# A Cooperative Framework for Fireworks Algorithm

Shaoqiu Zheng, Junzhi Li, Andreas Janecek, and Ying Tan

**Abstract**—This paper presents a cooperative framework for fireworks algorithm (CoFFWA). A detailed analysis of existing fireworks algorithm (FWA) and its recently developed variants has revealed that ($i$) the current selection strategy has the drawback that the contribution of the firework with the best fitness (denoted as core firework) overwhelms the contributions of all other fireworks (non-core fireworks) in the explosion operator, ($ii$) the Gaussian mutation operator is not as effective as it is designed to be. To overcome these limitations, the CoFFWA is proposed, which significantly improves the exploitation capability by using an independent selection method and also increases the exploration capability by incorporating a crowdness-avoiding cooperative strategy among the fireworks. Experimental results on the CEC2013 benchmark functions indicate that CoFFWA outperforms the state-of-the-art FWA variants, artificial bee colony, differential evolution, and the standard particle swarm optimization SPSO2007/SPSO2011 in terms of convergence performance.

**Index Terms**—Swarm intelligence, fireworks algorithm, explosion amplitude, cooperative strategy

## 1 Introduction

In the past two decades, many stochastic and population-based swarm intelligence (SI) algorithms have been proposed. These algorithms have shown great success in dealing with optimization problems in various application fields. SI refers to the ability presented by the swarm behavior to solve problems through the direct or indirect interaction among the agents with their environment [1]. Usually, SI system consists of a population of decentralized and simple natural or artificial agents, the interactions among the agents lead to the emergence of intelligent ability [2]. By the observing and modeling of the cooperative swarm behavior of living creatures, artificial systems and the physical properties of non-living objects, researchers have been trying to understand those mechanisms and use them to design new algorithms.

Ant colony optimization (ACO) [3] and particle swarm optimization (PSO) [4] are two famous SI algorithms. ACO tries to model the cooperative behavior which searches for the shortest path from the source to the food through the environment in the colony. Each ant senses the pheromone trails to decide which node to move towards with probability. The trails with higher pheromone value will have higher probability to be selected. When an ant reaches the food, the route the ant once visited will be added with the pheromone [5]. This positive feedback mechanism and probability based routing scheme help the ants to find the shortest path from the source to the food. In general, each ant in the swarm learns the effective information for the problem and updates the pheromone trails in the route through trail and error. Finally, the trails with higher pheromone in the route will be taken as the solutions to the problem.

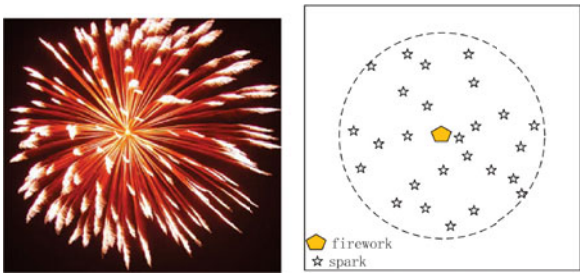
In contrast, PSO tries to mimick the aggregating motions of a flock of birds searching for food. Each particle (bird) denotes one feasible solution to the problem, and they cooperate with each other by sharing the effective information directly rather than by environment as ACO. Each particle has the ability to keep the best position in history and sense the global (local) best position of the particle swarm. The particles move under the guide of a cognitive component (relative to their individual past performance) and a social component (relative to the population of particles' performance) [1].

The great successes of ACO and PSO for solving optimization problems greatly push the SI developments forward. Algorithms inspired by collective biologic behaviors of bees [6], [7], [8], glowworms [9], fish schools [10], firefles [11], [12], cuckoos [13], krill herds [14], human beings [15], bacteria [16], bats [17] and collective artificial systems [18], [19], [20] were proposed. See [21] for a comprehensive survey of recently developed computational intelligence algorithms.

Inspired by the phenomenon of fireworks explosion in the night sky, a new SI algorithm called fireworks algorithm (FWA) [20] has been proposed recently. In FWA, the fireworks perform independent search at a number of locations by sampling points around the fireworks within the explosion amplitudes, and the information among these fireworks is also shared for adjusting the sampling probability for the global search. In fact, the explosion of fireworks and evaluation of the explosion sparks can be seen


- *S. Zheng is with the Department of Machine Intelligence, Key Laboratory of Machine Perception (Ministry of Education), Peking University, and the 28th Research Institute of China Electronics Technology Group Corporation, Nanjing 210007, Jiangsu, China. E-mail: zhengshaoqiu1214@foxmail.com.*
- *J. Li and Y. Tan are with the Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, and Key Laboratory of Machine Perception (Ministry of Education), Peking University, Beijing 100871, China. E-mail: {ljz, ytan}@pku.edu.cn.*
- *A. Janecek is with the University of Vienna, Research Group Theory and Applications of Algorithms 1090, Vienna, Austria. E-mail: andreas.janecek@univie.ac.at.*


*Manuscript received 20 Apr. 2015; revised 24 June 2015; accepted 23 Oct. 2015. Date of publication 2 Nov. 2015; date of current version 2 Feb. 2017. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TCBB.2015.2497227*




(a) firework explosion (b) solution search

Fig. 1. Comparison between firework explosion and solution search for optimization problems.

as one search manner for solutions in the feasible range (see Fig. 1). Since FWA has shown success in solving practical optimization problems, it is believed that FWA is efficient and worth for further study.

### 1.1 Related Work

Related work based on FWA can be grouped into three categories, theoretical analysis, algorithm developments and applications. In [22], Liu et al. presented a theoretical analysis for FWA, and proved that FWA is an absorbing Markov stochastic process. Moreover, the convergence property and time complexity of FWA were also discussed.

In [23], Pei et al. investigated the influences of landscape approximation approaches on accelerating FWA by introducing the elite point of the approximation landscape into the swarm of fireworks. Additionally, Liu et al. [24] proposed new methods for calculation of the explosion amplitudes and the numbers of explosion sparks while still maintaining the core idea that fireworks with better fitness will generate more sparks within smaller explosion amplitudes.

The first comprehensive study of the operators of FWA can be found in [25], where Zheng et al. proposed the enhanced fireworks algorithm (EFWA), which incorporates five modifications compared to conventional FWA: ($i$) a new minimal explosion amplitude check strategy (MEACS), ($ii$) a new operator for generating explosion sparks, ($iii$) a new mapping strategy for sparks which are out of the search space, ($iv$) a new operator for generating Gaussian sparks, and ($v$) a new selection strategy. The limitations in conventional FWA are presented and the corresponding improvements are proposed.

Based on the work of EFWA, Zheng et al. proposed the dynamic search fireworks algorithm (dynFWA) [26]. In dynFWA, the firework with minimal fitness in each iteration is called *core firework (CF)* and uses a dynamic explosion amplitude strategy, while for the rest of fireworks, the same strategies as in EFWA are used (see Section 3.2.1 for the details of dynFWA). In addition, Li et al. [27] proposed an adaptive version of FWA (AFWA), in which the explosion amplitude of the firework with minimal fitness is calculated as the infinite norm distance between the firework and a certain candidate from the explosion sparks (see Section 3.2.2 for the details of AFWA). In [28], Zheng compared the performances among the FWA, EFWA, dynFWA and AFWA on ICSI2014 competition problems.

Additionally, a number of hybrid algorithms combining FWA with other algorithms have been proposed. Gao and Diao [29] proposed cultural fireworks algorithm which is the hybrid between cultural algorithm and FWA. Zheng et al. [30] proposed a hybrid algorithm FWA-DE of FWA and differential evolution (DE). Zhang et al. [31] proposed a hybrid biogeography-based optimization with fireworks algorithm while Yu et al. [32] also presented a hybrid algorithm between DE with FWA. In [33], the hybrid algorithm between FWA and firefly algorithm was proposed.

For multi-objective FWA (MOFWA), Zheng et al. proposed the framework of MOFWA [34] which uses the fitness assignment strategy from [35] and minimal pairwise distance metrics from [36]. Liu et al. proposed S-metric fitness calculation method, which can consider the strength and diversity of the solution simultaneously [37].

For Parallel FWA implementation, Ding et al. proposed GPU-FWA [38], a GPU implementation which maintains a high speedup value.

For the practical applications, FWA has been applied in FIR and IIR digital filters design [29], the calculation of nonnegative matrix factorization (NMF) [39], [40], [41], spam detection [42], image recognition [43], power system reconfiguration scheme [44], mass minimisation of trusses [45], [46], non-linear equation set [47] and 0/1 knapsack problems [48] and so on.

For more details on FWA's work, please visit FWA research forum, http://www.cil.pku.edu.cn/research/fwa/index.html or refer the recently published FWA book [49].

### 1.2 Contributions

The main contributions of this paper are: ($i$) an evaluation criterion of the cooperative strategies among the fireworks of EFWA, dynFWA and AFWA is conducted. The significance improvements of fireworks reveal that the fireworks except for the best firework contribute less for optimization while consuming lots of evaluation times. ($ii$) the proposal of the cooperative framework for FWA (CoFFWA) with independent selection method and crowdness-avoiding cooperative strategy. In the proposed cooperative framework, the independent selection method will ensure the information inheritance and improve the local search ability for each firework while the cooperative strategy can enhance the global search ability of the fireworks swarm.

### 1.3 Synopsis

The remainder of this paper is organized as follows: Section 2 introduces the framework of FWA and EFWA, and gives a comparison between FWA with other meta-heuristic algorithms. Section 3 describes two state-of-the-art improvement works on FWA, i.e., dynFWA and AFWA, which focus on the explosion amplitude strategies. In Section 4, we present a comprehensive analysis of cooperative strategies in conventional FWA framework. To overcome the limitations, the new cooperative framework of FWA is finally proposed. To validate the performance of the proposed algorithm, several experiments are conducted in Sections 5 and 6. Finally, concluding remarks are given in Section 7.

## 2 The Framework of FWA and EFWA

Assume $f$ is a minimization optimization problem.[1] A principal FWA works as follows: At first, $N$ fireworks are

1. In this paper, without loss of generality, the optimization problem $f$ is assumed to be a minimization problem.

initialized randomly, and their quality (i.e., fitness) is evaluated to determine the explosion amplitudes and the numbers of sparks. Subsequently, the fireworks explode and generate sparks within their local space. To ensure diversity and balance the global and local search, the explosion amplitudes and the population sizes of the newly generated explosion sparks differ among the fireworks. A firework with better fitness can generate a larger population of explosion sparks within a smaller range, i.e., within a small explosion amplitude. Contrary, a firework with lower fitness can only generate a smaller population within a larger range, i.e., within bigger explosion amplitude. This technique allows to balance between exploration and exploitation capabilities of the algorithm. Here, *exploration* refers to the ability of the algorithm to explore various regions of the search space in order to locate promising good solutions, while *exploitation* refers to the ability to conduct a thorough search within a smaller area recognized as promising in order to find the optimal solution [50]. Exploration is achieved by fireworks with large explosion amplitudes, since they have the capability to escape from local minima. Exploitation is achieved by fireworks with small explosion amplitudes, since they reinforce the local search ability in promising areas. After the explosion, another type of sparks are generated by Gaussian mutation operator. The idea behind this is to further ensure diversity of the swarm. To improve readability we assign notations to the two distinct types of sparks: "explosion sparks" are generated by the explosion operator, and "Gaussian sparks" are generated by Gaussian mutation operator. To retain the information of the fireworks swarm, and pass it to the next iteration, a subset of the whole population is selected at the end of each iteration. The algorithm continues until the terminal criterion is met. The framework of (E)FWA is given in Algorithm 1.

