cycle for what kind of characteristics.

Growth Radius (GR): A randomly generated number which controls the radius of the local search of an individual. A random number for GR can increase the randomness of the VEGE algorithm to adapt to different optimization problems rather than a fixed number. If the GC is set to the boundary, it evolves into a random search of the whole search space. Setting GR too large or too small is not conducive to growth. Certainly, the maximum radius should be determined based on the optimization problem. In our evaluation experiments, we set GR in $[-1, 1]$, which means the radius for a local search of each individual does not exceed 1.

Seed Individuals (SI): This is the number of seed individuals generated by each individual during its mature period. The larger the SI is, the more seed individuals are generated. Although many plants in nature may generate thousands of seeds to survive, it is unwise for the VEGE algorithm to generate too many seed individuals because it may slow the convergence of population. However, generating too few seed individuals will prevent an extensive search being achieved, and thus the search cannot jump out from a local area. Therefore, we must further study and observe the generation mechanism of vegetation and implement them into our proposal in a more rational way. We made all individuals equally generate the same number of seed individuals in this paper. In fact, one approach would be to make each individual generate a different number of seed individuals while keeping the same total number of generated seed individuals.

Moving Scaling (MS): A randomly generated number which controls the direction and scale length of the difference vector. In our evaluation experiments, the maximum size of the differential vector does not exceed two times itself. We used this parameter value because seeds are affected by many factors during the propagation process. It is expected to increase the diversity among seed individuals even when they are diffused in the same direction. Naturally, this parameter alone cannot cannot simulate the complex propagation process of natural plants. Thus, we are considering how to improve it to achieve more effective propagation.

Potential and future topics

In our first attempt, we proposed a new optimization framework by extracting some simple mechanisms from the evolution of vegetation, then simulated them repeatedly to find optima. There must be more effective and elaborate mechanisms waiting to be discovered and integrated into our framework to further improve its performance. Here, we list some future topics, although the possibilities are not limited by this list alone.

- Analysis of the effect of the two periods of an individual on the performance. Further, how to properly share the proportion of these two periods.
- Analysis of the relationship between parameter settings and optimization problem characteristics. Develop an adaptive version.
- Extending the algorithm to other application scenarios, such as multimodal optimization, multi-objective optimization, large scale problems, dynamic optimization and multi-objective and many-objective optimization.
- Development of hybrid algorithms to aggregate the advantages of different EC algorithms.

- Analysis of the convergence characteristics of our proposal and the characteristics of individual trajectories.
- Applications of the VEGE algorithm to real world tasks.

9.2 Analysis of Vegetation Evolution

9.2.1 Design of Analytical Experiments

We design a series of controlled experiments and analyze the contribution of each VEGE component and the effect of parameter settings on its performance to understand the VEGE algorithm deeply. We realize that a *growth period* and a *maturity period* are two core operation units of the VEGE from the above description. After initialization is completed, performing a *maturity period* firstly may generate many potential seed individuals, then top *PS* individuals with better fitness are selected and have the opportunity to perform a *growth period.* Compared to the original VEGE, VEGE with inverse order of two periods may have a high possibility of deep search in potential areas, which may speed up convergence in some cases. Thus, the first controlled experiment is designed to investigate the importance of these two operations as well as their order. Subsequently, we investigate the impact of the number of growth cycles in the *growth period*, the number of generated seeds by each individual in the *maturity period*, and population size on performance, respectively.

Experiment 1

The allocation of resources to two core operation units, a *growth period* and a *maturity period*, influence the performance of VEGE directly. Too much emphasis on either of them is not conducive to algorithm, but their balance is the key for efficient algorithms.

The Experiment 1 is designed to analyze the effect of two core operation periods and their implementation sequence. Four VEGE variants are compared to analyze the contributions of two operation periods to their performance in this experiment. All their parameters are set to the same. Here, the below numbers represent four VEGE variations.

1: original VEGE,
2: VEGE with inverse order of two periods,
3: VEGE without a maturity period,
4: VEGE without a growth period.

Experiment 2

The original VEGE controls individuals in a *growth period* to perform local searches. Each individual randomly generates a new offspring individual within a fixed search range, and the winner remains in the next generation. Here, a *growth cycle* plays an important role and controls the number of growth. If it is set large, an individual uses more computational resources for local search; if it is set small, local search does not go deep.

The Experiment 2 is designed to investigate the impact of the *growth cycle* parameter. We set all parameters to the same except the *growth cycle* parameter in this experiment. Here, below numbers represent three VEGE variations.

1: VEGE with 2 *growth cycles*,
2: VEGE with 6 *growth cycles*,
3: VEGE with 10 *growth cycles*.

Experiment 3

Maturity period designed for wide search is other important factor to determine VEGE performance. Each individual in this period generates many seed individuals rather than one-to-one relationship to increase diversity. Top PS individuals among them are selected to survive in the next generation according to their fitness. Thus, the number of generated seed individuals is one of factors that affects exploration ability. If it is set large, many search space may be searched, but its convergence speed slows down. Conversely, inadequate search makes it easy to get into local optimal areas.

The Experiment 3 is designed to investigate the effect of the number of generated seed individuals and expect to find appropriate parameter settings. Here, the below numbers represent three VEGE variations.

1: VEGE generating 2 seed individuals,
2: VEGE generating 6 seed individuals,
3: VEGE generating 10 seed individuals.

Experiment 4

Population size is also an important factor affecting VEGE performance. Although many practitioners want to find some generic rules for setting a population size, there is no exact way to guide setting it reasonably so far. Generally, inappropriate setting of a population size may affect convergence speed and even leads to inability to converge.

The Experiment 4 is designed to investigate the effect of population size and obtain some empirical settings hopefully. Here, the below numbers represent three VEGE variations which parameters are the same except population size.

1: VEGE with *population size* is 5,
2: VEGE with *population size* is 10,
3: VEGE with *population size* is 15.

Four control experiments have been introduced in detail, and each one has different investigative roles. Note that the parameter setting of the original VEGE in all above four experiments are exactly the same to use as a benchmark algorithm and carry out comparative analysis. Its parameter settings are shown in the Table 9.6.

9.2.2 Experimental Evaluations

To evaluate the performance of each VEGE variation, we use 28 benchmark functions from the CEC2013 benchmark test suite [91] in our experiments. They are designed for real parameter single-objective optimization. These landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal.

Table 9.6: VEGE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	10
growth cycle GC	6
growth radius GR	a random number in [-1,1]
total seed individuals for 2-D, 10-D, and 30-D search	60
moving scaling MS	a random number in [-2,2]
stop condition; max. # of fitness evaluations, MAX_{NFC}, for 2-D, 10-D, and 30-D search	1000, 10,000 and 40,000

The parameter setting of the original VEGE as our test baseline is shown in the Table 9.6. We run each benchmark function with 30 trial runs in 3 different dimensional spaces, 2 dimension (2D), 10D and 30D, respectively. The number of fitness calls is used to evaluate convergence against rather than generations for fair evaluations. We apply a statistical test to the fitness values at the stop condition, i.e. the maximum number of fitness calculations, to detect significant differences. The statistical results of four control experiments are shown in the Tables 9.7, 9.8, 9.9 and 9.10.

9.2.3 Discussions

The significant contribution of this paper is to analyze each component of VEGE and the effect of parameter settings on performance. It can help us to have a further understanding of VEGE and tune parameters optimally to adapt to a variety of problems with different characteristics. It can lead to develop more powerful improved versions, i.e. adaptive parameters to apply to multi-modal tasks. Some novel strategies inspired by many plants can be integrated into VEGE to overcome its limitations, e.g. new local search mechanism and seed generation mechanism. Subsequently, detailed analysis through four controlled experiments are described below.

A *growth period* and a *maturity period* are two core operation units of the VEGE, and undertake the roles of exploitation and exploration, respectively. A good balance between these two operation periods can make VEGE have high performance. The Experiment 1 (Table 9.7) analyzes the proportion of their contributions. We can obtain following conclusions from the experimental results.

(1) A *maturity period* operation played a key role, but the *Growth period* operation did not achieve the desired effect. In other words, the exploitation capability of the original VEGE needs to be improved. This is probably because a *maturity period* generates many seed individuals, which increases diverse and potential individuals. Meanwhile, a *growth period* used in the VEGE is only a rough simulation of conceptual plant growth and does not realize the actual plant growth mechanism.

As the dimension increases, the effect of the *maturity period* becomes weak. It may be because the population is not sufficient to provide sufficient difference vectors to generate diverse seed individuals. Actually, plants have several different ways to

Table 9.7: Statistical test results of Friedman test and Holm's multiple comparison for the average fitness values of 30 trial runs among 4 methods. Numbers in the table have the same meaning with those described in the Experiment 1 sub-section. ($A \gg B$) and ($A > B$) mean that A is significantly better than B with significantce levels of 1% and 5%, respectively. ($A \approx B$) means that there is no significant difference among them.

	2D	10D	30D
F_1	$1 \approx 2 \gg 4 \gg 3$	$4 \gg 2 \approx 1 \gg 3$	$2 \approx 1 \approx 3 > 4$
F_2	$2 \approx 1 \approx 4 \gg 3$	$4 \approx 2 \approx 1 \approx 3$	$3 > 4 \gg 1 \approx 2$
F_3	$2 \approx 1 \approx 4 \gg 3$	$2 \approx 1 > 4 \gg 3$	$2 \approx 1 \gg 4 > 3$
F_4	$1 \approx 2 \approx 4 \approx 3$	$2 \approx 1 \approx 4 \approx 3$	$1 \approx 4 \approx 2 \gg 3$
F_5	$4 \gg 1 > 2 \gg 3$	$4 \gg 1 \approx 2 \gg 3$	$4 \gg 2 \approx 1 \gg 3$
F_6	$4 \approx 1 \approx 2 \gg 3$	$4 \approx 1 \approx 2 \approx 3$	$1 \approx 3 \approx 2 > 4$
F_7	$4 \gg 2 \approx 1 \gg 3$	$4 \gg 1 \approx 2 \gg 3$	$4 \gg 1 \approx 2 \gg 3$
F_8	$4 > 2 > 1 \gg 3$	$1 \approx 4 \approx 2 \approx 3$	$1 \approx 4 \approx 2 \approx 3$
F_9	$4 \gg 2 \approx 1 \gg 3$	$4 \gg 2 \approx 1 \gg 3$	$2 \approx 1 \gg 4 \gg 3$
F_{10}	$4 \approx 1 \approx 2 \gg 3$	$4 \gg 2 \approx 1 \gg 3$	$3 \gg 1 \approx 2 \gg 4$
F_{11}	$4 > 2 \approx 1 \gg 3$	$2 \approx 1 \approx 4 \gg 3$	$4 \gg 1 \approx 2 \gg 3$
F_{12}	$2 \approx 1 \approx 4 \gg 3$	$1 \approx 2 > 4 \gg 3$	$2 \approx 1 \approx 4 \gg 3$
F_{13}	$2 \approx 1 \approx 4 \gg 3$	$2 \approx 1 \gg 4 \gg 3$	$2 \approx 1 > 4 \gg 3$
F_{14}	$2 \approx 1 \approx 4 \gg 3$	$4 \approx 2 \approx 1 \gg 3$	$2 \approx 1 > 3 \approx 4$
F_{15}	$2 \approx 4 \approx 1 \gg 3$	$2 \approx 1 \gg 3 \gg 4$	$2 \approx 1 \approx 3 \gg 4$
F_{16}	$3 \approx 1 \approx 4 \approx 2$	$3 > 2 \approx 1 \approx 4$	$3 \approx 1 \approx 2 \gg 4$
F_{17}	$2 \approx 4 \approx 1 \gg 3$	$4 > 1 \approx 2 \gg 3$	$4 \gg 1 \approx 2 \gg 3$
F_{18}	$2 \approx 1 \approx 4 \gg 3$	$1 \approx 2 \approx 4 \gg 3$	$1 \approx 2 \approx 4 \gg 3$
F_{19}	$4 \approx 2 \approx 1 \gg 3$	$4 > 1 \approx 2 \gg 3$	$4 \approx 2 \approx 1 \gg 3$
F_{20}	$4 > 2 > 1 \gg 3$	$2 \approx 1 \approx 4 \gg 3$	$1 \approx 2 \approx 4 \gg 3$
F_{21}	$4 \gg 2 > 1 \gg 3$	$3 \approx 2 \approx 1 \approx 4$	$3 \gg 4 \approx 2 \approx 1$
F_{22}	$4 \approx 2 \approx 1 \gg 3$	$4 > 2 \approx 1 \gg 3$	$4 \approx 1 \approx 2 \gg 3$
F_{23}	$4 \approx 1 \approx 2 \gg 3$	$2 \approx 1 \gg 3 > 4$	$1 \approx 2 \gg 3 \gg 4$
F_{24}	$4 > 1 \approx 2 \gg 3$	$2 \approx 4 > 1 \gg 3$	$4 \gg 2 \approx 1 \gg 3$
F_{25}	$1 \approx 2 \approx 4 \gg 3$	$4 \approx 2 \approx 1 \gg 3$	$4 \gg 1 \approx 2 \gg 3$
F_{26}	$4 \approx 2 \approx 1 \gg 3$	$2 \approx 1 \approx 4 \gg 3$	$2 \approx 1 \approx 4 \approx 3$
F_{27}	$2 \approx 1 \approx 4 \gg 3$	$4 \gg 3 \approx 1 \approx 2$	$4 \gg 3 \approx 2 \approx 1$
F_{28}	$2 \approx 1 \approx 4 \gg 3$	$4 \gg 2 \approx 1 \gg 3$	$1 \approx 2 \approx 4 \gg 3$

grow in nature, and there is great potential to obtain inspiration from them. Thus, how to achieve an efficient local search will be one of research directions to improve the VEGE performance.

(2) The combination of two period operations further improve performance in some cases. Because it can own the characteristics of two operations and avoid over-emphasizing either operation not to fall into local areas. The implemented order of two period operations slightly affect the VEGE performance though it is not obvious. It may be due to performing a *maturity period* firstly generates many potential seed individuals, then top *PS* individuals with better fitness are

Table 9.8: Statistical test results of the Friedman test and Holm's multiple comparison for the average fitness values of 30 trial runs among 3 methods. Numbers in the table have same meaning with those described in the Experiment 2 sub-section. The used symbols have the same meaning with the Table 9.7.

	2D	10D	30D
F_1	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$	$1 \approx 3 \approx 2$
F_2	$1 \approx 3 \approx 2$	$1 \approx 3 \approx 2$	$3 \approx 2 \gg 1$
F_3	$2 \approx 1 \approx 3$	$1 \gg 2 \approx 3$	$3 > 2 \gg 1$
F_4	$2 > 3 \approx 1$	$1 \approx 2 \approx 3$	$1 > 2 > 3$
F_5	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$	$3 \gg 2 \gg 1$
F_6	$1 \approx 2 \approx 3$	$3 \approx 2 \approx 1$	$2 \approx 3 > 1$
F_7	$1 \gg 2 > 3$	$1 \gg 2 \gg 3$	$1 > 2 \approx 3$
F_8	$1 \gg 2 \approx 3$	$2 \approx 1 \gg 3$	$2 \approx 3 \approx 1$
F_9	$1 \gg 2 \approx 3$	$1 \approx 2 \approx 3$	$2 \approx 3 \approx 1$
F_{10}	$1 > 2 > 3$	$1 \gg 2 \approx 3$	$3 > 2 \gg 1$
F_{11}	$1 \gg 3 \approx 2$	$1 \gg 3 \approx 2$	$3 \approx 1 \approx 2$
F_{12}	$1 \approx 2 \approx 3$	$1 \approx 3 \approx 2$	$1 \approx 2 \approx 3$
F_{13}	$1 \gg 2 \approx 3$	$1 \gg 2 > 3$	$1 \approx 2 \approx 3$
F_{14}	$1 \approx 2 \approx 3$	$1 \approx 2 \approx 3$	$3 \approx 2 \approx 1$
F_{15}	$1 \approx 2 \approx 3$	$1 \approx 2 \approx 3$	$3 \approx 1 \approx 2$
F_{16}	$1 \approx 2 \approx 3$	$3 > 2 \approx 1$	$3 \approx 2 \approx 1$
F_{17}	$1 \approx 3 \approx 2$	$1 > 2 > 3$	$1 \gg 2 \gg 3$
F_{18}	$1 \approx 2 \approx 3$	$1 \approx 2 \approx 3$	$1 > 2 > 3$
F_{19}	$2 \approx 1 > 3$	$1 \approx 2 \approx 3$	$1 > 2 > 3$
F_{20}	$1 \gg 2 \approx 3$	$1 \approx 2 \approx 3$	$1 \approx 2 \approx 3$
F_{21}	$1 \gg 3 \approx 2$	$2 > 3 \gg 1$	$1 > 2 \approx 3$
F_{22}	$1 > 2 \approx 3$	$1 \approx 2 \approx 3$	$1 \approx 2 \approx 3$
F_{23}	$2 \approx 1 \approx 3$	$1 \approx 3 \approx 2$	$1 \approx 2 > 3$
F_{24}	$2 > 1 \gg 3$	$3 \gg 1 > 2$	$1 > 2 \approx 3$
F_{25}	$2 \approx 1 \gg 3$	$1 \gg 2 \approx 3$	$1 > 2 > 3$
F_{26}	$1 \approx 2 \approx 3$	$2 \approx 1 \approx 3$	$1 \gg 3 \approx 2$
F_{27}	$2 > 1 > 3$	$1 > 2 > 3$	$1 \approx 2 \approx 3$
F_{28}	$1 \gg 2 > 3$	$2 > 3 \gg 1$	$3 \approx 2 \gg 1$

selected and have the opportunity to perform a *growth period*, and the remaining seed individuals are eliminated. Compared to performing the *growth period* firstly on randomly individuals used in the original VEGE, VEGE with inverse order of two periods has a high possibility of deep search in potential areas, which may speed up convergence in some cases. However, using fitness only to select seed individuals may also lead to multiple individuals being selected in the same area and thus to premature.

Thus, how to balance the two operations would been also a promising topic. Adaptively assigning the frequency of the two operations may be one of good choices to balance their search ability.

The Experiment 2 (Table 9.8) further analyzes the *growth period* operation.

Table 9.9: Statistical test results of the Friedman test and Holm's multiple comparison for the average fitness values of 30 trial runs among 3 methods. Numbers in the table have same meaning with those described in the Experiment 3 sub-section. The used symbols have the same meaning with the Table 9.7.

	2D	10D	30D
F_1	$2 \approx 1 \approx 3$	$3 \approx 2 \gg 1$	$2 \approx 1 \approx 3$
F_2	$1 \approx 2 \approx 3$	$1 \approx 3 \approx 2$	$1 \gg 2 \approx 3$
F_3	$3 \approx 1 \approx 2$	$3 \approx 2 \approx 1$	$1 \approx 3 \approx 2$
F_4	$2 \approx 1 \approx 3$	$2 \approx 3 \approx 1$	$3 \approx 2 \gg 1$
F_5	$2 > 3 \approx 1$	$2 \approx 3 \gg 1$	$1 \gg 2 \gg 3$
F_6	$2 \approx 3 \gg 1$	$2 \approx 3 \approx 1$	$2 \approx 1 \approx 3$
F_7	$2 \approx 3 \gg 1$	$3 \approx 2 \gg 1$	$3 \approx 2 > 1$
F_8	$3 \approx 2 > 1$	$2 \approx 1 \approx 3$	$2 \approx 3 \approx 1$
F_9	$2 \approx 3 \approx 1$	$3 \approx 2 > 1$	$2 \approx 3 \approx 1$
F_{10}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$1 \gg 2 \gg 3$
F_{11}	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$
F_{12}	$2 \approx 3 \approx 1$	$2 \approx 3 \approx 1$	$3 > 2 \approx 1$
F_{13}	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \gg 1$
F_{14}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$2 \approx 3 \approx 1$
F_{15}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$
F_{16}	$3 \approx 1 \approx 2$	$1 \gg 3 \approx 2$	$1 \approx 2 > 3$
F_{17}	$3 \approx 1 \approx 2$	$3 \approx 2 \gg 1$	$3 > 2 \gg 1$
F_{18}	$3 \approx 2 \approx 1$	$3 \approx 2 \gg 1$	$3 \approx 2 \gg 1$
F_{19}	$3 \approx 2 \approx 1$	$2 \approx 3 \gg 1$	$3 \approx 2 \gg 1$
F_{20}	$3 > 2 \approx 1$	$3 \approx 2 \approx 1$	$2 \approx 3 > 1$
F_{21}	$3 \approx 2 \approx 1$	$1 \approx 3 \approx 2$	$1 \approx 3 \approx 2$
F_{22}	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$	$2 \approx 3 \approx 1$
F_{23}	$2 \approx 3 \approx 1$	$3 \approx 2 > 1$	$2 \approx 3 > 1$
F_{24}	$2 \approx 3 \approx 1$	$3 \approx 2 \gg 1$	$3 \approx 2 \gg 1$
F_{25}	$2 \approx 3 > 1$	$3 > 2 \approx 1$	$3 \approx 2 > 1$
F_{26}	$3 \approx 1 \approx 2$	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$
F_{27}	$3 \approx 2 \approx 1$	$2 \approx 3 \gg 1$	$3 \approx 2 \approx 1$
F_{28}	$3 \approx 2 \approx 1$	$3 \approx 2 \gg 1$	$2 \approx 3 \approx 1$

Initially, it was designed to find more potential individuals within a local area, and its number is mainly determined by the number of local searches in the *growth period* and a search strategy. The first experiment confirmed that the random search strategy used in the *growth period* was not satisfactory. Thus, this experiment was redesigned to confirm the effect of the number of random searches in the *growth period*, a *growth cycle*. The experimental results revealed that the *growth cycle* parameter has an impact on the VEGE performance. Generally, the smaller the *growth cycle* parameter, the better the performance. As the dimension increases, the *growth cycle* also need to increase appropriately. Because a *maturity period* plays a key role in VEGE, its effects may be obscured and cannot be reflected. It also needs further verification in our future work. Anyway, we recommend that the *growth cycle*

Table 9.10: Statistical test results of the Kruskal-Wallis test and Holm's multiple comparison for the average fitness values of 30 trial runs among 3 methods. Numbers in the table have same meaning with those described in the Experiment 4 subsection. The used symbols have the same meaning with the Table 9.7.

	2D	10D	30D
F_1	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$
F_2	$3 \approx 2 \approx 1$	$1 \approx 3 \approx 2$	$1 \gg 2 > 3$
F_3	$1 \approx 3 \approx 2$	$2 \approx 1 \gg 3$	$1 \approx 2 \gg 3$
F_4	$2 \approx 3 \approx 1$	$2 \approx 1 \approx 3$	$1 \gg 2 \approx 3$
F_5	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$
F_6	$3 \approx 2 \approx 1$	$1 > 3 \approx 2$	$2 \approx 1 > 3$
F_7	$2 \gg 1 \gg 3$	$2 > 3 \approx 1$	$2 \approx 3 \approx 1$
F_8	$1 \gg 2 \approx 3$	$2 \approx 1 \approx 3$	$2 \approx 3 \approx 1$
F_9	$1 \gg 2 > 3$	$1 \approx 2 > 3$	$1 \approx 2 \approx 3$
F_{10}	$1 \approx 2 \gg 3$	$1 \gg 2 \gg 3$	$1 \gg 2 \gg 3$
F_{11}	$3 \approx 2 \approx 1$	$2 \approx 3 > 1$	$3 \approx 2 \approx 1$
F_{12}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$2 \approx 3 \approx 1$
F_{13}	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \gg 1$
F_{14}	$2 > 3 \approx 1$	$2 \approx 3 > 1$	$2 \approx 1 \approx 3$
F_{15}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$
F_{16}	$1 \approx 2 \approx 3$	$1 \approx 3 \approx 2$	$2 \approx 1 \approx 3$
F_{17}	$1 > 3 \approx 2$	$1 \approx 2 \gg 3$	$2 \approx 1 \gg 3$
F_{18}	$1 \approx 3 \approx 2$	$1 > 2 > 3$	$2 \approx 1 > 3$
F_{19}	$2 \approx 1 \approx 3$	$1 > 2 \gg 3$	$3 \gg 2 \approx 1$
F_{20}	$1 \gg 2 \approx 3$	$2 \approx 1 \approx 3$	$2 \approx 3 \approx 1$
F_{21}	$3 > 2 > 1$	$3 \approx 2 \approx 1$	$1 \approx 2 > 3$
F_{22}	$2 \approx 3 \approx 1$	$3 \approx 2 \approx 1$	$2 \approx 3 \approx 1$
F_{23}	$3 \approx 2 \approx 1$	$3 \approx 2 > 1$	$2 \approx 3 \gg 1$
F_{24}	$2 \approx 3 \approx 1$	$2 \approx 1 \approx 3$	$2 \approx 1 \approx 3$
F_{25}	$3 \approx 2 \approx 1$	$2 \approx 1 \approx 3$	$2 \approx 1 \gg 3$
F_{26}	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$	$3 \approx 2 \approx 1$
F_{27}	$3 \approx 2 \approx 1$	$2 \gg 1 \approx 3$	$1 \approx 2 \approx 3$
F_{28}	$3 \approx 2 \approx 1$	$2 > 3 \gg 1$	$1 \gg 2 \gg 3$

parameter should not be set too large and should be also gradually increased as the dimension increases. Adaptive tuning *growth cycle* may be a promising solution to achieve better performance.