**Algorithm 1.** The General Structure of (E)FWA

```
Initialize N fireworks and evaluate their fitness
repeat
    Calculate the explosion amplitudes
    Calculate the numbers of explosion sparks
    Generate "explosion sparks" (check if out-of-bounds)
    Evaluate the fitness of explosion sparks
    Generate "Gaussian sparks" (check if out-of-bounds)
    Evaluate the fitness of Gaussian sparks
    Select fireworks for the next iteration
until termination criterion (time, max. # evals, convergence, ... ) is met
```

## 2.1 Explosion Operator

To perform the search for a firework, the points around the firework's position are sampled with uniform distribution taking inspiration from the fireworks explosion process (see Fig. 1). For this sampling operation, the sampling model and the sampling points number are two important factors which will influence the optimization results. The sampling points number is denoted as *explosion sparks number* while the sampling model is characterized with uniform distribution probability function within the sampling range (denoted as *explosion amplitude*).

### 2.1.1 Explosion Amplitude and Explosion Sparks Number

Assume the fireworks number is $N$, the explosion amplitude $A$ and the number of explosion sparks $s$ for firework $X_i$ are calculated as follows:

$$A_i = \hat{A} \cdot \frac{f(X_i) - \min_{\mathrm{k}}(f(X_k)) + \varepsilon}{\sum_{j=1}^{N}(f(X_j) - \min_{\mathrm{k}}(f(X_k))) + \varepsilon}, \tag{1}$$

$$s_i = M_e \cdot \frac{\max_{\mathrm{k}}(f(X_k)) - f(X_i) + \varepsilon}{\sum_{j=1}^{N}(\max_{\mathrm{k}}(f(X_k)) - f(X_j)) + \varepsilon}, \tag{2}$$

where, $\hat{A}$ and $M_e$ are two constants controlling the explosion amplitudes and the numbers of explosion sparks, respectively, and $\varepsilon$ is the machine epsilon. Additionally, $s_i$ is also limited by the lower/upper bounds.

Eq. (1) reveals that the explosion amplitude of the firework at the best location $X_{CF}$ (denoted as core firework, refer Section 3.1) is usually very small [close to 0], the explosion sparks of $X_{CF}$ will be located at the same location as $X_{CF}$. To overcome this problem, EFWA uses the *minimal explosion amplitude check strategy* to bound the explosion amplitude $A_i$ of firework $X_i$ as follows:

$$A_i = \begin{cases} A_{\min} & \text{if } A_i < A_{\min}, \\ A_i & \text{otherwise}, \end{cases} \tag{3}$$

where, $A_{\min}$ decreases non-linearly with increasing number of function evaluations such that,

$$A_{\min} = A_{\text{init}} - \frac{A_{\text{init}} - A_{\text{final}}}{E_{\max}} \sqrt{(2E_{\max} - t)\, t}, \tag{4}$$

where, $t$ refers to the number of function evaluation at the beginning of the current iteration, and $E_{\max}$ is the maximum number of evaluations. $A_{\text{init}}$ and $A_{\text{final}}$ are the initial and final minimum explosion amplitude, respectively.

### 2.1.2 Generating Explosion Sparks

Now, each firework explodes and creates a different number of *explosion sparks* within a given range of its current location. For firework $X_i$, it will perform Algorithm 2 for $s_i$ times. Here, $\mathrm{uniform}(a, b)$ denotes a random number from uniform distribution in $[a, b]$.

**Algorithm 2.** Generating One "Explosion Spark" in EFWA

```
Initialize the location: x̂_i = X_i
for each dimension k of x̂_i do
    if uniform(0, 1) < 0.5 then
        x̂_ik ← x̂_ik + A_i · uniform(−1, 1)
    end if
end for
return x̂_i
```

### 2.1.3 Mapping Out-of-Bounds Sparks

When the explosion spark $\hat{x}_i$ exceeds the search range in dimension $k$, it will be mapped to another location in the search space with uniform distribution according to $\hat{\mathrm{x}}_{ik} = X^k_{min} + \mathrm{uniform}(0, 1) \cdot (X^k_{max} - X^k_{min})$.

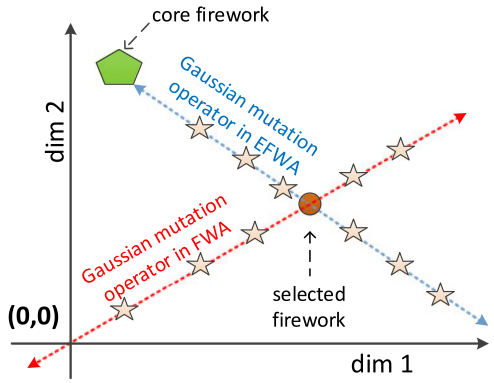



Fig. 2. Gaussian mutation operator in FWA and EFWA.

## 2.2 Gaussian Mutation Operator

In the history of FWA's developments, two typical Gaussian mutation operators were proposed. In FWA [20], the Gaussian mutation operator focuses on the mutation ability for each firework alone by multiplying a random value under Gaussian distribution with the positions of the firework in the selected dimensions. Later on in EFWA [25], the introduced Gaussian mutation operator emphasizes the cooperative evolution in the fireworks swarm (see Fig. 2).

### 2.2.1 Generating Gaussian Sparks

Algorithm 3 gives the details about how one Gaussian spark is generated in EFWA. This algorithm is performed $M_g$ times, each time with a randomly selected firework $X_i$. Here, $M_g$ is a constant to control the number of Gaussian sparks, $\text{normal}(a, b)$ denotes a random value from a normal distribution with expected value and variance set to $a$ and $b$, respectively. $X_{CF}$ is the position of the best firework (refer Section 3.1).

**Algorithm 3.** Generating One "Gaussian Spark" in EFWA

1: Initialize the location: $\bar{\mathrm{x}}_i = X_i$
2: $e = \text{normal}(0, 1)$
3: **for** each dimension $k$ of $\bar{x}_i$ **do**
4: **if** $\text{uniform}(0, 1) < 0.5$ **then**
5: $\bar{\mathrm{x}}_{ik} \leftarrow \bar{\mathrm{x}}_{ik} + (X_{CF,k} - \bar{\mathrm{x}}_{ik}) \times e$
6: **end if**
7: **end for**
8: **return** $\bar{\mathrm{x}}_i$

### 2.2.2 Mapping Out-of-Bounds Sparks

When the Gaussian spark $\bar{\mathrm{x}}_i$ exceeds the search range in dimension $k$, the mapping method is similar as the explosion sparks' method (see Section 2.1.3).

## 2.3 Selection Strategy

To retain the information of the swarm and pass it to the next iteration, a subset of the whole population has to be selected. EFWA applies a computationally efficient selection method called *Elitism-Random Selection* (ERS, [51]). The optimum of the set including all fireworks, the explosion sparks, and the Gaussian sparks will be selected firstly and the rest of individuals are selected randomly. Although this method is rather simple compared to distance based selection method in FWA (cf. [20], [52]), our analysis in [25] has shown that there is almost no difference in terms of convergence speed, final fitness and standard deviation between the distance based selection method of conventional FWA and ERS.

## 2.4 Comparison with Other Meta-Heuristic Algorithms

Since its introduction, FWA has shown great success for optimization problems, which indicates that some mechanisms of this algorithm are efficient. Here, we pick two famous algorithms, particle swarm optimization (PSO) and genetic algorithm (GA) for comparison.

### 2.4.1 Reproduction Mechanism

From the offspring generation view, we present the following definitions: "*cross-propagation*" means that the offspring is generated by at least two parents, while "*self-propagation*" means the offspring is generated by only one parent.

Thus, it can be seen that both cross-propagation and self-propagation are presented in GA, and there is no bijection from generation to generation. In PSO, the position of the son particle is based on this particle's history information and the global (local) best particle's information. PSO uses only the cross-propagation and there is a bijection from generation to generation. The reproduction mechanism in FWA is similar to GA to some extent. The explosion sparks are generated by the explosion operator performed by one firework and mutation sparks are generated by the mutation operator, i.e., both cross-propagation and self-propagation are presented in FWA and there is no bijection. However, different from the GA, the explosion operator in FWA plays the major role for optimization.

### 2.4.2 Exploration and Exploitation Capabilities

Previous work in [53] has given a comparison between PSO and GA. For GA, the crossover operator is usually very effective. At the early searching phase, the differences between the chromosomes make crossover operator be able to move the offspring to a far place in the solution space, which allows GA to maintain high exploration capability. While at the later searching phase, as the population of chromosomes converges, the crossover operator will not be able to move it to a far place as the two chromosomes usually have the similar structure, which makes the GA maintain high exploitation capability. In addition, the mutation operator can move the chromosome to anywhere in the solution space which enhances the exploration capability of GA. In PSO, each particle is determinated by its position in the last iteration, its best position in history and the global (local) best position of the population. This has similar function as the crossover operator in GA [53], and finally the whole population of particles will converge to one position, which makes the swarm maintain a high exploitation capability. The particles have to move step by step from one place to another, but cannot jump very far in one iteration [53]. It seems that GA and FWA can do this by using a high mutation rate and a high explosion amplitude, respectively. For FWA, the exploration and exploitation capabilities are gained by taking different explosion amplitudes and explosion sparks numbers for each firework. A small explosion amplitude with large explosion sparks number

leads to a comprehensive local search. To some extent, FWA is partly similar to GA, the small (big) explosion amplitude has the similar effect as mutation operator with small (big) mutation rate. Moreover, the explosion amplitude together with the selection operation in FWA is also partly similar to the velocity of particle in PSO, the larger values enable the firework or particle move fast to the global best position.

#### 2.4.3 Survival Strategy

In GA, the selection strategy is performed in each iteration to maintain and pass the information to the next iteration. The fittest candidate will be surely selected to the next iteration, while for the rest of candidates, they are selected with probability, which means that there is no bijection from generation to generation. For PSO, it does not utilize the selection operator, and there is a bijection from generation to generation. In FWA, the selection operator is performed like GA in each iteration. Section 4.1 will give a detailed discussion of the survival strategy.

## 3 State-of-the-Art Improvement Work: dynFWA and AFWA

### 3.1 Definition

For the convenience of discussion, the following definitions are introduced at first:

*Core firework (CF) and non-core firework (non-CF)*: Among the fireworks swarm, the firework with minimal fitness is denoted as CF. All other fireworks except the CF are denoted as non-CFs.

The fireworks which have the same parent firework as current CF may have the similar performance as the CF due to the closeness in position. Thus, we extend the CF to general CF. The detailed definition is given below.