The Experiment 3 (Table 9.9) is to confirm how many generated seed individuals are adequate to obtain a high VEGE performance. It showed that a *maturity period* operation was crucial to the VEGE performance and explore as wide areas as possible by generating many seed individuals. It bears the responsibility of wide search by simulating the propagation of many generated seed individuals. The Experiments 1 and 2 indicate that the seed generation strategy can provide potential seeds to the next cycle. The experimental results in the Table 9.9 showed that generating too few seed individuals may result no potential search and is not recommended though it is not a key factor affecting VEGE performance. Conversely, generating too much seed individuals may create unnecessary waste of fitness calls. Because only a small number of seed individuals can remain in the next generation, and many seed individuals die out during selection competition. Thus, how to improve the utilization of seed individuals would be also one of our future works. In a way, the parameter, *number of seeds* is also related to the population size. Balancing the resource allocation between population size and total generated seed individuals is one of our next works. It is also a potential topic that each individual determines the number of its generated seeds adaptively based on its own optimization process, i.e. fitness, potential, and others.

The Experiment 4 (Table 9.10) is to find some rules for setting a reasonable population size for optimal performance. Population size is a long-lasting topic in EC community, and many researchers tried to propose its general tuning rules. We can roughly conclude that it is proportional to the dimension of tasks from the experimental results. It may be due to the fact that large population size can provide sufficient number of difference vectors to spread the diversity of seed individuals in high dimensional search space. Suppose that the population size is set to N, then it can provide the $(N^2 - N)$ directions of propagation. Thus, it should be increased according to the increase of the task dimensions.

We are going to focus on the relationship between population sizes and the characteristics of tasks and develop a dynamic population strategy combining the task characteristics and optimization processes.

9.3 Improvements to Vegetation Evolution

9.3.1 Mutation Strategy

VEGE observes the growth pattern of vegetation, then abstracts and simulates their survival mechanisms simply to find the global optimum. Although our previous studies have shown that VEGE is effective and promising, it can still be further improved by introducing new strategies inspired by various real plants that use more sophisticated strategies to adapt to their disparate environments. For example, different plants have different growth patterns, such as phototropism and aggregate, and they also reproduce both sexually and asexually. Inspired by these real and effective mechanisms, we propose two strategies to enhance the performance of conventional VEGE. The first mutation strategy introduces different mutation methods to increase diversity and avoid falling into local optima. The aggregate growth strategy is expected to accelerate the growth of non-optimal individuals to more promising areas rather than completely random growth. Here, two proposed strategies are described in detail.

Mutation is an important means of increasing diversity in plants to face various unknown environments. Plant mutations can be roughly divided into two categories: those caused by external factors and those caused by internal factors. External factors, e.g. surroundings and radiation may make individuals change dramatically immediately or even lose some of their abilities, while internal factors, e.g. gene mutation, make a large change in individuals with a small probability. Usually these

Algorithm 16 Two different mutation methods are used in the growth period and the maturity period, respectively. D: dimension; MR: mutation probability; x^j_{min}: lower of the j-th dimension; x^j_{max}: upper of the j-th dimension.

```
1: for $j = 0; j < D; j++$ do
2:     if $rand(0,1) < MR$ then
3:         if a individual, $x$ is in the growth period then
4:             $x^j = rand(x^j_{min}, x^j_{max})$
5:         else
6:             $x^j = x^j + Gaussian(0,1)$
7:         end if
8:     end if
9: end for
```

mutated individuals belong to the same species and can mate with others individuals. Inspired by these observations, a random mutation method is used in the growth period to simulate uncertain external factors and a Gaussian mutation is used during the maturity period to simulate mutations at the gene level. Algorithm 16 outlines the two kinds of mutations in pseudo-code. Note that a standard Gauss mutation is employed in following evaluation experiments, and MR is set to a constant 0.05.

9.3.2 Global-based Growth Strategy

The growth period is originally designed to further improve the convergence accuracy of individuals by motivating them to evolve into promising areas. Based on our previous findings, the random growth strategy used in conventional VEGE does not achieve the desired effect. Thus, a new growth strategy with higher efficiency is needed to replace the original strategy and achieve the above objectives.

Through observations of the growth patterns of real plants, we observe that there are many different growth strategies to ensure the species' survival after a long period of natural selection and evolution. For example, many plants show a tendency to grow towards the sun to receive more light. Additionally, they also tend to be develop in aggregate rather than independent, which is a good defense against unknown risks. Inspired by these observations, we roughly extract the characteristics of their growth and propose a aggregate growth strategy. Corresponding to the VEGE proposal, we set the current best individual as the sun, i.e. the source of attraction, which causes other individuals to converge toward it to achieve accelerated convergence. Thus, the new growth strategy uses Eq. (9.1) to replace Eq. (9.2) used in conventional VEGE for controlling the local growth of individuals. Figure 9.3 demonstrates the difference between the two growth strategies. So far, both proposed strategies have been described in detail and Algorithm 17 shows the generic framework used in conventional VEGE combined with our two proposed strategies.

$$\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x}_{\boldsymbol{i}} + rand(-1,1) \times GR + \omega \times (\boldsymbol{x}_{\boldsymbol{best}} - \boldsymbol{x}_{\boldsymbol{i}}) \tag{9.1}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x}_{\boldsymbol{i}} + rand(-1,1) \times GR \tag{9.2}$$

Algorithm 17 The general framework of VEGE with our two proposed strategies. Our proposal consists of steps 6, 7 and 13.

```
1: Initialize population randomly.
2: Evaluate the population.
3: while The termination condition is not reached do
4:    if individuals are still in the growth period then
5:       for $i = 0; i < PS; i++$ do
6:          the $i$-th individual performs proposed aggregate growth.
7:          Perform a random mutation described in Algorithm 15.
8:          Better offspring replaces its parent, otherwise keep the $i$-th individual.
9:       end for
10:   else
11:      for $i = 0; i < PS; i++$ do
12:         the $i$-th individual performs maturity operation.
13:         Perform the Gaussian mutation described in Algorithm 15.
14:      end for
15:      Select the next generation from a mixed pool consisting of the current population and all generated seed individuals.
16:   end if
17: end while
18: Output the found optima.
```

where, $\boldsymbol{x_i}$ and $\boldsymbol{x_{best}}$ are the i-th and best individual of current population, respectively. GR is a constant used to control the maximum radius of growth. ω is a weight for tuning the intensity of the influence of the optimal individual and we set it to 0.5 in our following evaluation experiments.

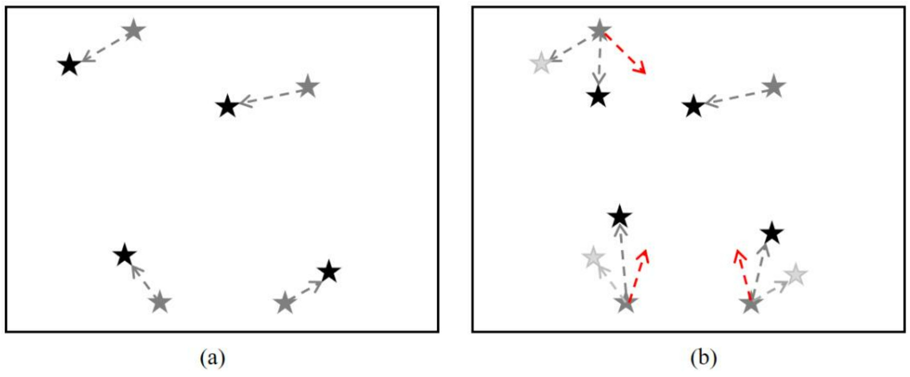


Figure 9.3: (a) is random growth strategy used in conventional VEGE, where an offspring is randomly generated within GR; (b) is our proposed growth strategy, where the red dashed lines indicate the effect of the current best individual in attracting other individuals to achieve aggregation.

9.3.3 Experimental Evaluations

In our evaluation experiments, we use 28 benchmark functions from the CEC2013 suite [91], which contains a series of competitions for solving single-objective optimizations. To evaluate and analyze the performance of the two proposed strategies, we design four VEGE variants. Variant 1, 2, 3, and 4 are, respectively: conventional VEGE; conventional VEGE + mutation strategy; conventional VEGE + aggregate growth strategy; and conventional VEGE + both strategies. We run these four VEGE variants on each function with 30 trial runs on three different dimensions: 2-D, 10-D and 30-D, independently. The parameter settings of VEGE used in the four variants are shown in the Table 9.11.

Table 9.11: VEGE algorithm parameter settings.

population size for 2-D, 10-D, and 30-D search	20
growth cycle GC	5
growth radius GR	a random number in [-1,1]
total seed individuals SI for 2-D, 10-D, and 30-D search	100
moving scaling MS	a random number in [-2,2]
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	1000, 10,000, 40,000

So that the evaluations are fair, we evaluate convergence along with the number of fitness calls rather than generations, and apply the Friedman test and Holm's multiple comparison test at the stop condition, i.e., the maximum number of fitness evaluations, to check if there is a significant difference among the four variants. The results of the statistical tests are shown in the Table 9.12.

9.3.4 Discussions

Discussion on the Mutation Strategy

Gene mutations introduce randomness and uncertainty, and can generate diverse individuals to face complex and changeable environments. Although mutant individuals may become better or poorer, some individuals, which adapt to new natural selection, survive thanks to the large population. Because of the rapid development of technology, humans have been able to intervene and guide mutations, even integrating genes among different species. We can say that mutation is one of the important means for plant populations to survive.

However, in conventional VEGE there is no mutation operation. We therefore introduce two different mutation methods into VEGE to simulate the mutation characteristics of plants in different periods to increase their risk resistance. The random mutation is used in the growth period and does not have any regularity to alter genetic information randomly. We expect that this will prevent it from falling into local optimum areas. Gaussian mutation is used in the maturity period and the greater the magnitude of its difference, the smaller the probability of individual occurrence, which we expect will increase the diversity of the population. A new parameter, MR, is introduced to control the frequency of mutations in our proposal. Although the mutational probability is equal on each dimension, the mutational probability of individuals increases as the number of dimension increases. The experimental results confirm our hypothesis, and the convergence curve indicates that the mutation strategy can jump out of local areas in some cases; however

Table 9.12: Statistical test results of the Friedman test and Holm's multiple comparison test for average fitness values of 30 trial runs among four variants at the stop condition. $A \gg B$ and $A > B$ means that A is significantly better than B with a significance levels of 1% and 5%, respectively. $A \approx B$ means that although A is better than B, there is no significant difference between them. Numbers in the Table have same meaning with that described in above experiments.