*General core firework (GCF) and non-general core firework (non-GCF)*: In iteration $t$, after the generation of explosion sparks, in the candidates set ($S_c$), which includes the fireworks and explosion sparks,[2] the selection operation is performed. Let us define $\mathrm{x}_b$ as the "best" newly created explosion spark of all fireworks in the swarm, and $X_{CF}$ as the current core firework. $P(\mathrm{x}_i)$ denotes the firework which generated $\mathrm{x}_i$, and $S^t_{GCFs}$ denotes the set of GCF.

- If $f(\mathrm{x}_b) - f(X_{CF}) < 0$, and the candidate $x_i$ which is from the candidate set $S_c$, and subjected to the following condition,

$$\mathrm{x}_i = \arg\limits_{\mathrm{x}_i} \quad (P(\mathrm{x}_i) == P(\mathrm{x}_b))||(\mathrm{x}_i == P(\mathrm{x}_b))$$

  and $\mathrm{x}_i$ is selected as firework into next iteration, then $\mathrm{x}_i \in S^{t+1}_{GCFs}$.
- If $f(\mathrm{x}_b) - f(X_{CF}) \geq 0$, and the candidate $\mathrm{x}_i$ which is from the candidate set $S_c$, and subjected to the following condition,

$$\mathrm{x}_i = \arg\limits_{\mathrm{x}_i} \quad (P(\mathrm{x}_i) == X_{CF})||(\mathrm{x}_i \in S^t_{GCFs})$$

  and $\mathrm{x}_i$ is selected as firework into next iteration, then $\mathrm{x}_i \in S^{t+1}_{GCFs}$.

All fireworks except the GCF are denoted as non-GCFs.

2. Here, the Gaussian mutation operator is eliminated to analysis the performance of explosion sparks.

### 3.2 Explosion Strategies in dynFWA and AFWA

In EFWA, the MEACS (refer Section 2.1) enforces the exploration capabilities at the early phase of the algorithm (larger $A_{\min} \Rightarrow$ global search), while at the final phase of the algorithm, the exploitation capabilities are enforced (smaller $A_{\min} \Rightarrow$ local search). Additionally, the non-linear decrease of $A_{\min}$ (cf. Eq. (4)) enhances exploitation already at an earlier stage of the EFWA algorithm. However, this procedure decreases the explosion amplitude solely with the current number of function evaluations which heavily depends on the preseted number of iterations for the algorithm. The explosion amplitude strategy should consider the optimization process information and the information of the explosion sparks rather than solely the information about the current iteration [evaluation] count. To approach this problem, the adaptive explosion amplitude strategies for fireworks were proposed, which are *dynamic search in FWA* (dynFWA) and *adaptive FWA* (AFWA).

In dynFWA and AFWA, the fireworks are separated into two groups. The first group consists of the CF, while the second group consists of all remaining fireworks. The responsibility of the CF is to perform a local search around the best location, while the responsibility of the second group is to maintain the global search capability.

**Algorithm 4.** Explosion Amplitude Update Strategy in dynFWA

**Require:** Define:
$X_{CF}$ is the current location of the CF;
$\mathrm{x}_b$ is the best location among all explosion sparks;
$A^t_{CF}$ is the CF's explosion amplitude in iteration $t$;
$C_a$ is the amplification factor;
$C_r$ is the reduction factor;
**Ensure:**
**if** $f(\mathrm{x}_b) - f(X_{CF}) < 0$ **then**
  $A^t_{CF} \leftarrow A^{t-1}_{CF} \times C_a$;
**else**
  $A^t_{CF} \leftarrow A^{t-1}_{CF} \times C_r$;
**end if**

#### 3.2.1 The dynFWA

For an optimization problem, the ideal strategy for fireworks is to make them move towards the global optimum as fast as possible, which means the firework should 1) move fast, 2) in the right direction. In dynFWA, the explosion amplitude adapts itself (through amplification and reduction) according to the quality of the generated sparks.

Assume that the current position of the CF is far away from the global/local minimum, then the amplification of explosion amplitude is one direct and effective approach to increase the step-size towards the global/local optimum, i.e., it allows for faster movements. However, we should make sure that the fireworks walk towards to global/local optimal position. In fact, the probability to find a position with better fitness decreases with the increasing of explosion amplitude due to the increased search space, which makes it a challenge for firework's performance improvement.

**Case 1)** One or several explosion sparks have found a better position. It is possible that ($i$) an explosion spark

generated by the CF has found the best position, or that ($ii$) an explosion spark generated by a *different* firework than the CF has found the best position. Both cases indicate that the swarm has found a new promising position and that $x_b$ will be the CF for the next iteration.

(i) In most cases, $x_b$ has been created by the CF. In such cases, in order to speed up the convergence of the algorithm, the explosion amplitude of the CF for the next iteration will be increased compared to the current iteration.

(ii) In other cases, a firework different from the CF creates $x_b$. In such cases, $x_b$ will become the new CF for the next iteration. Since the position of the CF is changed, the current explosion amplitude of the current CF will not be effective to the newly selected CF. However, it is possible that $x_b$ is located in rather close proximity to the previous CF due to the fact that the random selection method may select several sparks created by the CF, which are initially located in close proximity to the CF. If so, the same consideration as in ($i$) applies. If $x_b$ is created by a firework which is not in close proximity to the CF, the explosion amplitude can be re-initialized to the preseted value. However, since it is difficult to define "close" proximity, we do not compute the distance between $x_b$ and $X_{CF}$ but rely on the dynamic explosion amplitude update ability. Similarly to ($i$), the explosion amplitude is increased. If the new CF cannot improve its location in the next iteration, the new CF is able to adjust the explosion amplitude itself dynamically.

**Case 2)** None of the explosion sparks of the CF or of all other fireworks has found a position with better fitness compared to the CF. Under this circumstance, the explosion amplitude of the CF is reduced in order to narrow down the search to a smaller region around the current location and to enhance the exploitation capability of the CF. The probability for finding a position with better fitness usually increases with the decreasing of explosion amplitude.

#### 3.2.2 The AFWA

In AFWA, the motivation is to calculate the most suitable amplitude for the CF without preseted parameters by making full use of the region's information based on the generated explosion sparks around the current position [27].

AFWA aims at finding a specific spark and using its distance to the best individual (which is either the $X_{CF}$ or $x_b$, and will be selected as the CF for the next iteration) as the explosion amplitude for the next iteration. The specific spark that AFWA chooses should be subject to the following two conditions:

- Its fitness is worse than the one of the current CF, which guarantees that the specific spark is not too close to the best individual;
- The distance between the specific spark with the best individual is minimal among all the candidates, which is to ensure the convergence.

As in FWA, the explosion sparks are generated independently in each dimension, the *infinite norm* is taken as the distance measure as follows:

$$\|x\|_\infty = \max_i(|x_i|). \tag{5}$$

AFWA's explosion amplitude update strategy includes two cases (it uses only one firework for analysis):

**Case 1)** The CF does not generate any sparks with smaller fitness than the firework's, i.e., the *best fitness of all the explosion sparks* $f(x_b)$ is worse than the core firework's fitness $f(X_{CF})$, and the explosion amplitude will be set to the infinite norm between the core firework and the specific spark. If so, the explosion amplitude in the next iteration will be reduced according to Algorithm 5.

**Case 2)** The generated $x_b$ has smaller fitness than $X_{CF}$, according to Algorithm 5, it is hard to determinate whether the explosion amplitude will be increased or decreased. Generally, the explosion amplitude will be increased with high probability while be reduced with smaller probability.

Moreover, as the infinite norm between the specific spark and the firework may change radically, AFWA introduces the smoothing strategy, $A_{CF}^t \leftarrow 0.5(A_{CF}^{t-1} + \lambda \cdot A_{CF}^t)$.

**Algorithm 5.** Explosion Amplitude Update Strategy in AFWA

**Require:** Define:
$M$ is explosion sparks number of CF;
$A_{CF}^t$ is the CF's explosion amplitude in iteration $t$;
$x_k$ is the $k$th explosion spark of CF;
$x_b$ is the best explosion spark;
$\lambda$ is a constant parameter which controls the update rate.
**Ensure:**
1: **for** $k = 1$ **to** $M$ **do**
2: **if** $||x_k - x_b||_\infty > A_{CF}^t$ **and** $f(x_k) > f(X_{CF})$ **then**
3: $A_{CF}^t \leftarrow ||x_k - x_b||_\infty$
4: **end if**
5: **end for**
6: $A_{CF}^t \leftarrow 0.5 \cdot (A_{CF}^{t-1} + \lambda \cdot A_{CF}^t)$

### 3.3 dynFWA versus AFWA

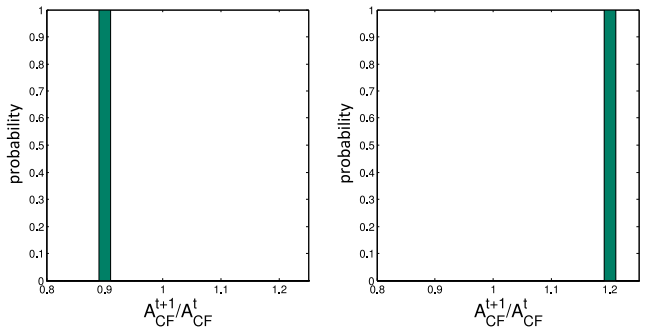
From Algorithms 4 and 5, it can be seen that if the CF cannot generate any sparks with better fitness, then the explosion amplitude of the firework will be reduced in both AFWA and dynFWA. If the CF has generated a spark with better fitness, the explosion amplitude of the firework in dynFWA will be amplified. However, for AFWA, the determination of amplification or reduction is more complex.

To investigate the update of explosion amplitude (amplification and reduction) during the optimization process, we record the values of $A_{CF}^t/A_{CF}^{t-1}$.

For dynFWA, if a better solution is found, the recorded value $A_{CF}^t/A_{CF}^{t-1}$ will be $C_a$ and the explosion amplitude is updated with the amplification factor. Otherwise, the recorded value $A_{CF}^t/A_{CF}^{t-1}$ will be $C_r$ and the explosion amplitude is updated with the reduction factor (cf. Figs. 3b and 3a).

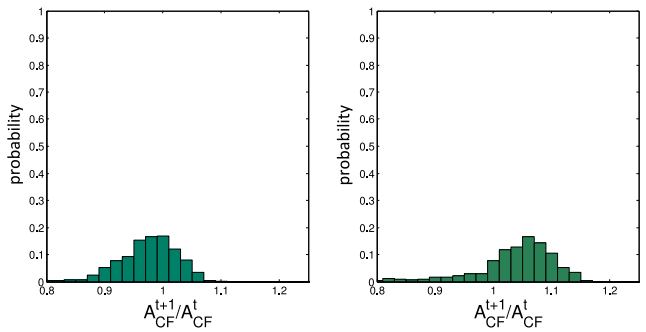
To investigate the explosion amplitude update process of AFWA, Sphere function is chosen as the benchmark function to study the explosion amplitude amplification and reduction when the firework has found or not found a better solution.

Fig. 3d presents the histogram of $A_{CF}^t/A_{CF}^{t-1}$ in the iterations when a better solution is found, here the y-coordinate denotes the probability. Fig. 3c presents the histogram of




(a) dynFWA, the explosion amplitude is updated when no better solutions are found.

(b) dynFWA, the explosion amplitude is updated when a better solution is found.




(c) AFWA, the explosion amplitude is updated when no better solutions are found.

(d) AFWA, the explosion amplitude is updated when a better solution is found.

Fig. 3. The comparison of explosion amplitude update between dynFWA and AFWA.

$A_{CF}^{t}/A_{CF}^{t-1}$ in the iterations when no better solutions are found. The geometric mean in Fig. 3d is 1.017, while the geometric mean in Fig. 3c is 0.976. It can been seen that if the firework has found a better position with smaller fitness, the explosion amplitude in AFWA will be increased with higher probability, and reduced with smaller probability.

Here, a special case is also considered to compare the two explosion amplitude strategies. Assume that explosion sparks number $M$ is a very big value, *i.e.* $M \to +\infty$, and the generated sparks are located in a float and continues region. Under this circumstance, if the generated sparks have found any better positions, then usually the best explosion spark will be located at the edge of the region (see Fig. 4).[3] For AFWA, the specific spark will be located quite near the firework. As $M \to +\infty$, the calculated $A_{CF}^{t}$ by operation in line 3 of Algorithm 5 will be equal with $A_{CF}^{t-1}$, and the final $A_{CF}^{t}$ calculated by operation in line 6 of Algorithm 5 will be $A_{CF}^{t} = 1.15A_{CF}^{t-1}$, $(\lambda = 1.3)$. Thus, the explosion amplitude for next iteration will be amplified. For dynFWA, as the fireworks have found a spark with better fitness, the amplitude will be amplified according to Algorithm 4, $A_{CF}^{t} = 1.2A_{CF}^{t-1}$, $(C_a = 1.2)$.

Thus, we can draw the following conclusions:

- The explosion amplitude is one of the most crucial parameters for FWA to balance the global and local search ability.

3. In fact, the search space for one firework is hypercube rather than the hypersphere.

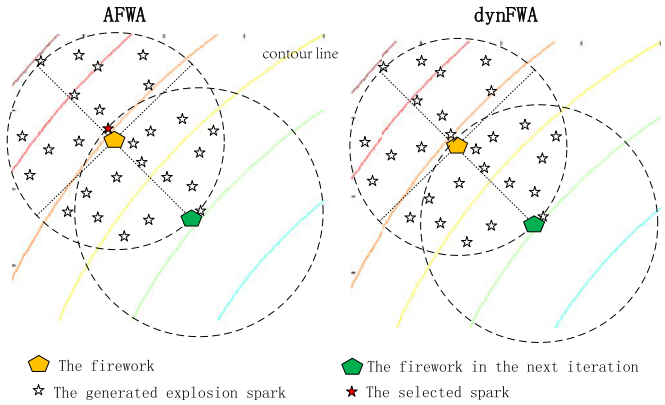



Fig. 4. dynFWA versus AFWA in a special case.

- From the statistical view, the two methods, dynFWA and AFWA are different in terms of adjusting the explosion amplitude, but reach the same goal by different means.

## 4 The Proposed FWA Cooperative Framework

In this section, we first give a detailed analysis of the cooperative strategies in conventional FWA framework and the limitations are pointed out, and then the new cooperative FWA framework (CoFFWA) is finally proposed, which includes the independent selection method for each firework and crowdness-avoiding cooperative strategy among the fireworks.

### 4.1 Analysis of Cooperative Strategies in Conventional FWA Framework

FWA is designed as one swarm intelligence algorithm, in which the fireworks in the swarm cooperate with each other to deal with the task that one firework cannot work well on. The prerequisites of the successful cooperative strategy are the individuals participated in the cooperative behaviour maintain effective but different information. In EFWA, two cooperative operations are used for the implementation of this idea:

- The sharing of the fitness values in the population for calculation of explosion amplitudes and explosion sparks numbers in the explosion operator (cf. Section 2.1);
- The Gaussian mutation strategy in the Gaussian mutation operator (cf. Section 2.2).

The sharing of fitness values among the fireworks makes the fireworks with smaller fitness have smaller explosion amplitudes and larger number of explosion sparks, which maintains the capability of exploitation, while the fireworks with bigger fitness have bigger explosion amplitudes and smaller numbers of explosion sparks, which maintains the capability of exploration. As for the Gaussian mutation operator, the generated Gaussian sparks will locate along the direction between the core firework and the selected firework. The Gaussian sparks are able to inherit the effective information of selected firework and also learn from the core firework, which are assumed to improve the diversity of the fireworks swarm.

However, for EFWA, dynFWA, and AFWA, all of them take the probability based selection method from the *candidates* set which includes the fireworks, the generated explosion sparks and Gaussian sparks. For all of the FWA variants, the basic principle of the selection method is that the optimum among the *candidates* is always kept while for the rest of fireworks, different methods have different selection probability calculation methods.

In the following, we will present an observation that these kinds of selection methods result in the fact that, the non-CFs of the explosion operator contribute less for the optimization while taking up the most evaluation times, and the Gaussian mutation operator proposed in EFWA is not as effective as it was designed to be.

#### 4.1.1 Cooperative Performance Analysis of CF and Non-CFs in Explosion Operator

In FWA, the calculation method of explosion amplitude will make the explosion amplitude of the CF close to zero. To avoid this limitation, EFWA introduces the MEACS (cf. Section 2.1), where the CF's explosion amplitude is in fact calculated based on the MEACS which is not relevant to the non-CFs in the population. In dynFWA or AFWA, the situation is similar, that the explosion amplitude of CF is calculated based on the dynamic search strategy or adaptive strategy solely which is not relevant to the non-CF's fitness. Thus, it can be seen that the CF's explosion amplitude strategies are independent of the non-CFs' in the fireworks swarm, the only interaction between the CF and non-CFs except for the selection method is the calculation of explosion sparks number which is powerless for dealing with complex problems.

For the selection methods, the sparks generated by CF and non-CFs are brought together. The candidate with minimal fitness [will be the CF in the next iteration] is always selected at first, while for the rest of fireworks [will be non-CFs in the next iteration], they are selected with probability.

In summary, the differences between the CF and non-CFs lie in two aspects, the explosion amplitudes' calculation method and the selection probability calculated by the selection strategy. In the following, we will present that the selection method makes the initial idea that the non-CFs are to increase the diversity of the fireworks swarm not effective.

In the selection, If a selected non-CF candidate is generated by the CF, then it will have similar performance as the CF. Alternatively, the selected candidate is generated by the non-CFs. In this case, the non-CFs are usually generated by the parent fireworks within large explosion explosion amplitudes which can be seen as randomly generating of sparks within the search range as the information of the non-CF can only be passed to the next iteration consecutively within few iterations with the conventional selection method, thus after a number of iterations, the position of the non-CF will be reinitialized in a new position and will be kept only in few iterations.

In fact, these kind of selection strategies will make the generated sparks by the non-CFs have the similar performance with randomly generated sparks in the search space to some extent or have the similar performance with sparks generated by CF. We cannot expect the good performance by randomly generating the sparks if there is no much heuristic information.

In Section 5, experiments are designed to evaluate the performance of CF and GCF to validate our idea that the non-CFs and non-GCFs make little contribution to the optimization.

#### 4.1.2 Cooperative Strategy in Gaussian Mutation Operator

The motivation of Gaussian sparks is to increase the diversity of the swarm. The prerequisite for successful performance by cooperating in the fireworks population is that the heuristic information of each firework is different and effective.

In EFWA, the newly generated Gaussian sparks locate along the direction between the selected firework ($X_i$) and CF ($X_{CF}$). For the selected firework $X_i$, it can be classified into two categories. The first category comprises fireworks which are very close to $X_{CF}$, usually these fireworks have the same parent firework as the CF. Then, the newly generated Gaussian sparks may have similar performance with the explosion sparks which cannot increase the diversity of the swarm effectively. The other category is the fireworks which are not close to the CF, usually it is because these fireworks are generated by a firework different from the parent firework of CF. If so, the newly generated Gaussian sparks will be $(i)$ close to CF, $(ii)$ close to the selected firework, $(iii)$ not close to any fireworks. If the newly generated Gaussian spark is close to either the CF or the selected firework, it has similar performance with the explosion sparks generated by them. If the generated Gaussian spark is not close to them, then it can be seen as the spark generated by firework with large explosion amplitude. Thus, the newly generated Gaussian spark will not work effectively to increase the diversity of the fireworks population.

Moreover, assume $X_i$ is a non-CF with a random position due to the selection method, then the generated Gaussian spark will have the similar effects with randomly generating a spark in the search range under the case $(ii)$ and case $(iii)$. Thus, the generated Gaussian sparks will not be able to improve the diversity of the fireworks swarm.

#### 4.1.3 Conventional FWA Framework versus Evolutionary Strategy (ES)

In the previous subsections, we have pointed out that the non-CFs and Gaussian sparks do not contribute much to the optimization due to the selection method in the conventional FWA framework. If we simply eliminate the Gaussian mutation operator and non-CFs in the explosion operator, we will get the *minimalist fireworks algorithm* (MFWA) using only one firework, as shown in Algorithm 6.

For comparison, we briefly describe a kind of evolutionary algorithms: *evolutionary strategy* (ES) [54]. Generally speaking, ES conducts the search process by keeping the iterations of sampling and learning. In the sampling step, a number of points are randomly sampled from a Gaussian distribution. In the learning step, the parameters of the

Gaussian distribution are updated according to the qualities of the sampled points.

**Algorithm 6.** The MFWA

1: Initialize the firework and explosion amplitude
2: **repeat**
3: Generate explosion sparks (cf. Algorithm 2)
4: Update explosion amplitude (cf. Algorithm 4 or Algorithm 5)
5: Select the best individual as the new firework
6: **until** termination criterion (time, max. # evals, convergence, ...) is met

To some extent, the iteration process of MFWA is partly similar to a $(1+\lambda)$-ES [54]: one parent generates many sons, and the sampling parameters are updated according to the qualities of them (the position and the fitness). The explosion amplitude in FWA and the variance in ES can both be regarded as the step size. The methodologies to control them are also partly similar.

The dynamic explosion amplitude strategy in dynFWA is corresponding to the 1/5 rule in $(1+1)$-ES [54]. In dynFWA, if the fireworks swarm has found a better position, then the explosion amplitude will be amplified, else the explosion amplitude will be reduced. In $(1+1)$-ES, if the successful rate is higher than $1/5$, the mutation strength $\sigma$ will be increased, otherwise, the $\sigma$ will be reduced.