	2D	10D	30D
F_1	$4 \approx 3 \gg 1 \approx 2$	$3 \approx 4 \gg 1 \approx 2$	$3 \gg 4 \gg 1 \gg 2$
F_2	$2 \approx 3 \approx 1 \approx 2$	$3 \approx 4 \gg 2 > 1$	$3 \gg 4 \gg 1 \gg 2$
F_3	$3 \approx 4 \approx 1 \approx 2$	$2 > 4 \approx 3 > 1$	$3 \approx 4 \gg 1 \gg 2$
F_4	$2 \approx 1 \approx 4 \approx 3$	$4 \approx 3 \approx 1 \approx 2$	$4 \approx 3 \gg 1 \approx 2$
F_5	$3 \approx 4 \gg 2 \approx 1$	$4 \approx 3 \gg 1 > 2$	$3 \gg 4 \gg 1 \gg 2$
F_6	$3 \approx 4 \gg 2 \approx 1$	$1 \approx 2 \approx 3 \approx 4$	$1 \approx 3 \approx 4 \gg 2$
F_7	$4 \approx 3 \gg 2 \approx 1$	$4 \approx 3 \gg 2 \approx 1$	$4 \gg 3 > 1 \approx 2$
F_8	$3 \approx 4 \gg 1 \approx 2$	$1 \approx 4 \approx 3 \approx 2$	$1 \approx 3 \approx 2 \approx 4$
F_9	$4 \approx 3 \gg 2 \approx 1$	$4 \approx 3 \gg 2 > 1$	$4 \gg 3 \approx 2 > 1$
F_{10}	$3 \approx 4 \gg 2 \approx 1$	$3 \approx 4 \gg 1 \approx 2$	$3 \gg 4 \gg 1 \gg 2$
F_{11}	$4 \approx 3 \gg 2 \approx 1$	$4 \approx 3 \approx 2 \gg 1$	$4 \gg 2 \gg 1 \approx 3$
F_{12}	$4 \approx 3 \gg 2 \approx 1$	$4 > 2 \approx 3 \approx 1$	$4 \gg 3 \approx 1 \approx 2$
F_{13}	$3 \approx 4 > 1 \approx 2$	$2 \approx 1 \gg 4 \approx 3$	$4 \gg 3 \gg 2 \approx 1$
F_{14}	$3 \approx 4 \approx 2 \approx 1$	$2 \gg 4 \approx 1 > 3$	$2 \gg 4 \gg 1 > 3$
F_{15}	$4 \approx 3 \approx 2 \approx 1$	$1 \approx 2 \approx 3 > 4$	$1 \gg 2 > 3 \approx 4$
F_{16}	$2 \approx 1 \approx 4 \approx 3$	$1 \approx 2 \approx 4 \approx 3$	$1 \gg 3 \approx 2 \approx 4$
F_{17}	$2 \approx 4 \approx 3 \approx 1$	$3 \approx 4 \gg 2 \approx 1$	$4 \approx 3 \gg 2 \approx 1$
F_{18}	$2 \approx 3 \approx 4 \approx 1$	$4 \approx 3 \gg 2 \approx 1$	$4 \approx 3 \gg 2 \approx 1$
F_{19}	$3 \approx 4 \gg 2 \approx 1$	$4 \approx 3 \gg 1 \approx 2$	$4 > 3 \gg 1 \approx 2$
F_{20}	$4 \approx 3 \gg 2 \approx 1$	$3 \approx 4 \approx 2 \approx 1$	$3 \approx 4 \gg 2 \approx 1$
F_{21}	$3 \approx 4 \gg 2 \approx 1$	$1 \approx 2 \approx 3 > 4$	$3 \approx 4 \approx 1 \gg 2$
F_{22}	$3 \approx 4 \approx 2 \approx 1$	$2 \gg 4 > 1 \approx 3$	$2 \gg 4 \gg 1 \approx 3$
F_{23}	$3 \approx 4 \gg 2 \approx 1$	$1 \approx 2 \approx 4 \approx 3$	$1 > 2 \gg 3 \approx 4$
F_{24}	$3 \approx 4 \gg 1 \approx 2$	$4 \gg 3 \gg 2 \approx 1$	$4 \gg 3 \approx 2 \approx 1$
F_{25}	$4 \approx 3 > 2 \approx 1$	$3 \approx 4 \gg 2 \approx 1$	$4 \gg 3 \approx 2 \approx 1$
F_{26}	$3 \approx 4 > 2 \approx 1$	$2 \approx 1 \approx 4 \approx 3$	$2 \approx 1 \approx 4 \approx 3$
F_{27}	$4 \approx 3 \approx 2 \approx 1$	$4 \approx 3 \gg 2 \approx 1$	$4 \gg 3 \gg 2 \gg 1$
F_{28}	$3 \approx 4 \gg 2 \approx 1$	$2 \approx 1 \approx 4 \approx 3$	$4 \gg 1 \approx 2 \gg 3$

the acceleration effect is not stable and obvious.

One of the promising topics is how to use mutations to accelerate convergence efficiently. Actually, mutations in real plants are affected by many factors, and even different individuals belonging to the same species have different mutational probabilities. We have assigned the same mutation probability to each dimension without considering their fitness and the characteristics of the optimization problems. Perhaps an adaptive mutation probability could be more powerful and intelligent according to the optimization process; we will pay attention to this aspect in subsequent research.

Discussion on Aggregate Growth Strategy

Even after extensive research of plants, humans have discovered that plants have many almost-magical mechanisms and unknown methods of ensuring their growth. We did not extract these effective mechanisms and simulate them in original VEGE; instead we only used a random growth strategy to simulate plant growth. Here, we propose an aggregate growth strategy to accelerate the convergence of individuals in the growth period.

In a first attempt, the new growth strategy consists of two parts: random search and aggregate attraction; the current best individual is used to attract others and favor them. We introduce a new parameter, ω, to control the weight of the optimal individual’s influence. If ω is set to 0, the proposed growth strategy degenerates to the random growth strategy used in conventional VEGE. As ω increases, so too does the strength of the effect of the optimal individual on the others, i.e., the faster the aggregation speed. The experimental results confirmed that the aggregate growth strategy does accelerate the convergence speed, especially in the early stages.

Although the proposed growth strategy shows a powerful acceleration effect, it still needs be further improved. Here, several open topics are given. (1) the current best individual still grows randomly because the aggregate attraction is 0, and it affects the convergence of other non-optimal individuals. Thus, how to accelerate the growth of the optimal individual reasonably becomes one topic for our future work. Perhaps using historical information is a good choice to guide the growth of the current best individual. (2) ω is the key factor affecting the performance of our proposal. We set it to be a constant in our evaluation experiments. However, this results in us falling into the local optimum areas in some case, e.g. F_{24} to F_{28} on 30-D, because the aggregate attraction is strong. To overcome these limitations, we are going to propose an adaptive version wherein ω is tuned according/ to optimized processes and the evolution information of individuals.

Analysis of Additional Cost

We would like to discuss the advantages confirmed by the two strategies we have proposed. Both strategies are inspired by deep observation of the behaviour of real plants and extraction of their characteristics to simulate plant evolution accurately. Although both strategies introduce new parameters, they do not add any fitness calculation costs. In the case of the mutation strategy, we observed that it jumps out of local areas by increasing diversity - but the acceleration effect is not obvious. Conversely, the aggregate growth strategy can accelerate the convergence speed of the population significantly, but it may may lead to a population falling into local areas as mentioned above. The combination of both strategies, however, can balance their shortcomings and achieve better performance. Furthermore, in any case we can say that our proposals are *low cost, high return* strategies.

We apply the Friedman test and the Holm multiple comparison test among the four variants to check for significant difference. From the results of statistical test, the aggregate growth strategy demonstrated better acceleration than the mutation strategy in lower dimensions. However, the mutation strategy shows stronger effects in some cases with an increase in the number of dimensions. This is because the probability of a high-dimensional individual experiencing mutation becomes higher; the higher the dimension, the more obvious the effect therefore is. For F_{15} and F_{16}, our proposal has not played any role, and even worsened the performance. It may be because there are too many local optima, and these have made our proposal fall into local optima and hindered evolution. This indicates to us that for such problems, increasing randomness may be beneficial in that it allows the population to jump out of the local optima. More detailed explanations require our further analysis, which will allow us to develop more suitable strategies for VEGE.

9.4 Chapter Summary

We proposed a new population-based algorithm which balances exploitation and exploration by simulating the generation and propagation of vegetation. Each individual performs a wide search by inter-individual cooperation after performing multiple local searches through independent competition. The controlled experiments reveal that this mechanism is effective in comparison with other popular EC algorithms.

Secondly, we designed a series of experiments to analyze performance of VEGE and obtain reasonable parameter settings rules. From these experiments, we have found that the *maturity period* operation had a major contribution to VEGE performance, while the effect of the *growth period* operation was not obvious. The results of our statistical test confirmed that the parameter, *number of seeds*, does not cause huge fluctuations in performance, while the *growth cycle* parameter affects VEGE performance; the smaller it is, the better the performance. We do not recommend setting them too large. Finally, we have found that insufficient population size might hinder convergence and it should have been gradually increased as the dimension of the task increases.

Finally, we propose two strategies to improve the performance of VEGE. The controlled experiments confirmed that the mutation strategy increases the diversity of the population and avoids falling into local optima prematurely, and that the aggregate growth strategy can accelerate the convergence speed of non-optimal individuals. Furthermore, the combination of both strategies together can further enhance the performance of VEGE.

In future work, we will continue to introduce novel mechanisms inspired by real plants, such as e.g. a dynamic population mechanism and adaptive parameter tuning, to improving the performance of VEGE and use it to solve many real world applications.

Chapter 10

Discussions and Conclusions

10.1 Main Contribution

Since EC has been applied to various industrial problems and achieved great success, many practitioners try to further improve its performance to solve new and complex problems. How to theoretically enhance the performance of EC algorithm has thus become one of the most promising research topics. The main contribution of this dissertation is to track this topic within three research directions, and to propose multiple strategies to accelerate EC algorithms in each of those direction.

The first direction is to use estimated convergence point(s) as elite individual(s) to accelerate EC algorithms from a new perspective. The basic estimation method uses evolutionary information from two consecutive generations rather than multiple iterations of the population to find the possible area of the global optimum. Not limited to a particular EC algorithm, the basic estimation method can be combined with any EC algorithm without changing the original optimization framework. Although an additional fitness calculation is needed to evaluate the estimated convergence point, when it has high accuracy and is close to the optimum, it can help save a lot of resources and converge the optimum quickly in later generations. When it is not close to the global optimum, the elite may unfortunately not be remarkable - but it is still better than the worst individual which it replaces. We can say that the basic estimation method is a *low risk, high return* strategy and makes an innovative contribution to the EC community. In this dissertation, we extend the basic estimation method to multimodal and multi-objective tasks, discuss the possibility of using it in IEC, and propose new strategies to improve the precision of the estimated convergence point. There are still many valuable research topics in this direction which deserve further attention.

The second direction is to analyze the characteristics of existing EC algorithms and then propose targeted improvement strategies to overcome their original shortcomings and improve performance. In this dissertation, we focus on developing variants of FWA and DE by integrating proposed strategies. Taking FWA as an example, we propose new explosion strategies which use local fitness information to better guide evolution, and integrate competitive strategies into DE to quickly eliminate poorer individuals. Generally, most practitioners are more committed to improving the performance of existing EC algorithms as opposed to developing a new one. Although it is difficult to make a breakthrough work via minor improvements, once they receive sufficient attention, they can also speed up the spread of these EC algorithms into industry. Improving the performance of existing EC algorithms is thus of great significance and an important means to enrich the EC community.

The third direction is to develop a more powerful EC algorithm inspired by the growth of plants and the spread of their seeds. So far, most EC algorithms are inspired by biological evolution, natural phenomena, and human behavior; in each case, they repeatedly simulate the extracted mechanism from to find the global optimum. Few practitioners focus on obtaining inspiration from the widely distributed and very successful plants to propose a new optimization framework. Actually, plant life accounts for by far the most life on Earth, and plants are scattered everywhere; they have evolved a variety of mechanisms to deal with complex environments through long-term evolution. In this dissertation, we roughly summarize the growth patterns of vegetation and propose a new population-based algorithm: VEGE. We hope that this work may attract the attention of other researchers, and various effective strategies can be drawn from natural plants to further improve VEGE performance, or even propose other new EC algorithms.

We use multiple benchmark functions that contain a variety of different characteristics extracted from real-world problems to evaluate the performance of our proposed strategies. The results of statistical tests indicate that all our proposals have shown stronger performance in most cases and the performance is usually more prominent as the dimension increases. Besides, we realize that the estimation method has great potential to solve expensive problems because it can use very little information to estimate the optimal solution and omit a lot of fitness iterative process. Improving existing EC algorithms can help us understand the underlying optimization mechanism well, their advantages and disadvantages, understand their applicable scenarios, and help to design new EC algorithms. Finally, a vertical comparison among various EC algorithms confirm that VEGE has stronger performance thanks to balance exploitation and exploration well by alternately emphasizing two different abilities. In short, the experimental results show that these strategies have more or less enhanced performance but still have room for improvement.

10.2 Limitations

Although our proposal shows better performance from the cost-performance point of view, we still need extra CPU calculation time to achieve our proposed strategies; our strategies require matrix operations be performed, separation methods be executed for multimodal problems, synthetic fireworks be calculated, weights be given to moving vectors, etc. Furthermore, we need to introduce new parameters with our proposal, which increases the complexity of the optimization framework. Unfortunately, we have only considered the fitness cost without considering CPU calculation time and increased complexity in this dissertation. When we apply our proposal to real world applications, we should consider the balance between additional overhead and performance. Especially for some tasks where the fitness cost is low but the CPU calculation cost is high. Since CPU calculation time is also one of important factors that may restrict real world applications of our proposal, it is necessary that we consider multiple factors to more reasonably evaluate the performance of these proposed strategies.

Population diversity is a topic that is often discussed. Fast convergence usually results in loss of population diversity and thus causes a new problem: premature convergence. Based on our research, we found that the estimation method makes it easier to fall into local optima areas when applied to PSO. It may be because an estimated convergence point affects other particles and leads to the wrong convergence area. This also reminds us that we need to develop appropriate acceleration methods for different EC algorithms to avoid premature convergence. Additionally, competitive strategies for DE make the trade-off of fast convergence at the cost of less diversity. Although performance is improved, we still need to get a better balance between population diversity and convergence speed.

VEGE is a new population-based EC algorithm which shows strong optimization performance; however, the underlying working principles are not yet entirely clear. Through the analysis of our experiments, we have confirmed the effectiveness of the two operations, i.e. the *growth period* and the *maturity period*. We came to the realization that some of these operations do not achieve the desired results; for example, the random growth strategy does not assume responsibility for exploitation. Accordingly, the VEGE optimisation framework itself still needs to be further optimized and requires rigorous mathematical theory to support its working scheme.

Another limitation of our work is that we have used experimental evaluations of benchmark functions to assess the theoretical performance of our EC algorithms. As the no-free-lunch theorems [178] claims, it is impossible for an EC algorithm to perform better on all kinds of optimization problems. As a preliminary attempt, we use benchmark functions to evaluate the performance of our proposal. Although they are the extraction and simplification of real world problems that have more complex characteristics, such as, noise, dynamic and others, it is a challenge for our proposal to achieve the desired effects when they are actually applied to real world applications. This part also needs to be further studied in the future.

10.3 Future Work

We followed three research directions to enhance the performance of EC algorithms and reduce fitness consumption costs. The results of the statistical tests confirm our proposed strategies are effective and promising; however, the work is only partially completed in each research direction and there are certainly other ideas worth exploring. Here, we list a few related research topics that should be investigated.

- We should continue to delve further into the basic estimation to improve its accuracy and extend it to more optimization scenarios, e.g. dynamic systems, constraint problems, and real world applications. Realizing that the construction of the moving vectors has a great influence on the accuracy of the estimated convergence point, one of our future work should focus on how to construct moving vectors. Although we propose two ways to improve the accuracy of an estimated convergence point, practitioners can still propose new ways from other perspectives. Furthermore, we would focus on how to use estimated convergence points more efficiently to speed up EC algorithms and proposing different acceleration strategies for EC algorithms by combining their own characteristics. Finally, we summarize some general experiences about the use of estimated convergence points.

- Fitness landscape approximation is an effective approach for reducing costs, especially for complex problems and expensive problems. Most practitioners use such approximations to assist evolution and reduce the number of fitness evaluations, however, little attention is paid to hidden information contained within the fitness approximation model. We will focus on how to extract new knowledge from models and then use them to accelerate the convergence of EC algorithms. In addition, we will try to develop the use of the error between optimization problems and approximation models as a new indicator and use it to evaluate the complexity of optimization problems. We will also analyze the characteristics of different approximation methods, e.g. support vector machine and local polynomial regression, and their applicability. In short, we will focus on mining fitness approximation models and making full use of the information they contain.
- We will try to combine EC algorithms with other technologies to improve their performance. For example, we will integrate the learning ability of neural networks so that EC algorithms can predict unknown individuals and even unexplored areas. Since EC algorithms are almost meta-heuristic algorithms based on random search, introducing deterministic search strategies will be one of our main directions. We will also introduce fuzzy concepts to develop fuzzy fitness to conform to human ideas. In short, we will draw on the strengths and features of other technologies to enrich EC algorithms.
- Since there is a gap between real world problems and benchmark functions, we will apply our proposed strategies to real world problems and catalogue the new problems that may arise. Next we will propose methodologies to solve these new problems so that our algorithms have better adaptability. Practical problems have become an important driving force for theoretical improvement, and they can form an invaluable part of a good cycle for sustainable development.
- We are only discussing the possibility of an estimated convergence point for IEC; the other proposed strategies were not also applied to IEC. One of our future work is to (1) apply the estimation method into real IEC applications; (2) improve these proposed strategies for IEC to alleviate user fatigue. In addition, building high-precision user models and realising the collaboration of multiple models will become our focus.

10.4 Conclusion

Improving the performance of EC algorithms is one of the most heated topics in the field, and we have tracked this topic from three research directions. We proposed multiple strategies to speed up convergence in each research direction. However, although they are not perfect and there are still new remaining problems that need to be solved, we hope that our work can give new inspiration to practitioners and attract new researchers to the field. We also hope that more and more researchers will engage in relevant research and expect that our work may help them a little bit.

Acknowledgements

I would like to thank Professor Hideyuki Takagi, my supervisor, for his priceless guidance and unconditional support during my doctoral research. He not only taught me a lot of professional knowledge on scientific methodology, but more importantly taught me how to become a qualified researcher. I will always remember his teachings and keep his attitude in my future work.

I would like to thank three referees, Professors Hideyuki Takagi, Mario Köppen, and Osamu Maruyama for their time to review this dissertation and their valuable comments for revising this dissertation.

I would like to thank all of laboratory members and my family members. Thanks to their support and understanding, I have been able to conduct my doctoral research smoothly, and let me realize that I should give more concern to others.

I would like to thank doctoral course scholarship of Kyushu University and JSPS Research Fellowships for Young Scientists for their supports to my study in Japan. I should also mention that my doctoral research was supported in part by Grants-in-Aid of the Japan Society for the Promotion of Science (15K00340, 18K11470 and 19J11792).

Jun YU

Fukuoka, Japan
February 28, 2019

Bibliography

[1] K. Abboud and M. Schoenauer. *Surrogate deterministic mutation: Preliminary results*, volume 2310 of *Lecture Notes in Computer Science*. 2002.

[2] M. A. Abido. Optimal power flow using particle swarm optimization. *International Journal of Electrical Power and Energy Systems*, 24(7):563–571, 2002.

[3] J. Adeyemo and D. Stretch. Review of hybrid evolutionary algorithms for optimizing a reservoir. *South African Journal of Chemical Engineering*, 25:22–31, 2018.

[4] M.R. Akbarzadeh-T, M. Davarynejad, and N. Pariz. Adaptive fuzzy fitness granulation for evolutionary optimization. *International Journal of Approximate Reasoning*, 49(3):523–538, 2008.

[5] A. Alajmi and J. Wright. Selecting the most efficient genetic algorithm sets in solving unconstrained building optimization problem. *International Journal of Sustainable Built Environment*, 3(1):18–26, 2014.

[6] M. R. AlRashidi and M. E. El-Hawary. A survey of particle swarm optimization applications in electric power systems. *IEEE Transactions on Evolutionary Computation*, 13(4):913–918, 2009.

[7] L. Altenberg. The evolution of evolvability in genetic programming. In *Advances in genetic programming*, pages 47–74, 1994.

[8] J.D. Anderson. *Computational Fluid Dynamics: The Basics with Applications.* McGraw Hill, 1995.

[9] K. S. Anderson and Y. Hsu. Genetic crossover strategy using an approximation concept. In *the 1999 Congress on Evolutionary Computation*, volume 1, pages 527–533, 1999.

[10] E. Atashpaz-Gargari and C. Lucas. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In *2007 IEEE Congress on Evolutionary Computation*, pages 4661–4667, 2007.

[11] N. Bacanin and M. Tuba. Fireworks algorithm applied to constrained portfolio optimization problem. In *the IEEE Congress on Evolutionary Computation*, pages 1242–1249, 2015.

[12] T. Back, U. Hammel, and H. . Schwefel. Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17, 1997.

[13] D. Bajer, G. Martinovi, and J. Brest. A population initialization method for evolutionary algorithms based on clustering and cauchy deviates. *Expert Systems with Applications*, 60:294–310, 2016.

[14] A.J. Barton, J.J. Valds, and R. Orchard. Neural networks with multiple general neuron models: A hybrid computational intelligence approach using genetic programming. *Neural Networks*, 22(5-6):614–622, 2009.

[15] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*, pages 1–853. Nonlinear Programming: Theory and Algorithms. 2005.

[16] H.G. Beyer and H.P. Schwefel. Evolution strategies -- a comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[17] A. Biswas, S. Dasgupta, S. Das, and A. Abraham. A synergy of differential evolution and bacterial foraging optimization for global optimization. *Neural Network World*, 17(6):607–626, 2007.

[18] T. Blickle and L. Thiele. A comparison of selection schemes used in evolutionary algorithms. *Evolutionary computation*, 4(4):361–394, 1996.

[19] F. Boschetti and L. Moresi. Compasion between interactive (subjective) and traditional (numerical) inversion by genetic algorithms. In *Congress on Evolutionary Computation*, pages 522–528, La Jolla, CA, USA, July 2000.

[20] F. Boschetti, L. Moresi, and K. Covin. Interactive inversion in geological applications. In *International Conference on Knowledge-based Intelligent Information Engineering Systems*, pages 276–279, Aug. 1999.

[21] H. A. Bouarara, R. M. Hamou, A. Amine, and A. Rahmani. A fireworks algorithm for modern web information retrieval with visual results mining. *International Journal of Swarm Intelligence Research*, 6(3):1–23, 2015.

[22] M. Brad and D. Goldberg. Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9:193212, 1995.

[23] P. Bradshaw, G. A. Mizner, and K. Unsworth. Calculation of compressible turbulent boundary layers on straight-tapered swept wings. *AIAA Journal*, 14(3):399–400, 1976.

[24] J. Brest, A. Zamuda, B. Bskovc, M. S. Macec, and V. umer. Dynamic optimization using self-adaptive differential evolution. In *the IEEE Congress on Evolutionary Computation*, pages 415–422, 2009.

[25] A. Buzdalova and M. Buzdalov. Adaptive selection of helper-objectives with reinforcement learning. In *11th International Conference on Machine Learning and Applications*, pages 66–67, 2012.

[26] C. Caldwell and V.S. Johnston. Tracking a criminal suspect through "face-space" with a genetic algorithm. In *the Fourth International Conference on Genetic Algorithm*, pages 416–421, San Diego, California, USA, July 1991.

[27] C. Candan, A. Goffon, F. Lardeux, and F. Saubion. Adaptive selection of helper-objectives with reinforcement learning. In *the 14th International Conference on Genetic and Evolutionary Computation*, pages 1253–1260, 2012.

[28] B. Carse, T.C. Fogarty, and A. Munro. Evolving fuzzy rule based controllers using genetic algorithms. *Fuzzy Sets and Systems*, 80(3):273–293, 1996.

[29] A. L. Cauchy. Méthode générale pour la résolution des systémesd' équations simultanées. *Comptes Rendus Hebd. Séances Acad. Sci*, 25:536–538, 1847.