The algorithm of calculating the explosion amplitude in AFWA can be replaced (in most cases) by multiplying the distance between the current firework and the previous firework with a mutation coefficient. In this way, each explosion spark actually carries a different "explosion amplitude" with it, which is mutated from the father's explosion amplitude. Although the new firework is selected because of its fitness value, the distance can also be regarded as a reasonable step size around this location (because the best individual is found using such a step size). In this sense, the principle of AFWA is comparable to that of $(1+\lambda)$-ES where the step sizes of the sons' generation are also mutated from the father's and unsuitable step sizes are also removed through selection. The common idea of these two algorithms is "the step size that created (and thus is carried by) the good individual is considered proper", which is the key of self-adaption.

So the properties of these algorithms are similar: they are all locally convergent, they all work poor on dimensionally sensitive functions, and they are all easily trapped in local minimum. After the comparison with ES, it leaves a profound insight that the new FWA framework which can utilize all the fireworks' information should be designed and the powerful and efficient cooperative mechanisms are essential for the future developments.

## 4.2 Cooperative Framework of FWA

To make FWA a successful swarm intelligence algorithm [51], the heuristic information for each firework should be passed to the next iteration and fireworks can cooperate with each other.

However, the conventional FWA framework lacks the local search ability for non-CFs, while the cooperative strategy in the Gaussian mutation operator is not very effective. To tackle these limitations, the CoFFWA with independent selection method and cooperative strategy is finally proposed.

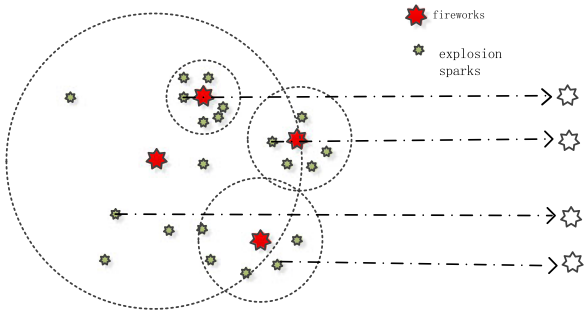



Fig. 5. The independent selection method.

**Algorithm 7.** The Cooperative Framework for FWA

1: Initialize $N$ fireworks and evaluate their fitness
2: **repeat**
3: Calculate the explosion sparks numbers and explosion amplitudes
4: **for** each firework **do**
5: Generate the explosion sparks
6: Select best candidate as firework
7: Perform the cooperative strategy
8: **end for**
9: **until** termination criterion (time, max. # evals, convergence, ...) is met

### 4.2.1 *Independent Selection Method*

The analyses in previous subsections suggest that the probability based random selection methods in FWA and its variants cause that non-CFs and non-GCFs do not contribute much to the optimization for a problem while consuming a lot of evaluation times in the explosion operator, and the cooperative scheme of Gaussian mutation operator among the fireworks is powerless to solve complex problems. Furthermore, the selection method in conventional FWA framework is seen as the main reason why the non-CFs/non-GCFs cannot evolve for a number of consecutive iterations.

To implement the initial idea of FWA that fireworks in the swarm cooperate together to solve the optimization problem, it is needed to ensure that the information for each firework is passed to the next iteration. In the new framework of FWA, the independent selection strategy will be performed for each firework, i.e., each firework will select the best candidate among its all generated sparks and itself in each iteration respectively (cf. Algorithm 7 and Fig. 5).

### 4.2.2 *The Crowdness-Avoiding Cooperative Strategy among the Fireworks*

In the fireworks swarm, each firework will generate a number of explosion sparks which can represent the quality of the local regions. The quality of each firework's position is shared in the population to accelerate the convergence speed.

For the calculation of explosion amplitudes and explosion sparks numbers in CoFFWA, CF still takes the dynamic explosion amplitude strategy, while for the rest of

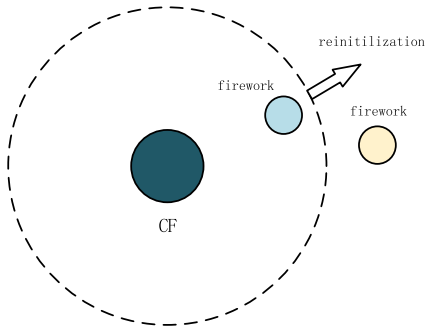



Fig. 6. The crowdness-avoiding cooperative strategy.

fireworks, the explosion amplitudes are calculated as in dynFWA by Eq. (1). For explosion sparks number calculation, it is still by Eq. (2). After the generating of explosion sparks, each firework performs the independent selection method, respectively.

In CoFFWA, a crowdness-avoiding operation is introduced in the fireworks population (see Fig. 6). For the crowdness-avoiding operation, it means that whenever a firework in the fireworks swarm is close to CF, i.e., within a fixed range of CF, then the position of this firework will be reinitialized in the feasible search space (refer to Algorithm 8). The following gives two main reasons for the crowdness-avoiding operation:

- The selected firework ($X_i$) has worse fitness and smaller explosion sparks number than the CF, thus the region that $X_i$ locates is not promising, it is seen as a waste of evaluation times to continue the search for the fireworks with worse fitness around the CF.
- To increase the diversity of the fireworks swarm, it is better that the fireworks are not located close to each other.

The upper bound of explosion amplitude for CF is set to the feasible search range value to avoid the too frequent reinitiation for non-CFs.

**Algorithm 8.** The Crowdness-Avoiding Cooperative Strategy

1: **if** $||X_i - X_{CF}||_\infty < \tau A_{CF}$ **then**
2: Reinitialize the position of $X_i$
3: **end if**

## 5 Experiments Design

To validate the ideas presented previously, several groups of experiments are designed.

### 5.1 Significance Analysis of Non-CFs/Non-GCFs in Explosion Operator

To investigate whether the non-CFs/non-GCFs are effective or not compared with the CF/GCF in the explosion operator, the evaluation criterions, significant improvement and resources cost are designed.

For these evaluation criterions, FWA variants without Gaussian mutation operator (i.e., the algorithms only generate explosion sparks) are used to offset the influences of Gaussian mutation operators.

#### 5.1.1 Significant Improvement

Among the fireworks, if a firework $X_i$ generates the explosion spark with the minimal fitness among all the explosion sparks and fireworks, then $X_i$ is seen to make one time significant improvement to the optimization. In the optimization process of FWA, at each iteration, at most one firework can make the significant improvement, thus to compare the performance of CF, GCF and non-CFs, non-GCFs, the significant improvement times of them during each run are recorded.

- $\alpha_{CF}$, the percentage of CF's significant improvement times among all the significant improvement times.
- $\beta_{CF}$, the percentage of CF's significant improvement times among all the significant improvement times recorded from the $\frac{1}{30}E_{\max}$th evaluation time.
- $\alpha_{GCF}$, the percentage of GCF's significant improvement times among all the significant improvement times.
- $\beta_{GCF}$, the percentage of GCF's significant improvement times among all the significant improvement times recorded from the $\frac{1}{30}E_{\max}$th evaluation time.

Here, $E_{max}$ denotes the max number of evaluation times. It is thought that $\beta_{CF}$, $\beta_{GCF}$ are better than $\alpha_{CF}$, $\alpha_{GCF}$ for the performance comparison as at the early phase of the optimization, it is likely that non-CFs gain more significant improvement times due to the great number of explosion sparks. However, significant improvements in the later phase of the optimization are more important.

#### 5.1.2 Resources Cost

For the optimization, $E_{\max}$ is usually set to $D \times 10{,}000$, where $D$ is the dimension of the problem [55].

- $\theta_{CF}$, the percentage of CF's evaluation times among all the evaluation times.
- $\theta_{GCF}$, the percentage of GCF's evaluation times among all the evaluation times.

### 5.2 Significance Analysis of Gaussian Mutation Operator

To validate whether the Gaussian mutation operator introduced in EFWA is effective or not, the following comparisons are made.

- EFWA-G versus EFWA-NG,
- dynFWA-G versus dynFWA-NG,
- AFWA-G versus AFWA-NG,

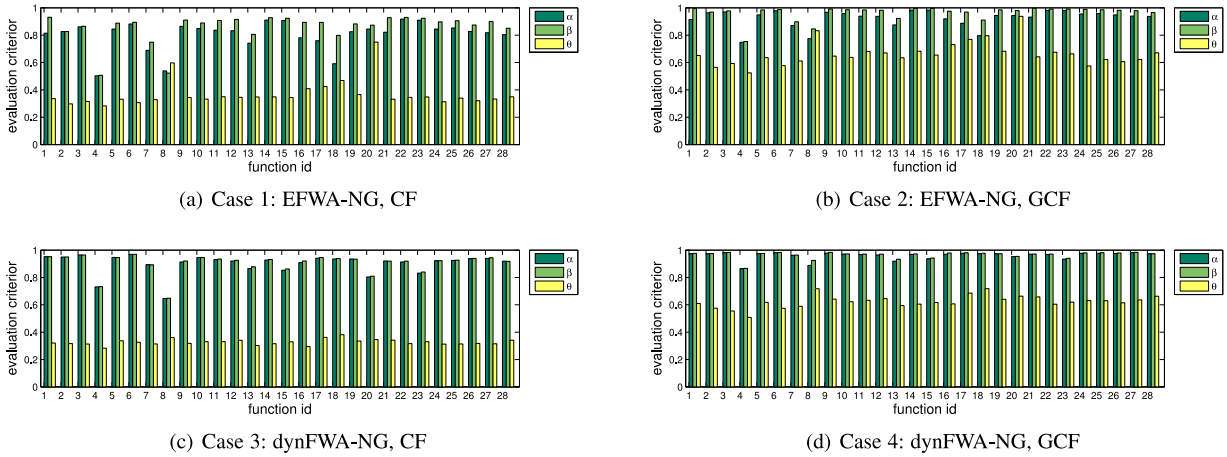
here, "G" and "NG" refer to with and without Gaussian mutation operator, respectively.

### 5.3 Significance Analysis of the Proposed CoFFWA

To validate the performance of the proposed CoFFWA, experiments are conducted to compare the performance of CoFFWA with EFWA [25], dynFWA [26], AFWA [27], artificial bee colony (ABC) [8], [56], differential evolution [57], SPSO2007 [58], and the most recent SPSO2011 [59].

### 5.4 Experimental Setup and Platform

Similar to dynFWA, the number of fireworks in CoFFWA is set to 5 and the explosion sparks number is set to 150.




(a) Case 1: EFWA-NG, CF

(b) Case 2: EFWA-NG, GCF

(c) Case 3: dynFWA-NG, CF

(d) Case 4: dynFWA-NG, GCF

Fig. 7. Significant improvement and resources cost results of CF and GCF.

The reduction and amplification factors $C_r$ and $C_a$ are also empirically set to $0.9$ and $1.2$, respectively. For the parameter $\tau$, it is set to 10, and the maximum explosion amplitude for CF is bounded with the search range, which is set to 200 in CEC2013 competition problems. For the rest of parameters in CoFFWA, they are identical to dynFWA [26]. The parameters for AFWA, dynFWA, DE, ABC, SPSO2007 and SPSO2011 are listed in [8], [26], [27], [57], [58], [60], respectively.

In the experiments, the CEC2013 benchmark functions with 28 functions are used as the test suite and dimension is set to 30 [55]. For the recorded results in each algorithm, the number of runs is set to 51, and Wilcoxon signed-rank test is used to validate the performance improvement. The experimental platform is MATLAB 2011b (Windows 7; Intel Core i7-2600 CPU @ 3.7 GHZ; 8 GB RAM).