[30] P. Chakraborty, G.G. Roy, S. Das, D. Jain, and A. Abraham. An improved harmony search algorithm with differential mutation operator. *Fundamenta Informaticae*, 95(4):401–426, 2009.

[31] A. Charnes and C. E. Lemke. Computational problems of linear programing. In *the 1952 ACM National Meeting (Pittsburgh)*, pages 97–98, 1952.

[32] A. Chaudhuri and K. De. Fuzzy genetic heuristic for university course timetable problem. *International Journal of Advances in Soft Computing and its Applications*, 2(1):100–123, 2010.

[33] S.B. Cho. A human-oriented interface for emotional media manipulation with an a-life technology. *Three Dimensional Images*, 13(3):71–76, 1999.

[34] C. Darwin. *On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life.* 1859.

[35] S. Das, A. Abraham, and A. Konar. Automatic clustering using an improved differential evolution algorithm. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, 38(1):218–237, 2008.

[36] S. Das, S. S. Mullick, and P. N. Suganthan. Recent advances in differential evolution-an updated survey. *Swarm and Evolutionary Computation*, 27:1–30, 2016.

[37] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):4–31, 2011.

[38] W. David. An overview of schema theory. *Neural and Evolutionary Computing*, pages 1–17, 2014.

[39] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2014.

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

[41] K. Dervis and B. Bahriye. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, 39(3):459–471, 2007.

[42] K. Ding, Y. Chen, Y. Wang, and Y. Tan. Regional seismic waveform inversion using swarm intelligence algorithms. In *the IEEE Congress on Evolutionary Computation*, pages 1235–1241, 2015.

[43] W. Du and B. Li. Multi-strategy ensemble particle swarm optimization for dynamic optimization. *Information Sciences*, 178(15):3096–3109, 2008.

[44] N. Duvvuru and K.S. Swarup. A hybrid interior point assisted differential evolution algorithm for economic dispatch. *IEEE Transactions on Power Systems*, 26(2):541–549, 2011.

[45] R. C. Eberhart and Y. Shi. Particle swarm optimization: Developments, applications and resources. In *the IEEE Conference on Evolutionary Computation*, volume 1, pages 81–86, 2001.

[46] A.E Eiben and S.K. Smit. Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, 1(1):19–31, 2011.

[47] W. Fang, J. Sun, Z. . Xie, and W. . Xu. Convergence analysis of quantum-behaved particle swarm optimization algorithm and study on its control parameter. *Wuli Xuebao/Acta Physica Sinica*, 59(6):3686–3694, 2010.

[48] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[49] J. W. Fristrom, M. A. Yund, and B. Judd. Genetic programming for development in drosophila. *Critical reviews in biochemistry and molecular biology*, 1(4):537–570, 1973.

[50] Z. Gaing. Particle swarm optimization to solving the economic dispatch considering the generator constraints. *IEEE Transactions on Power Systems*, 18(3):1187–1195, 2003.

[51] Z. Gaing. A particle swarm optimization approach for optimum design of PID controller in avr system. *IEEE Transactions on Energy Conversion*, 19(2):384–391, 2004.

[52] A. H. Gandomi and A. H. Alavi. Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation*, 17(12):4831–4845, 2012.

[53] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *the Second International Conference on Genetic algorithms and their application*, pages 41–49, Hillsdale, NJ, USA, July 1987.

[54] M. Graff and R. Poli. Practical performance models of algorithms in evolutionary program induction and other domains. *Artificial Intelligence*, 174(15):1254–1276, 2010.

[55] D. E. Grierson and W. H. Pak. Optimal sizing, geometrical and topological design using a genetic algorithm. *Structural Optimization*, 6(3):151–159, 1993.

[56] F. Gu, K. Li, L. Yang, and Y. Chen. Combinatorial optimization of multi-agent differential evolution algorithm. *Open Cybernetics and Systemics Journal*, 8:1022–1026, 2014.

[57] R. Günter. Finite markov chain results in evolutionary computation: A tour d'horizon. *Fundamenta Informaticae*, 35(1-4):67–89, 1998.

[58] R. T. Haftka, E. P. Scott, and J. R. Cruz. Optimization and experiments: A survey. *Applied Mechanics Reviews*, 51(7):435–448, 1998.

[59] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *the IEEE Conference on Evolutionary Computation*, pages 312–317, 1996.

[60] G. Harik. Finding multimodal solutions using restricted tournament selection. In *the 6th International Conference on Genetic Algorithms*, pages 24–31, San Francisco, CA, USA, July 1995.

[61] M. Herdy. Evolutionary optimization based on subjective selection - evolving blends of coffee. In *5th European Congress on Intelligent Techniques and Soft Computing*, pages 640–644, Aachen, Germany, 1997.

[62] J. H. Holland. Outline for a logical theory of adaptive systems. *J. Assoc. Comput. Mach.*, 3:297–314, 1962.

[63] J. H. Holland. Genetic algorithms. *Scientific American*, 267(1):66–72, 1992.

[64] Y. Hong, L. E. E. Hungu, and M. Tahk. Acceleration of the convergence speed of evolutionary algorithms using multi-layer neural networks. *Engineering Optimization*, 35(1):91–102, 2003.

[65] X. Hu, R. C. Eberhart, and Y. Shi. Particle swarm with extended memory for multiobjective optimization. In *the IEEE Swarm Intelligence Symposium*, pages 193–197, 2013.

[66] De J. and Kenneth A. An analysis of the behavior of a class of genetic adaptive systems. In *PhD thesis, University of Michigan*, Ann Arbor, MI, USA, 1975.

[67] S.S. Jadon, R. Tiwari, H. Sharma, and J.C. Bansal. Hybrid artificial bee colony algorithm with differential evolution. *Applied Soft Computing Journal*, 58:11–24, 2017.

[68] H. Jain and K. Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622, 2014.

[69] X. Jie and X. Deyun. New metropolis coefficients of particle swarm optimization. In *Chinese Control and Decision Conference*, pages 3518–3521, 2008.

[70] N. Jin and Y. Rahmat-Samii. Advances in particle swarm optimization for antenna designs: Real-number, binary, single-objective and multiobjective implementations. *IEEE Transactions on Antennas and Propagation*, 55(3I):556–567, 2007.

[71] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Comput.*, 9(1):3–12, 1976.

[72] Y. Jin, M. Olhofer, and B. Sendhoff. Managing approximate models in evolutionary aerodynamic design optimization. In *the IEEE Conference on Evolutionary Computation*, volume 1, pages 592–599, 2001.

[73] L. Jongsoo and H. Prabhat. Parallel genetic algorithm implementation in multidisciplinary rotor blade design. *Journal of Aircraft*, 33(5):962–969, 1996.

[74] M. Kandil, S. Shahin, A. Atiya, and M. Fayek. Evolutionary optimization of neural networks for fire recognition. In *the International Conference on Computer Engineering and Systems*, pages 431–435, 2006.

[75] S. Karl. Interactive evolution of dynamical systems. In *the First European Conference on Artificial Life*, pages 171–178, Paris, France, Dec. 1992.

[76] B. Kazimipour, X. Li, and A. K. Qin. A review of population initialization techniques for evolutionary algorithms. In *the 2014 IEEE Congress on Evolutionary Computation*, pages 2585–2592, 2014.

[77] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[78] D. J. Kenneth, F. David, and H.P. Schwefel. *A history of evolutionary computation*, pages A2.3:1–12. 1997.

[79] H.S. Kim and S.B. Cho. Application of interactive genetic algorithm to fashion design. *Engineering Applications of Artificial Intelligence*, 13(6):635–644, 2000.

[80] H.S. Kim and S.B. Cho. An efficient genetic algorithm with less fitness evaluation by clustering. In *the IEEE Conference on Evolutionary Computation*, volume 2, pages 887–894, 2001.

[81] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[82] J. Klockgether and H.P. Schwefel. Two-phase nozzle and hollow core jet experiments. *11th Symp. on Engineering Aspects of Magnetohydrodynamics*, pages 141–148, 1970.

[83] A. Kosorukoff. Human based genetic algorithm. In *the IEEE International Conference on Systems, Man and Cybernetics*, volume 5, pages 3464–3469, 2001.

[84] J. R. Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2):87–112, 1994.

[85] M. Kronfeld and A. Zell. Towards scalability in niching methods. In *2010 IEEE Congress on Evolutionary Computation*, pages 1–8, Barcelona, Spain, July 2010.

[86] J. Li and Y. Tan. Loser-out tournament-based fireworks algorithm for multimodal function optimization. *IEEE Transactions on Evolutionary Computation*, 22(5):679–691, 2018.

[87] J. Li, S. Zheng, and Y. Tan. The effect of information utilization: Introducing a novel guiding spark in the fireworks algorithm. volume 21, pages 153–166, 2017.

[88] J. P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson. A species conserving genetic algorithm for multimodal function optimization. *Evol. Comput.*, 10(3):207–234, 2002.

[89] J. Z. Li and Y. Tan. The effect of information utilization: Introducing a novel guiding spark in the fireworks algorithm. *IEEE Transactions on Evolutionary Computation*, 21(1):153–166, 2017.

[90] X. Li. A multimodal particle swarm optimizer based on fitness euclidean-distance ratio. In *Genetic and Evolutionary Computation Conference*, pages 78–85, 2007.

[91] J.J. Liang, B.Y. Qu, P.N. Suganthan, and A. G. Hernández-Díaz. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. 2013.

[92] T. W. Liao. Two hybrid differential evolution algorithms for engineering design optimization. *Applied Soft Computing Journal*, 10(4):1188–1199, 2010.

[93] Q. Ling, G. Wu, Y. Yang, Z, and Q.P. Wang. Crowding clustering genetic algorithm for multimodal function optimization. *Appl. Soft Comput.*, 8(1):88–95, 2008.

[94] H. Liu, Z. Cai, and Y. Wang. Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Applied Soft Computing Journal*, 10(2):629–640, 2010.

[95] J. Liu and J. Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9(6):448–462, 2005.

[96] T. Mansouri, A.Z. Ravasan, and M.R. Gholamian. A novel hybrid algorithm based on k-means and evolutionary computations for real time clustering. *International Journal of Data Warehousing and Mining*, 10(3):1–14, 2014.

[97] R. Mendes and A. S. Mohais. Dynde: A differential evolution for dynamic optimization problems. In *the IEEE Congress on Evolutionary Computation*, volume 3, pages 2808–2815, 2005.

[98] E. Mezura-Montes, M. E. Miranda-Varela, and R. Del Carmen Gmez-Ramn. Differential evolution in constrained numerical optimization: An empirical study. *Information Sciences*, 180(22):4223–4262, 2010.

[99] E. Mezura-Montes, M. Reyes-Sierra, and C. A. Coello. *Multi-objective optimization using differential evolution: A survey of the state-of-the-art*, volume 143 of *Studies in Computational Intelligence*. 2008.

[100] A. Milani. Online genetic algorithms. *International Journal of Information Theories and Applications*, 11:20–28, 2004.

[101] T. Morimoto, J. De Baerdemaeker, and Y. Hashimoto. An intelligent approach for optimal control of fruit-storage process using neural networks and genetic algorithms. *Computers and Electronics in Agriculture*, 18(2-3):205–224, 1997.

[102] M. A. Muñoz, Y. Sun, M. Kirley, and S. K. Halgamuge. Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges. *Information Sciences*, 317:224–245, 2015.

[103] N. Murata, R. Nishii, H. Takagi, and Y. Pei. Analytical estimation of the convergence point of populations. In *2015 IEEE Congress on Evolutionary Computation*, pages 2619–2624, 2015.

[104] J. Mdar, J. Abonyi, and F. Szeifert. Interactive particle swarm optimization. In *5th International Conference on Intelligent Systems Design and Applications*, volume 2005, pages 314–319, 2005.

[105] B. Naudts and L. Kallel. A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 4(1):1–15, 2000.

[106] H. Nishino, H. Takagi, S.B. Cho, and K. Utsumiya. A 3D modeling system for creative design. In *The 15th International Conference on Information Networking*, pages 479–486, Beppu, Japan, Jan. 2001.

[107] H. Nishino, H. Takagi, and K. Utsumiya. A digital prototyping system for designing novel 3d geometries. In *6th International Conference on Virtual Systems and MultiMedia*, pages 473–482, Gifu, Japan, Oct. 2000.