## 6 Experimental Results

### 6.1 Significance Analysis of Non-CFs/Non-GCFs in Explosion Operator

To validate the performance of CF and GCF, the $\alpha_{CF}$, $\beta_{CF}$, $\theta_{CF}$ and $\alpha_{GCF}$, $\beta_{GCF}$, $\theta_{GCF}$ are calculated on EFWA-NG and dynFWA-NG respectively, and the recorded results on 28 functions can be found in Fig. 7.

Compare the performance between CF and non-CFs, it can be seen that, for both EFWA-NG and dynFWA-NG, CF takes smaller percentage of resources, but makes the more significant improvement times to the search.

Compare the evaluation criterion $\alpha_{CF}$ and $\beta_{CF}$, it can be seen that for all functions, $\beta_{CF}$ which records the statistical results from the $\frac{1}{30}E_{\max}$th evaluation times is higher than $\alpha_{CF}$, which means that at the beginning of the optimization, the non-CFs have high probability for making significant improvements, while at the later phase, the chance goes smaller. For GCF, the above situation is similar to CF.

Compare the results of CF with GCF, GCF makes more significant improvements while taking more resources, due to that some fireworks except for CF may locate close to CF, thus to have a high chance to make improvements.

In summary, from the results, it can be concluded that the CF makes much more contributions than non-CFs while taking smaller resources, and the non-GCFs seem to make almost no contributions to the optimization.

### 6.2 Significance Analysis of Gaussian Mutation Operator

Table 1 gives the experimental results of the versions of EFWA, dynFWA, AFWA with and without Gaussian mutation operator. Comparing EFWA with EFWA-NG, it can be seen that EFWA-G performs significantly better than EFWA-NG on four functions, which suggests that Gaussian mutation operator is effective to improve the performance

TABLE 1
Wilcoxon Signed-Rank Test Results for EFWA-G versus EFWA-NG and dynFWA-G versus dynFWA-NG and AFWA-G *versus* AFWA-NG (Bold Values Indicate the Performance Difference is Significant, while 1/0/-1 Denotes the Version with Gaussian Sparks Operator is Significant Better / Not Significant Different / Significant Worse than the Version without Gaussian Sparks)

| F. | EFWA-G *versus* EFWA-NG | | dynFWA-G *versus* dynFWA-NG | | AFWA-G *versus* AFWA-NG | |
|---|---|---|---|---|---|---|
| 1 | **2.316E-03** | 1 | 1.000E+00 | 0 | 1.000E+00 | 0 |
| 2 | 4.256E-01 | 0 | 9.328E-01 | 0 | 4.647E-01 | 0 |
| 3 | 8.956E-01 | 0 | 2.339E-01 | 0 | 7.191E-02 | 0 |
| 4 | 7.858E-01 | 0 | 7.492E-02 | 0 | **5.689E-03** | -1 |
| 5 | **4.290E-02** | 1 | 7.646E-02 | 0 | 5.239E-01 | 0 |
| 6 | 1.654E-01 | 0 | 7.858E-01 | 0 | 9.030E-01 | 0 |
| 7 | 9.552E-01 | 0 | 6.869E-01 | 0 | 2.728E-01 | 0 |
| 8 | 9.776E-01 | 0 | 4.704E-01 | 0 | 8.808E-01 | 0 |
| 9 | 5.178E-01 | 0 | 4.997E-01 | 0 | 7.571E-01 | 0 |
| 10 | 3.732E-01 | 0 | 5.057E-01 | 0 | **9.545E-03** | -1 |
| 11 | 5.830E-02 | 0 | 6.629E-01 | 0 | 3.204E-01 | 0 |
| 12 | 6.193E-01 | 0 | 3.783E-01 | 0 | 1.801E-01 | 0 |
| 13 | 8.220E-01 | 0 | 1.863E-01 | 0 | 4.590E-01 | 0 |
| 14 | 4.101E-02 | 0 | 3.834E-01 | 0 | 5.239E-01 | 0 |
| 15 | 6.869E-01 | 0 | 3.438E-01 | 0 | 4.879E-01 | 0 |
| 16 | 2.811E-01 | 0 | 4.256E-01 | 0 | 8.220E-01 | 0 |
| 17 | 9.179E-01 | 0 | 1.863E-01 | 0 | 2.339E-01 | 0 |
| 18 | 6.938E-01 | 0 | 4.762E-01 | 0 | 6.460E-01 | 0 |
| 19 | 9.402E-01 | 0 | 1.542E-01 | 0 | 7.786E-01 | 0 |
| 20 | **1.559E-02** | -1 | 5.830E-02 | 0 | 4.997E-01 | 0 |
| 21 | **6.910E-04** | 1 | 7.997E-01 | 0 | 5.937E-01 | 0 |
| 22 | 9.776E-01 | 0 | 3.583E-01 | 0 | 4.202E-01 | 0 |
| 23 | 7.217E-01 | 0 | 7.642E-01 | 0 | 6.260E-01 | 0 |
| 24 | **1.079E-02** | 1 | 5.486E-01 | 0 | 7.500E-01 | 0 |
| 25 | 8.734E-01 | 0 | 2.091E-01 | 0 | **1.369E-02** | -1 |
| 26 | 2.687E-01 | 0 | 7.217E-01 | 0 | 3.834E-01 | 0 |
| 27 | 3.534E-01 | 0 | 8.734E-01 | 0 | 1.597E-01 | 0 |
| 28 | 6.460E-01 | 0 | **0.000E+00** | -1 | **3.285E-03** | -1 |

TABLE 2
Mean Fitness and Mean Fitness Rank of ABC, DE, SPSO2007, SPSO2011, EFWA, AFWA, dynFWA, and CoFFWA (AR Denotes the Average of Mean Rank Value)

| F. | ABC | | DE | | SPSO2007 | | SPSO2011 | | EFWA | | AFWA | | dynFWA | | CoFFWA | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.00E+00 | 1 | 1.89E-03 | 7 | 0.00E+00 | 1 | 0.00E+00 | 1 | 8.50E-02 | 8 | 0.00E+00 | 1 | 0.00E+00 | 1 | 0.00E+00 | 1 |
| 2 | 6.20E+06 | 8 | 5.52E+04 | 1 | 6.08E+06 | 7 | 3.38E+05 | 2 | 5.85E+05 | 3 | 8.92E+05 | 6 | 8.71E+05 | 4 | 8.80E+05 | 5 |
| 3 | 5.74E+08 | 7 | 2.16E+06 | 1 | 6.63E+08 | 8 | 2.88E+08 | 6 | 1.16E+08 | 3 | 1.26E+08 | 5 | 1.23E+08 | 4 | 8.04E+07 | 2 |
| 4 | 8.75E+04 | 7 | 1.32E-01 | 1 | 1.03E+05 | 8 | 3.86E+04 | 6 | 1.22E+00 | 2 | 1.14E+01 | 4 | 1.04E+01 | 3 | 2.01E+03 | 5 |
| 5 | 0.00E+00 | 1 | 2.48E-03 | 7 | 0.00E+00 | 2 | 5.42E-04 | 3 | 8.05E-02 | 8 | 6.00E-04 | 5 | 5.51E-04 | 4 | 7.41E-04 | 6 |
| 6 | 1.46E+01 | 2 | 7.82E+00 | 1 | 2.52E+01 | 4 | 3.79E+01 | 8 | 3.22E+01 | 7 | 2.99E+01 | 5 | 3.01E+01 | 6 | 2.47E+01 | 3 |
| 7 | 1.25E+02 | 7 | 4.89E+01 | 1 | 1.13E+02 | 6 | 8.79E+01 | 2 | 1.44E+02 | 8 | 9.19E+01 | 4 | 9.99E+01 | 5 | 8.99E+01 | 3 |
| 8 | 2.09E+01 | 6 | 2.09E+01 | 2 | 2.10E+01 | 7 | 2.09E+01 | 5 | 2.10E+01 | 8 | 2.09E+01 | 4 | 2.09E+01 | 3 | 2.09E+01 | 1 |
| 9 | 3.01E+01 | 8 | 1.59E+01 | 1 | 2.93E+01 | 6 | 2.88E+01 | 5 | 2.98E+01 | 7 | 2.48E+01 | 4 | 2.41E+01 | 3 | 2.40E+01 | 2 |
| 10 | 2.27E-01 | 5 | 3.24E-02 | 1 | 2.38E-01 | 6 | 3.40E-01 | 7 | 8.48E-01 | 8 | 4.73E-02 | 3 | 4.81E-02 | 4 | 4.10E-02 | 2 |
| 11 | 0.00E+00 | 1 | 7.88E+01 | 3 | 6.26E+01 | 2 | 1.05E+02 | 7 | 2.79E+02 | 8 | 1.05E+02 | 6 | 1.04E+02 | 5 | 9.90E+01 | 4 |
| 12 | 3.19E+02 | 7 | 8.14E+01 | 1 | 1.15E+02 | 3 | 1.04E+02 | 2 | 4.06E+02 | 8 | 1.52E+02 | 5 | 1.58E+02 | 6 | 1.40E+02 | 4 |
| 13 | 3.29E+02 | 7 | 1.61E+02 | 1 | 1.79E+02 | 2 | 1.94E+02 | 3 | 3.51E+02 | 8 | 2.36E+02 | 4 | 2.54E+02 | 6 | 2.50E+02 | 5 |
| 14 | 3.58E-01 | 1 | 2.38E+03 | 3 | 1.59E+03 | 2 | 3.99E+03 | 7 | 4.02E+03 | 8 | 2.97E+03 | 5 | 3.02E+03 | 6 | 2.70E+03 | 4 |
| 15 | 3.88E+03 | 4 | 5.19E+03 | 8 | 4.31E+03 | 7 | 3.81E+03 | 3 | 4.28E+03 | 6 | 3.81E+03 | 2 | 3.92E+03 | 5 | 3.37E+03 | 1 |
| 16 | 1.07E+00 | 5 | 1.97E+00 | 8 | 1.27E+00 | 6 | 1.31E+00 | 7 | 5.75E-01 | 3 | 4.97E-01 | 2 | 5.80E-01 | 4 | 4.56E-01 | 1 |
| 17 | 3.04E+01 | 1 | 9.29E+01 | 2 | 9.98E+01 | 3 | 1.16E+02 | 5 | 2.17E+02 | 8 | 1.45E+02 | 7 | 1.43E+02 | 6 | 1.10E+02 | 4 |
| 18 | 3.04E+02 | 8 | 2.34E+02 | 7 | 1.80E+02 | 4 | 1.21E+02 | 1 | 1.72E+02 | 2 | 1.75E+02 | 3 | 1.88E+02 | 6 | 1.80E+02 | 5 |
| 19 | 2.62E-01 | 1 | 4.51E+00 | 2 | 6.48E+00 | 3 | 9.51E+00 | 7 | 1.24E+01 | 8 | 6.92E+00 | 5 | 7.26E+00 | 6 | 6.51E+00 | 4 |
| 20 | 1.44E+01 | 6 | 1.43E+01 | 5 | 1.50E+01 | 8 | 1.35E+01 | 4 | 1.45E+01 | 7 | 1.30E+01 | 1 | 1.33E+01 | 3 | 1.32E+01 | 2 |
| 21 | 1.65E+02 | 1 | 3.20E+02 | 6 | 3.35E+02 | 8 | 3.09E+02 | 3 | 3.28E+02 | 7 | 3.16E+02 | 5 | 3.10E+02 | 4 | 2.06E+02 | 2 |
| 22 | 2.41E+01 | 1 | 1.72E+03 | 2 | 2.98E+03 | 3 | 4.30E+03 | 7 | 5.15E+03 | 8 | 3.45E+03 | 6 | 3.33E+03 | 5 | 3.32E+03 | 4 |
| 23 | 4.95E+03 | 5 | 5.28E+03 | 6 | 6.97E+03 | 8 | 4.83E+03 | 4 | 5.73E+03 | 7 | 4.70E+03 | 2 | 4.75E+03 | 3 | 4.47E+03 | 1 |
| 24 | 2.90E+02 | 7 | 2.47E+02 | 1 | 2.90E+02 | 6 | 2.67E+02 | 2 | 3.05E+02 | 8 | 2.70E+02 | 4 | 2.73E+02 | 5 | 2.68E+02 | 3 |
| 25 | 3.06E+02 | 6 | 2.80E+02 | 1 | 3.10E+02 | 7 | 2.99E+02 | 5 | 3.38E+02 | 8 | 2.99E+02 | 4 | 2.97E+02 | 3 | 2.94E+02 | 2 |
| 26 | 2.01E+02 | 1 | 2.52E+02 | 3 | 2.57E+02 | 4 | 2.86E+02 | 7 | 3.02E+02 | 8 | 2.73E+02 | 6 | 2.61E+02 | 5 | 2.13E+02 | 2 |
| 27 | 4.16E+02 | 1 | 7.64E+02 | 2 | 8.16E+02 | 3 | 1.00E+03 | 7 | 1.22E+03 | 8 | 9.72E+02 | 5 | 9.80E+02 | 6 | 8.71E+02 | 4 |
| 28 | 2.58E+02 | 1 | 4.02E+02 | 5 | 6.92E+02 | 7 | 4.01E+02 | 4 | 1.23E+03 | 8 | 4.37E+02 | 6 | 2.96E+02 | 3 | 2.84E+02 | 2 |
| | AR: 4.14 | | AR: 3.18 | | AR: 5.04 | | AR: 4.64 | | AR: 6.79 | | AR: 4.25 | | AR: 4.43 | | **AR: 3.00** | |

for EFWA. However, for dynFWA-NG and AFWA-NG, the versions without Gaussian mutation operator have better performance. Moreover, in terms of the computational complexity, our previous results in [26] suggest that Gaussian mutation operator is more time consuming than explosion sparks operator for generating one spark. Thus, the Gaussian mutation operator will be removed from CoFFWA.

### 6.3 Significance Analysis of CoFFWA

Table 2 shows the mean fitness value over 51 runs of the 28 functions for ABC, DE, SPSO2007, SPSO2011, EFWA, AFWA, dynFWA and CoFFWA and the corresponding rank of each algorithm.[4] It can be seen that CoFFWA is able to achieve better results than dynFWA(AFWA) on 24(22) functions while dynFWA(AFWA) is better than CoFFWA on 3 (5) functions. For 1(1) function, the results are identical. For the comparison with EFWA, the advantage is especially obvious. Moreover, as for the performance comparison of each function, it can be seen that CoFFWA gains more advantages for basic multimodal functions ($f_6$-$f_{19}$) and composition functions ($f_{20}$-$f_{28}$) while for unimodal functions ($f_1$-$f_5$), the advantages are not evident.

The Wilcoxon signed-rank test results in Table 3 suggest that CoFFWA is significantly better than dynFWA on seven functions while significant worse on two functions.

Compared with ABC, DE,[5] SPSO2007, SPSO2011, EFWA, AFWA and dynFWA in terms of mean fitness rank, it can be seen that CoFFWA achieves the best optimization results, i.e., the minimal average of mean fitness rank (3.00 in Table 2), thus we can conclude that the proposed cooperative framework for FWA is significant.

## 7 Conclusion

In this paper, we have presented a cooperative framework for FWA. The contributions include three aspects.

1) The cooperative strategies in conventional FWA are analyzed and evaluated. In the explosion operator, the performance of CF, GCF, non-CFs and non-GCFs in EFWA, dynFWA and AFWA are investigated and the designed criterions are evaluated to support our opinion that the non-CFs and non-GCFs do not contribute much to the optimization while taking a lot of evaluation times. For the Gaussian mutation operator, experimental results suggest that it is not as effective as it was designed to be.

4. The results of ABC, DE, SPSO2011 on CEC2013 competition problems were reported in [8], [57] and [60] respectively. In addition, the detailed results which include the results of each single run are available at http://goo.gl/pXB1WH. However, some precisions of DE are not sufficient due to its incomplete data. As a result, the rankings on function 9 may be not correct.

5. In the experiments, the mean ranks between DE and CoFFWA are quite close with respect to different runs as they are heuristic algorithms.

TABLE 3
Wilcoxon Signed-Rank Test Results for CoFFWA versus dynFWA ( 1/0/-1 Denotes the CoFFWA is Significant Better / Not Significant Different / Significant Worse than dynFWA)

| F. | $p$ | H | F. | $p$ | H |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 15 | 0.000089 | 1 |
| 2 | 0.646019 | 0 | 16 | 0.115316 | 0 |
| 3 | 0.932769 | 0 | 17 | 0.000003 | 1 |
| 4 | 0 | −1 | 18 | 0.414791 | 0 |
| 5 | 0 | −1 | 19 | 0.256715 | 0 |
| 6 | 0.398886 | 0 | 20 | 0.625961 | 0 |
| 7 | 0.078034 | 0 | 21 | 0.000001 | 1 |
| 8 | 0.707706 | 0 | 22 | 0.851293 | 0 |
| 9 | 0.985043 | 0 | 23 | 0.119709 | 0 |
| 10 | 0.298129 | 0 | 24 | 0.252805 | 0 |
| 11 | 0.241324 | 0 | 25 | 0.378261 | 0 |
| 12 | 0.07963 | 0 | 26 | 0.002316 | 1 |
| 13 | 0.586675 | 0 | 27 | 0.005528 | 1 |
| 14 | 0.008676 | 1 | 28 | 0.001594 | 1 |
| | | 7+/19/2- | | | |

2) Based on the analysis of cooperative strategies in the conventional FWA framework, a cooperative framework of FWA with an independent selection method and crowdness-avoiding cooperative strategy is finally proposed to ensure the information inheritance and to improve the diversity of fireworks population.
3) Moreover, the explosion amplitude strategies in AFWA and dynFWA are compared and the relationship between them is discussed.

Compared with the search manner of typical swarm intelligence algorithms, like particle swarm optimization [4], artificial bee colony [8], [56], fish school search algorithm[10], firefly algorithm [11], [12], et al., FWA presents a differentiated explosive search manner—*each individual of the swarm generates different numbers of individuals* rather than these typical search manners—*each individual of the swarm generates one individual*. The explosive search manner has the ability to estimate the property of local search space, which makes the estimated search direction of the swarm stable. The differentiated resources allocation strategy among the fireworks will make the firework with better fitness have high sampling probability, which makes the balance between exploration and exploitation capabilities.

In the future, we will focus on this unique search manner and try to develop new kinds of cooperative search strategies among the fireworks swarm to further enhance the exploration and exploitation capabilities.

## Acknowledgments

This work was supported by the Natural Science Foundation of China (NSFC) under grant no. 61375119 and 61170057, and partially supported by National Key Basic Research Development Plan (973 Plan) Project of China with grant no. 2015CB352302. Y. Tan is the corresponding author.

## References

[1] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, vol. 1. Chichester, U.K.: Wiley, 2005.
[2] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in *Robots and Biological Systems: Towards a New Bionics?* New York, NY, USA: Springer, 1993, pp. 703–712.
[3] A. Colorni, M. Dorigo, V. Maniezzo, et al., "Distributed optimization by ant colonies," vol. 42, pp. 134–142, 1991.
[4] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Human Sci.*, New York, NY, 1995, vol. 1, pp. 39–43.
[5] M. Dorigo and T. Stützle, "An experimental study of the simple ant colony optimization algorithm," in *Proc. WSES Int. Conf. Evol. Comput.*, 2001, pp. 253–258.
[6] D. Karaboga. (2010). Artificial bee colony algorithm. *5(3)*, p. 6915 [Online]. Available: http://scholarpedia.org/article/Artificial_bee_colony_algorithm
[7] X.-S. Yang, "Engineering optimizations via nature-inspired virtual bee algorithms," in *Proc. 1st Int. Work-Conf. Interplay Between Natural Artif. Comput. Artif. Intell. Knowl. Eng. Appl.: Bioinspired Approach*, 2005, pp. 317–323.
[8] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Eng. Faculty, Comput. Eng. Dept., Erciyes Univ., Kayseri, Turkey, Tech. Rep. tr06, 2005.
[9] K. Krishnanand and D. Ghose, "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions," *Swarm Intell.*, vol. 3, no. 2, pp. 87–124, 2009.
[10] C. J. Bastos Filho, F. B. de Lima Neto, A. J. Lins, A. I. Nascimento, and M. P. Lima, "Fish school search," in *Nature-Inspired Algorithms for Optimisation*. New York, NY, USA: Springer, 2009, pp. 261–277.
[11] S. Łukasik and S. Żak, "Firefly algorithm for continuous constrained optimization tasks," in *Proc. 1st Int. Conf. Comput. Collective Intell. Semantic Web, Social Netw. Multiagent Syst.*, 2009, pp. 97–106.
[12] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *Int. J. Bio-Inspired Comput.*, vol. 2, no. 2, pp. 78–84, 2010.
[13] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Proc. World Congr. Nature Biologically Inspired Comput.*, 2009, pp. 210–214.
[14] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Commun. Nonlinear Sci. Numerical Simul.*, vol. 17, no. 12, pp. 4831–4845, 2012.
[15] Y. Shi, "Brain storm optimization algorithm," in *Proc. 2nd Int. Conf. Adv. Swarm Intell.*, 2011, vol. 6728, pp. 303–309.
[16] C. R. Blomeke, S. J. Elliott, and T. M. Walter, "Bacterial survivability and transferability on biometric devices," in *Proc. 41st Annu. IEEE Int. Carnahan Conf. Security Technol.*, 2007, pp. 80–84.
[17] X.-S. Yang, "A new metaheuristic Bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization*. Berlin, Germany: Springer, 2010, pp. 65–74.
[18] N. Tayarani and M. Akbarzadeh-T, "Magnetic optimization algorithms a new synthesis," in *Proc. IEEECong. Evolutionary Comput. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 2659–2664.
[19] H. Shah-Hosseini, "The intelligent water drops algorithm: A nature-inspired swarm-based optimization algorithm," *Int. J. Bio-Inspired Comput.*, vol. 1, no. 1, pp. 71–79, 2009.
[20] Y. Shi, "Brain storm optimization algorithm," in *Proc. 2nd Int. Conf. Adv. Swarm Intell.*, 2011, vol. 6728, pp. 303–309.
[21] B. Xing and W.-J. Gao, "Emerging biology-based CI algorithms," in *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*. Berlin, Germany: Springer, 2014, pp. 217–317.
[22] J. Liu, S. Zheng, and Y. Tan, "Analysis on global convergence and time complexity of fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, 2014, pp. 3207–3213.
[23] Y. Pei, S. Zheng, Y. Tan, and T. Hideyuki, "An empirical study on influence of approximation approaches on enhancing fireworks algorithm," in *Proc. IEEE Congr. Syst., Man Cybern.*, 2012, pp. 1322–1327.
[24] J. Liu, S. Zheng, and Y. Tan, "The improvement on controlling exploration and exploitation of firework algorithm," in *Proc. 4th Int. Conf. Adv. Swarm Intell.*, 2013, pp. 11–23.
[25] S. Zheng, A. Janecek, and Y. Tan, "Enhanced fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, 2013, pp. 2069–2077.
[26] S. Zheng, A. Janecek, J. Li, and Y. Tan, "Dynamic search in fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2014, pp. 3222–3229.
[27] J. Li, S. Zheng, and Y. Tan, "Adaptive fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2014, pp. 3214–3221.
[28] S. Zheng, L. Liu, C. Yu, J. Li, and Y. Tan, "Fireworks algorithm and its variants for solving ICSI2014 competition problems," in *Proc. 5th Int. Conf. Adv. Swarm Intell.*, 2014, pp. 442–451.