[108] G. Onwubolu and D. Davendra. Scheduling flow shops using differential evolution algorithm. *European Journal of Operational Research*, 171(2):674–692, 2006.

[109] R. Östermark. Scalability of the genetic hybrid algorithm on a parallel supercomputer. *Kybernetes*, 37(9-10):1492–1507, 2008.

[110] L. D. Pagliarini, M. Ariel, L. Filippo, and H. Henrik. ALife meets web: Lessons learned. In *The First International Conference on Virtual Worlds*, pages 156–167, Berlin, Heidelberg, 1998.

[111] E.I. Papageorgiou and P.P. Groumpos. A new hybrid method using evolutionary algorithms to train fuzzy cognitive maps. *Applied Soft Computing Journal*, 5(4):409–431, 2005.

[112] R. S. Parpinelli and H. S. Lopes. New inspirations in swarm intelligence: A survey. *International Journal of Bio-Inspired Computation*, 3(1):1–16, 2011.

[113] K. M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems*, 22(3):52–67, 2002.

[114] Y. Pei and H. Takagi. Fourier analysis of the fitness landscape for evolutionary search acceleration. In *the IEEE Congress on Evolutionary Computation*, pages 1–7, 2012.

[115] Y. Pei and H. Takagi. Accelerating IEC and EC searches with elite obtained by dimensionality reduction in regression spaces. *Evolutionary Intelligence*, 6(1):27–40, 2013.

[116] Y. Pei and H. Takagi. Triple and quadruple comparison-based interactive differential evolution and differential evolution. In *the 12th ACM Workshop on Foundations of Genetic Algorithms*, pages 173–182, 2013.

[117] H. Pekdemir and H. R. Topcuoglu. Enhancing fireworks algorithms for dynamic optimization problems. In *the IEEE Congress on Evolutionary Computation*, pages 4045–4052, 2016.

[118] E. Pellerin, L. Pigeon, and S. Delisle. Self-adaptive parameters in genetic algorithms. In *The International Society for Optical Engineering*, pages 53–64, 2004.

[119] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *the IEEE International Conference on Evolutionary Computation*, pages 798–803, Nagoya, Japan, May 1996.

[120] C.A. Pea-Reyes and M. Sipper. Fuzzy CoCo: A cooperative-coevolutionary approach to fuzzy modeling. *IEEE Transactions on Fuzzy Systems*, 9(5):727–737, 2001.

[121] S. Pierret and R. A. Van den Braembussche. Turbomachinery blade design using a navier-stokes solver and artificial neural network. *Journal of Turbomachinery*, 121(2):326–332, 1999.

[122] M. Plutowski and H. White. Selecting concise training sets from clean data. *IEEE Transactions on Neural Networks*, 4(2):305–318, 1993.

[123] K. Price. Differential evolution: a fast and simple numerical optimizer. In *Biennial Conference of the North American Fuzzy Information Processing Society - NAFIPS*, pages 524–527, 1996.

[124] K. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution A Practical Approach to Global Optimization*. Springer. 2005.

[125] S.N. Qasem and S.M. Shamsuddin. Memetic elitist pareto differential evolution algorithm based radial basis function networks for classification problems. *Applied Soft Computing Journal*, 11(8):5565–5581, 2011.

[126] B. Qian, L. Wang, D.X. Huang, and X. Wang. An effective hybrid DE-based algorithm for flow shop scheduling with limited buffers. *International Journal of Production Research*, 47(1):1–24, 2009.

[127] A. K. Qin and P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1785–1791, 2005.

[128] B. Y. Qu, J. J. Liang, P. N. Suganthan, and Q. Chen. Problem definitions and evaluation criteria for the CEC 2015 competition on single objective multi-niche optimization. 2015.

[129] B. Y. Qu, P. N. Suganthan, and S. Das. A distance-based locally informed particle swarm model for multimodal optimization. *IEEE Transactions on Evolutionary Computation*, 17(3):387–402, 2013.

[130] B. Y. Qu, P. N. Suganthan, and J. J. Liang. Differential evolution with neighborhood mutation for multimodal optimization. *IEEE Transactions on Evolutionary Computation*, 16(5):601–614, 2012.

[131] A. Rahmani, A. Amine, R. M. Hamou, M. E. Rahmani, and H. A. Bouarara. Privacy preserving through fireworks algorithm based model for image perturbation in big data. *International Journal of Swarm Intelligence Research*, 6(3):41–58, 2015.

[132] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama. Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation*, 12(1):64–79, 2008.

[133] R.S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama. A novel population initialization method for accelerating evolutionary algorithms. *Computers and Mathematics with Applications*, 53(10):1605–1614, 2007.

[134] K. Rasheed and H. Hirsh. Informed operators: Speeding up genetic-algorithm-based design optimization using reduced models. In *the Genetic and Evolutionary Computation Conference*, pages 628–635, 2000.

[135] A. Ratle. Optimal sampling strategies for learning a fitness model. In *the 1999 Congress on Evolutionary Computation*, volume 3, pages 2078–2085, 1999.

[136] Z. Ren, J. Wang, and Y. Gao. The global convergence analysis of particle swarm optimization algorithm based on markov chain. *Kongzhi Lilun Yu Yingyong/Control Theory and Applications*, 28(4):462–466, 2011.

[137] D. Richard. *The Blind Watchmaker.* Essex: longman, 1986.

[138] J. Robinson, S. Sinton, and Y. Rahmat-Samii. Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In *IEEE Antennas and Propagation Society International Symposium*, pages 314–317, 2002.

[139] T. Robi and B. Filipi. Demo: Differential evolution for multiobjective optimization. In *Lecture Notes in Computer Science*, volume 3410, pages 520–533, 2005.

[140] P. Rocca, G. Oliveri, and A. Massa. Differential evolution as applied to electromagnetics. *IEEE Antennas and Propagation Magazine*, 53(1):38–49, 2011.

[141] A. Sarangi, R.K. Mahapatra, and S.P. Panigrahi. DEPSO and PSO-QI in digital filter design. *Expert Systems with Applications*, 38(9):10966–10973, 2011.

[142] E.D. Schulze, E. Beck, and M.H. Klaus. *Plant Ecology.* Springer. 2005.

[143] P. S. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni. Particle swarm and ant colony algorithms hybridized for improved continuous optimization. *Applied Mathematics and Computation*, 188(1):129–142, 2007.

[144] Y. Shi and R. C. Eberhart. Fuzzy adaptive particle swarm optimization. In *the IEEE Conference on Evolutionary Computation*, volume 1, pages 101–106, 2001.

[145] K. Sims. Artificial evolution for computer graphics. In *the 18th Annual Conference on Computer Graphics and Interactive Techniques*, pages 319–328, 1991.

[146] K. Sims. Evolving 3d morphology and behavior by competition. *Artificial Life*, 1(4):353–372, 1994.

[147] S.K. Smit and A Eiben. Comparing parameter tuning methods for evolutionary algorithms. In *the IEEE Congress on Evolutionary Computation*, pages 399–406, 2009.

[148] R. E. Smith, B. A. Dike, and S. A. Stegmann. Fitness inheritance in genetic algorithms. In *the ACM Symposium on Applied Computing*, pages 345–350, 1995.

[149] K. Srinivas, N. Somanath, and D. Joel. Composite sandwich structure optimization with application to satellite components. *AIAA Journal*, 34(3):614–621, 1996.

[150] M. Srinivas and L. M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6):17–26, 1994.

[151] K. O. Stanley and R. Miikkulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

[152] R. Storn and K. Price. Differential evolution -- a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[153] P.N. Suganthan, N. Hansen, J. Liang, K. Deb, Y.P. Chen, A. Auger, and S. . Tiwari. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. 2005.

[154] B. H. Sumida, A. I. Houston, J. M. McNamara, and W. D. Hamilton. Genetic algorithms and evolution. *Journal of theoretical biology*, 147(1):59–84, 1990.

[155] H. Takagi. Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. *the IEEE*, 89(9):1275–1296, 2001.

[156] H. Takagi. Three research directions of interactive evolutionary computation. In *18th Online World Conference on Soft Computing in Industrial Applications*, pages 3–5, 2019.

[157] H. Takagi, S.B. Cho, and T. Noda. Evaluation of an IGA-based image retrieval system using wavelet coefficients. In *IEEE International Conference on Fuzzy Systems*, pages 1775–1780, Seoul, Korea, Aug. 1999.

[158] H. Takagi and M. Ohsaki. Interactive evolutionary computation-based hearing-aid fitting. *IEEE Transaction on Evolutionary Computation*, 11(3):414–427, 2007.

[159] H. Takagi and D. Pallez. Paired comparison-based interactive differential evolution. In *2009 World Congress on Nature and Biologically Inspired Computing*, pages 475–480, 2009.

[160] R. R. Tan. Hybrid evolutionary computation for the development of pollution prevention and control strategies. *Journal of Cleaner Production*, 15(10):902–906, 2007.

[161] Y. Tan and Y.C. Zhu. Fireworks algorithm for optimization. In *the First international conference on Advances in Swarm Intelligence*, pages 355–364, 2010.

[162] B. Thomas and Frank H. Extended selection mechanisms in genetic algorithms. In *the Fourth International Conference on Genetic Algorithms*, pages 92–99, 1991.

[163] R. Thomsen. Multimodal optimization using crowding-based differential evolution. In *the 2004 Congress on Evolutionary Computation*, volume 2, pages 1382–1389, 2004.

[164] H. R. Tizhoosh. Opposition-based learning: A new scheme for machine intelligence. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, volume 1, pages 695–701, 2005.

[165] A. Trivedi, D. Srinivasan, S. Biswas, and T. Reindl. A genetic algorithm - differential evolution based hybrid framework: Case study on unit commitment scheduling problem. *Information Sciences*, 354:275–300, 2016.

[166] M. Tuba, N. Bacanin, and A. Alihodzic. Multilevel image thresholding by fireworks algorithm. In *25th International Conference Radioelektronika*, pages 326–330, 2015.

[167] M. Tuba, N. Bacanin, and M. Beko. Fireworks algorithm for rfid network planning problem. In *25th International Conference Radioelektronika*, pages 440–444, 2015.

[168] K. Vaisakh and A. Srinivasa Reddy. MSFLA/GHS/SFLA-GHS/SDE algorithms for economic dispatch problem considering multiple fuels and valve point loadings. *Applied Soft Computing Journal*, 13(11):4281–4291, 2013.

[169] J. H. Van Sickel, K. Y. Lee, and J. S. Heo. Differential evolution and its applications to power plant control. In *International Conference on Intelligent Systems Applications to Power Systems*, pages 560–565, 2007.

[170] D. F. Votaw. Methods of solving some personnel-classification problems. *Psychometrika*, 17(3):255–266, 1952.

[171] V. Vucic and H H. Lund. Self-evolving arts -- organisms versus fetishes. *The Hungarian Journal of Modern Art*, 104:69–79, 1997.

[172] K. Walczak. Hybrid differential evolution with covariance matrix adaptation for digital filter design. In *the IEEE Symposium on Differential Evolution*, pages 1–7, 2011.

[173] L. Wang, Q.K. Pan, P.N. Suganthan, W.H. Wang, and Y.M Wang. A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems. *Computers and Operations Research*, 37(3):509–520, 2010.

[174] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen. Feature selection based on rough sets and particle swarm optimization. *Pattern Recognition Letters*, 28(4):459–471, 2007.

[175] D. Whitley and A. M. Sutton. *Genetic algorithms - a survey of models and methods*, volume 2-4 of *Handbook of Natural Computing*, pages 637–671. 2012.

[176] W. Wieczorek and Z.J. Czech. Selection schemes in evolutionary algorithms. In *Intelligent Information Systems 2002*, pages 185–194, 2002.

[177] P. Wolfe. Recent developments in nonlinear programming. 3(C):155–187, 1962.

[178] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[179] J. Wu and S. Feng. Improved biogeography-based optimization for the traveling salesman problem. In *the 2nd IEEE International Conference on Computational Intelligence and Applications*, volume 2017-January, 2017.

[180] H. Xiaohui and R. Eberhart. Multiobjective optimization using dynamic neighborhood particle swarm optimization. In *the IEEE Congress on Evolutionary Computation*, volume 2, pages 1677–1681, 2002.