[29] H. Gao and M. Diao, "Cultural firework algorithm and its application for digital filters design," *Int. J. Model., Identification Control*, vol. 14, no. 4, pp. 324–331, 2011.
[30] Y.-J. Zheng, X.-L. Xu, H.-F. Ling, and S.-Y. Chen, "A hybrid fireworks optimization method with differential evolution operators," *Neurocomputing*, vol. 148, pp. 75–82, 2015.
[31] B. Zhang, M.-X. Zhang, and Y.-J. Zheng, "A hybrid biogeography-based optimization and fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2014, pp. 3200–3206.
[32] C. Yu, J. Li, and Y. Tan, "Improve enhanced fireworks algorithm with differential mutation," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2014, pp. 264–269.
[33] N. Bacanin, M. Tuba, and M. Beko, "Hybridized fireworks algorithm for global optimization," *Math. Methods Syst. Sci. Eng.*, pp. 108–114, 2015.
[34] Y.-J. Zheng, Q. Song, and S.-Y. Chen, "Multiobjective fireworks optimization for variable-rate fertilization in oil crop production," *Appl. Soft Comput.*, vol. 13, no. 11, pp. 4253–4263, 2013.
[35] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," *Comput. Eng. Netw. Lab., Swiss Federal Instit. Technol.*, Tech. Rep. 103, 2001.
[36] J. Hájek, A. Szöllös, and J. Šístek, "A new mechanism for maintaining diversity of pareto archive in multi-objective optimization," *Adv. Eng. Softw.*, vol. 41, no. 7, pp. 1031–1057, 2010.
[37] L. Liu, S. Zheng, and Y. Tan, "S-metric based Multi-objective fireworks algorithm," in *Proc. IEEE Congr. Evol. Comput.*, 2015, pp. 1257–1264.
[38] K. Ding, S. Zheng, and Y. Tan, "A GPU-based parallel fireworks algorithm for optimization," in *Proc. 15th Annu. Conf. Genetic Evol. Comput. Conf.*, 2013, pp. 9–16.
[39] A. Janecek and Y. Tan, "Using population based algorithms for initializing nonnegative matrix factorization," in *Proc. 2nd Int. Conf. Adv. Swarm Intell.*, 2011, pp. 307–316.
[40] A. Janecek and Y. Tan, "Iterative improvement of the multiplicative update NMF algorithm using nature-inspired optimization," in *Proc. 7th Int. Conf. Natural Comput.*, 2011, vol. 3, pp. 1668–1672.
[41] A. Janecek and Y. Tan, "Swarm intelligence for non-negative matrix factorization," *Int. J. Swarm Intell. Res.*, vol. 2, no. 4, pp. 12–34, 2011.
[42] W. He, G. Mi, and Y. Tan, "Parameter optimization of local-concentration model for spam detection by using fireworks algorithm," in *Proc. 4th Int. Conf. Adv. Swarm Intell.*, 2013, pp. 439–450.
[43] S. Zheng and Y. Tan, "A unified distance measure scheme for orientation coding in identification," in *Proc. IEEE Congr. Inf. Sci. Technol.*, 2013, pp. 979–985.
[44] A. M. Imran and M. Kowsalya, "A new power system reconfiguration scheme for power loss minimization and voltage profile enhancement using fireworks algorithm," *Int. J. Electr. Power Energy Syst.*, vol. 62, pp. 312–322, 2014.
[45] N. Pholdee and S. Bureerat, "Comparative performance of meta-heuristic algorithms for mass minimisation of trusses with dynamic constraints," *Adv. Eng. Softw.*, vol. 75, pp. 1–13, 2014.
[46] S. Bureerat, "Hybrid population-based incremental learning using real codes," in *Proc. 5th Int. Conf. Learn. Intell. Optimization*, 2011, pp. 379–391.
[47] Z. Du, "Fireworks algorithm for solving non-linear equation set (in chinese)," *Modem Comput.*, vol. 3, pp. 19–21, 2013.
[48] Z. Du, "Fireworks algorithm for solving 0/1 knapsack problem (in chinese)," *J. Wuhan Eng. Inst.*, vol. 3, pp. 61–63, 2011.
[49] Y. Tan, *Fireworks Algorithm*. New York, NY, USA: Springer, 2015.
[50] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *Trans. Evol. Comp*, vol. 6, no. 1, pp. 58–73, 2002.
[51] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*. New York, NY, USA: Wiley, 2006.
[52] G. Lu, D. Tan, and H. Zhao, "Improvement on regulating definition of antibody density of immune algorithm," in *Proc. 9th Int. Conf. Neural Inf. Process.*, 2002, vol. 5, pp. 2669–2672.
[53] R. C. Eberhart and Y. Shi, "Comparison between genetic algorithms and particle swarm optimization," in *Proc. 7th Int. Conf. Evol. Program. VII*, 1998, pp. 611–616.
[54] H.-G. Beyer and H.-P. Schwefel, "Evolution Strategies—a comprehensive introduction," *Natural Comput.*, vol. 1, no. 1, pp. 3–52, 2002.
[55] J. Liang, B. Qu, P. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," *Comput. Intell. Lab., Zhengzhou Univ.*, Zhengzhou China And Tech. Rep., Nanyang Technol. Univ., Singapore, Tech. Rep. 201212, 2013.
[56] M. El-Abd, "Testing a particle swarm optimization and artificial bee colony hybrid algorithm on the CEC 13 benchmarks," in *Proc. IEEE Congr. Evol. Comput.*, 2013, pp. 2215–2220.
[57] N. Padhye, P. Mittal, and K. Deb, "Differential evolution: Performances and analyses," in *Proc. IEEE Congr. Evol. Comput.*, 2013, pp. 1960–1967.
[58] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. Swarm Intell. Symp.*, 2007, pp. 120–127.
[59] (2012). Standard particle swarm optimisation. Particle Swarm Central, Tech. Rep., pp. 1–15 [Online]. Available: http://clerc.maurice.free.fr/pso/SPSO descriptions.pdf
[60] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas, "Standard particle swarm optimisation 2011 at CEC 2013: A baseline for future PSO improvements," in *Proc. IEEE Congr. Evol. Comput.*, 2013, pp. 2337–2344.

**Shaoqiu Zheng** received the BEng degree from the Department of Acoustic, Northwestern Polytechnical University, in June 2010, and the PhD degree from the Department of Computer Science and Technology, Peking University, in June 2015. He is currently an engineer at the 28th Research Institute of China Electronics Technology Group Corporation. His research interests include evolutionary computation, swarm intelligence, machine learning, biometrics, and pattern recognition. He is a student member of the IEEE.

**Junzhi Li** received the BS degree from the Department of Machine Intelligence and minor degree of philosophy from the Department of Philosophy and of Religious Studies in 2013 from Peking University. He is currently working toward the PhD degree in the Key Laboratory of Machine Perception (Ministry of Education) and the Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. His research interests include evolutionary computation, artificial neural networks, and machine learning. He is a student member of the IEEE.

**Andreas Janecek** received the PhD degree in computer science in 2010 from the University of Vienna. From 2010 to 2012, he was a postdoctoral researcher at the Peking University, China, and the Universitade Politecnica de Pernambuco in Recife, Brazil. His main research activities are in the areas of data mining / machine learning, and (nature-inspired) computational intelligence techniques. He has been very active in combining these disciplines, which has resulted in several interdisciplinary research activities. Moreover, he is very interested in cellular floating car data applications for road traffic monitoring. He has published more than 30 peer-reviewed publications, including several book-chapters, and he has received the Best Paper awards of IC-SI 2011 and 2013 (International Conference on Swarm Intelligence).

**Ying Tan** (M'98-SM'02) received the BEng, MS, and PhD degrees from Southeast Univ., in 1985, 1988, and 1997, respectively. He is a full professor and a PhD advisor in the School of Electronics Engineering and Computer Science, Peking University, and the director in Computational Intelligence Laboratory, Peking University (PKU). He is the inventor of Fireworks Algorithm (FWA). He serves as the editor-in-chief of the *International Journal of Computational Intelligence and Pattern Recognition (IJCIPR)* and the associate editor of the *IEEE Transactions on Cybernetics (Cyb), IEEE Transactions on Neural Networks and Learning Systems (NNLS)*, *International Journal of Artificial Intelligence (IJAI)*, *International Journal of Swarm Intelligence Research (IJSIR)*, etc. He also served as an editor of Springer's Lecture Notes on Computer Science (LNCS) for more than 12 volumes, and guest editor of several referred journals, including *Information Science, Softcomputing, Neurocomputing*, *IJAI*, *IJSIR*, *B&B*, *CJ*, and *IEEE/ACM Transactions on Computational Biology and Bioinformatics (IEEE/ACM TCBB)*. He is a member of the Emergent Technologies Technical Committee (ETTC), Computational Intelligence Society of IEEE since 2010. He is the founder and the chair of the ICSI International Conference series. He was the joint general chair of the first and second BRICS CCI, program committee co-chair of WCCI 2014, etc. His research interests include computational intelligence, swarm intelligence, data mining, pattern recognition, and intelligent information processing for information security. He has published more than 260 papers in refereed journals and conferences in these areas, and authored/co-authored six books and 10 chapters in book, and received three invention patents. He is a senior member of the IEEE and ACM and a senior member of the CIE.