[181] G. Xu and G. Yu. On convergence analysis of particle swarm optimization algorithm. *Journal of Computational and Applied Mathematics*, 333:65–73, 2018.

[182] S. Yang and C. Li. A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. *IEEE Transactions on Evolutionary Computation*, 14(6):959–974, 2010.

[183] X.S. Yang. *Flower pollination algorithm for global optimization*, volume 7445 LNCS of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2012.

[184] X.S. Yang and S. Deb. Cuckoo search via lévy flights. In *2009 World Congress on Nature and Biologically Inspired Computing*, pages 210–214, 2009.

[185] X.S. Yang and A. H. Gandomi. Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations*, 29(5):464–483, 2012.

[186] Z. Yang, K. Tang, and X. Yao. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15):2985–2999, 2008.

[187] J. Yu, YH. Li, Y. Pei, and H. Takagi. Accelerating evolutionary computation using a convergence point estimated by weighted moving vectors. *Complex and Intelligent Systems*, x(x):xx–xx (accepted), 2019.

[188] J. Yu, Y. Pei, and H. Takagi. Accelerating evolutionary computation using estimated convergence point. In *the IEEE Congress on Evolutionary Computation*, pages 1438–1444, 2016.

[189] J. Yu, Y. Pei, and H. Takagi. Competitive strategies for differential evolution. In *The 2018 IEEE International Conference on Systems, Man, and Cybernetics*, pages 268–273, 2018.

[190] J. Yu and H. Takagi. Clustering of moving vectors for evolutionary computation. In *7th International Conference on Soft Computing and Pattern Recognition*, pages 169–174, 2015.

[191] J. Yu and H. Takagi. Acceleration for fireworks algorithm based on amplitude reduction strategy and local optima-based selection strategy. In *8th International Conference on Swarm Intelligence*, pages 477–484, 2017.

[192] J. Yu and H. Takagi. Acceleration for fireworks algorithm based on amplitude reduction strategy and local optima-based selection strategy. In *8th International Conference on Swarm Intelligence*, pages 477–484, 2017.

[193] J. Yu and H. Takagi. Estimation of convergence points of population using an individual pool. In *10th International Workshop on Computational Intelligence and Applications*, pages 67–72, 2017.

[194] J. Yu and H. Takagi. Vegetation evolution for numerical optimization. In *JPNSEC International Workshop on Evolutionary Computation*, pages 49–54, 2018.

[195] J. Yu, H. Takagi, and Y. Tan. Multi-layer explosion-based fireworks algorithm. *International Journal of Swarm Intelligence and Evolutionary Computation*, 7(3), 2018.

[196] J. Yu, H. Takagi, and Y. Tan. Fireworks algorithm for multimodal optimization using a distance-based exclusive strategy. In *2019 IEEE Congress on Evolutionary Computation*, pages 2216–2221, 2019.

[197] J. Yu, Y. Tan, and H. Takagi. Accelerating fireworks algorithm with an estimated convergence point. In *9th International Conference on Swarm Intelligence*, pages 263–272, 2018.

[198] J. Yu, Y. Tan, and H. Takagi. Scouting strategy for biasing fireworks algorithm search to promising directions. In *The Genetic and Evolutionary Computation Conference Companion*, pages 99–100, 2018.

[199] A. Zamuda, J. Brest, B. Boovi, and V. umer. Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution. In *the IEEE Congress on Evolutionary Computation*, pages 3718–3725, 2008.

[200] D. Zhan and C. Xie. An improved multi-objective fireworks algorithm. In *9th International Symposium on Intelligence Computation and Applications*, pages 204–218, 2018.

[201] J. Zhang and A. C. Sanderson. Jade: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13(5):945–958, 2009.

[202] M. Zhang, W. Luo, and X. Wang. Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences*, 178(15):3043–3074, 2008.

[203] Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[204] W. Zhang and Y. Liu. Multi-objective reactive power and voltage control based on fuzzy optimization strategy and fuzzy adaptive particle swarm. *International Journal of Electrical Power and Energy Systems*, 30(9):525–532, 2008.

[205] W. Zhang and X. Xie. Depso: Hybrid particle swarm with differential evolution operator. In *the IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 3816–3821, 2003.

[206] F. Zhao, Q. Zhang, D. Yu, X. Chen, and Y. Yang. A hybrid algorithm based on PSO and simulated annealing and its applications for partner selection in virtual enterprise. In *Lecture Notes in Computer Science*, volume 3644, pages 380–389, 2005.

[207] S.Q. Zheng, A. Janecek, J.Z. Li, and Y. Tan. Dynamic search in fireworks algoritm. In *the IEEE Congress on Evolutionary Computation*, pages 3222–3229, 2014.

[208] S.Q. Zheng, A. Janecek, and Y. Tan. Enhanced fireworks algorithm. In *the IEEE Congress on Evolutionary Computation*, pages 2069–2077, 2013.

[209] Y.L. Zheng, L.H. Ma, Zhang. L.Y., and J.X. Qian. Empirical study of particle swarm optimizer with an increasing inertia weight. In *The IEEE Congress on Evolutionary Computation*, volume 1, pages 221–226, 2003.

[210] E. Zitzler, D. Brockhoff, and L. Thiele. The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In *Evolutionary Multi-criterion Optimization*, pages 862–876, 2007.

[211] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

Appendix: Publications Related to Each Chapter

This is a list of publications during doctoral research in each chapter.

Chapter 1: Introduction

Chapter 2: Related Techniques and Research

Chapter 3: Accuracy Improvements for an Estimated Convergence Point

1. Jun Yu and Hideyuki Takagi, "*Improving Estimation Precision of the Convergence Point of Individuals Using Weight-based Moving Vectors*," 13th Evolutionary Computing Meeting, pp. 59-63, Kusatu, Japan (Sept. 1-2, 2017) (in Japanese).

2. Jun Yu and Hideyuki Takagi, "*Estimation of Convergence Points of Population Using an Individual Pool*," 10th International Workshop on Computational Intelligence & Applications, pp. 67-72, Hiroshima, Japan (Nov. 11-12, 2017).

3. Jun Yu, Yuhao Li, Yan Pei and Hideyuki Takagi, "*Accelerating Evolutionary Computation Using a Convergence Point Estimated by Weighted Moving Vectors*," Complex & Intelligent Systems, vol. x, no. x, pp. xx-xx (accepted). DOI: 10.1007/s40747-019-0111-6.

Chapter 4: Accelerating Evolutionary Computation using Estimated Convergence Points

1. Jun Yu and Hideyuki Takagi, "*Correction methods for improving estimated convergence points for multi-modal optimization*," 9th Evolutionary Computing Meeting, pp. 92-97, Kobe, Japan (Sept. 7-9, 2015) (in Japanese).

2. Yan Pei, Jun Yu and Hideyuki Takagi, "*Evaluation of EC Acceleration by Using Estimated Points*," Evolutionary Computation Symposium 2015, pp. 292-297, Nishio, Japan (Dec. 19-20, 2015) (in Japanese).

3. Jun Yu and Hideyuki Takagi, "*Estimation of local optima areas based on individual distance ranks and fitness ranks*," 12th Evolutionary Computing Meeting, pp. 203-206, Fukuoka, Japan (Mar. 13-14, 2017) (in Japanese).

4. Jun Yu and Hideyuki Takagi, "*Accelerating Interactive Evolutionary Computation Using an Estimated Convergence Point*," The 33rd Fuzzy System Symposium, pp. 47-50, Yonezawa, Japan (Sept. 13-15, 2017) (in Japanese).

5. Jun Yu and Hideyuki Takagi, "*Estimated Convergence Points Applied to Evolutionary Computation*," SIG ECOmp of Japan Society for Fuzzy Theory and Intelligeny Informatics, Fukuoka, Japan (Feb. 28, 2018).

6. Jun Yu and Hideyuki Takagi, "*Clustering of Moving Vectors for Evolutionary Computation*," 7th International Conference on Soft Computing and Pattern Recognition, pp. 169-174, Fukuoka, Japan (Nov. 13-15, 2015).

7. Jun Yu, Yan Pei and Hideyuki Takagi, "*Accelerating Evolutionary Computation Using Estimated Convergence Point*," 2016 IEEE Congress on Evolutionary Computation, pp. 1438-1444, Vanvouver, Canada (July 24-29, 2016).

8. Yan Pei, Jun Yu and Hideyuki Takagi, "*Search Acceleration of Evolutionary Multi-objective Optimization Using an Estimated Convergence Point*," Mathematics, vol. 7, no. 2, pp. 129-147 (Jan., 2019).

Chapter 5: Accelerating Fireworks Algorithm Using an Estimated Convergence Point

1. Jun Yu, Ying Tan and Hideyuki Takagi, "*Accelerating Fireworks Algorithm with an Estimated Convergence Point*," 9th International Conference on Swarm Intelligence, pp. 263-272, Shanghai, China (June 17-22, 2018).

Chapter 6: Search Strategies for Fireworks Algorithm

1. Jun Yu and Hideyuki Takagi, "*Two-stage Explosions for Optimization Based on Fireworks Algorithm*," 10th Evolutionary Computing Meeting, pp. 19-23, Kawasaki, Japan (Mar. 17-18, 2016) (in Japanese).

2. Jun Yu and Hideyuki Takagi, "*Fireworks Algorithm with local fitness gradient estimated from fitness values of local individuals*," Evolutionary Computation Symposium 2016, pp. 131-137, Chiba, Japan (Dec. 10-11, 2016) (in Japanese).

3. Jun Yu, Ying Tan and Hideyuki Takagi, "*Scouting Strategy Applied to Fireworks Algorithm*," 14th Evolutionary Computing Meeting, Fukuoka, pp. 106-110, Koganei, Japan (Mar. 5-6, 2018) (in Japanese).

4. Jun Yu and Hideyuki Takagi, "*Acceleration for Fireworks Algorithm Based on Amplitude Reduction Strategy and Local Optima-based Selection Strategy*," 8th International Conference on Swarm Intelligence, pp. 477-484, Fukuoka, Japan (July 27 - Aug. 1, 2017).

5. Jun Yu, Ying Tan and Hideyuki Takagi, "*Scouting Strategy for Biasing Fireworks Algorithm Search to Promising Directions*," The Genetic and Evolutionary Computation Conference Companion, pp. 99-100, Kyoto, Japan (July 15-19, 2018).

6. Jun Yu, Hideyuki Takagi and Ying Tan, "*Multi-layer Explosion-Based Fireworks Algorithm*," International Journal of Swarm Intelligence and Evolutionary Computation, vol. 7, no. 3 (Dec., 2018).

Chapter 7: Niche Strategies for Fireworks Algorithm

1. Jun Yu, Hideyuki Takagi and Ying Tan, "*Niche Fireworks Algorithm by Distance-based Exclusive Strategy*," 15th Evolutionary Computing Meeting, pp. 27-30, Yokohama, Japan (Mar. 7-8, 2019) (in Japanese).

2. Jun Yu, Hideyuki Takagi and Ying Tan, "*Fireworks Algorithm for Multimodal Optimization Using a Distance-based Exclusive Strategy*," The 2019 IEEE Congress on Evolutionary Computation, pp. 2216-2221, Wellington, New Zealand (June 10-13, 2019).

Chapter 8: Competitive Strategies for Differential Evolution

1. Jun Yu, Yan Pei and Hideyuki Takagi, "*Competitive Strategies for Differential Evolution*," The 2018 IEEE International Conference on Systems, Man, and Cybernetics, pp. 268-273, Miyazaki, Japan (Oct. 7-10, 2018).

Chapter 9: Proposal of New Algorithm: Vegetation Evolution

1. Jun Yu and Hideyuki Takagi, "*Vegetation Evolution for Numerical Optimization*," JPNSEC International Workshop on Evolutionary Computation, pp. 49-54, Shenzhen, China (Aug. 31 - Sept. 1, 2018).

2. Jun Yu and Hideyuki Takagi, "*Performance Analysis of Vegetation Evolution*," The 2019 IEEE International Conference on Systems, Man, and Cybernetics, pp. xx-xx, Bari, Italy (Oct. 6-9, 2019) (accepted).

Chapter 10: Discussions and Conclusions