Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(4): 7234–7252.
DOI: 10.3934/mbe.2023313
Received: 22 December 2022
Revised: 10 January 2023
Accepted: 10 January 2023
Published: 10 February 2023

Research article

A triple-spark guiding strategy to enhance the loser-out tournament-based fireworks algorithm

Sicheng Li, Junhao Zhu, Mingzhang Han, Mingjie Fan and Xinchao Zhao*

School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

* **Correspondence:** Email: zhaoxc@bupt.edu.cn; Tel: 8618710137352.

Abstract: Loser-out tournament-based fireworks algorithm (LoTFWA) is a new baseline of fireworks algorithm (FWA). However, its ability to search deeply in local areas and communicate among fireworks is not satisfying enough. Therefore, this paper proposes a new triple-spark guiding strategy for LoTFWA to deal with the mentioned problems. Among the three sparks generated for guiding, one is exactly the same as the original one in LoTFWA, the second one uses the centroid of good sparks to enhance the local exploitation, and the last one is based on Differential evolution (DE) mutation and used to enhance cooperation and exploration. Experimental results show that with low computational cost, the proposed guiding strategy attains significantly better results than LoTFWA and is competitive with state-of-the-art FWA variants. Furthermore, comprehensive experiments show that the proposed strategy has the potential to combine with other FWA variants to achieve better results.

Keywords: single-object optimization; fireworks algorithm; loser-out tournament; differential evolution; guiding strategy

1. Introduction

The FWA [1] is an optimization algorithm inspired by the phenomenon of fireworks explosion. The fireworks in FWA interchange information to dynamically control the resource allocation and the search manner.

Due to its impressive performance in black box optimization (BBO) problems, variants of FWA have been successfully applied to many kinds of real-world problems, including blind source separation in radar signals [2], electromagnetic optimization problem [3], person re-identification [4], traffic flow prediction [5], multi-object and multi-domain interference system [6], energy efficiency of metro railway line [7], Welding robot path planning [8], task scheduling in fog system [9], medical image registration [10], optimizing the dispatch of controllable units in microgrid [11].

DE [12] is another optimization algorithm that has been shown to be simple yet efficient. By tuning

its parameters, it can show strong ability in exploration or exploitation, and thus it is widely used in hybridization with other optimization algorithms [13–16]. Specifically, several attempts have been made to cooperate DE with FWA. In [17], DE operators are applied to the population that has gone through selection in FWA; in [18] and [19], DE operators are applied to the population after explosion in FWA; and in [20], DE operators are used to generate DE explosion sparks in FWA.

In LoTFWA [21], a competitive loser-out tournament strategy (LoT) is proposed as an alternative manner of information interaction.

As a new baseline of FWA, a number of improved variants have been proposed based on LoTFWA to enhance its performance. According to the focus of improvement, they can be categorized into two types. The first type mainly focuses on LoT strategy. An improved LoT strategy (ILoT) based on the idea of population migration and mutation in biogeography-based optimization is proposed in [22]. The expected fitness improvement of each firework is divided into four degrees to follow different mechanisms to change their search pattern. In Multi-Scale Collaborative Fireworks Algorithm (MSCFWA) [23], LoT is adjusted to coordinate independent local searches of non-optimal fireworks. Fireworks that have not improved for many iterations are adjusted with reference to the better firework. In Adaptive Niche Radius Fireworks Algorithm (ANRFWA) [24], the concept of domination space is proposed and used to extend the mechanism of LoT. The fireworks in another firework's domain space are directly eliminated to save computational resource. In Last-Position Elimination-Based Fireworks Algorithm (LEFWA) [25], an elimination strategy is proposed to cooperate with LoT. More fireworks are initialized at the beginning, and the worst one is eliminated for every fixed number of generations. The second type mainly focuses on explosion. In Neighborhood Information Utilization Fireworks Algorithm (NiFWA) [5], fireworks are divided into subpopulations, and the explosion operation is improved to give further utilization of the neighborhood subpopulations' information. In Exponentially Decaying Fireworks Algorithm (EDFWA) [26], a mechanism called exponentially decaying explosion, which can generate series of explosion sparks whose amplitudes and numbers decay in a exponential manner, was proposed to enhance the explosion's local search ability. In Enhanced Multi-Modal Loser-Out Tournament-Based Fireworks Algorithm (ELoTFWA) [27], a position-based mapping rule called PMR and a self-adaptive strategy for adjusting the number of explosion sparks are proposed. The proposed strategy can better balance the global and local search abilities of the algorithm. There are also a few works that improve both operations. In a hybrid algorithm of Covariance Matrix Adaptation Evolutionary Strategies and LoTFWA (CMAFWA) [28], explosion is replaced by an adaptation strategy for a Gaussian distribution, a collaboration method based on search space partition is proposed to arrange the search areas of fireworks, and LoT is also improved by introducing more restart conditions.

However, it is indicated [29] that there is still much room for improvement. For example, the communication among fireworks is not sufficient, and the reinitialization has too much cost. At the same time, the explosion and guiding cannot deal with a local optimum efficiently enough.

Thus, a triple-spark (TS) guiding strategy is proposed in this paper. Unlike many other variants of LoTFWA, the new guiding strategy focuses on improving the guiding stage of LoTFWA and makes the following two main contributions:

1) The new strategy significantly improves the baseline algorithm, providing a new perspective of enhancing the performance of FWA. In detail, the triple-spark guiding strategy contains three different

executing mechanisms to generate guiding sparks. The first guiding strategy is the same as that of LoTFWA. The barycenter guiding strategy uses the central position of several explosion sparks as guiding spark and aims at improving the ability to search in local areas of LoTFWA. The DE guiding strategy takes the advantage of DE mutation, and it aims at providing communications among fireworks at the guiding stage.

2) The new strategy does not conflict with most FWA variants, allowing it to easily cooperate with other strategies. Specifically, for variants containing guiding strategy, like LoTFWA and ILoTFWA, cooperation can be done by substituting its guiding strategy with triple-spark (TS) guiding strategy, as shown later in the article. For variants that do not generate guiding spark, like EDFWA, cooperation can be done by adding TS guiding strategy after the explosion of fireworks. Moreover, the computational cost of TS guiding strategy is low, so there is no need to worry about significant increase in running time of the hybrid algorithm.

The remainder of this paper is organized as follows. Section 2 introduces the background knowledge that provides inspirations for this paper. Section 3 presents the component guiding strategies and describes the whole strategy. Section 4 shows the experimental results to illustrate the performance of the proposed strategy. Section 5 concludes this paper.

2. Related works

In this section, the framework of LoTFWA, DE mutation, and the improved LoT mechanism of ILoTFWA are introduced in details.

2.1. LoTFWA

As a new baseline of FWA variants, LoTFWA [21] has shown strong competitiveness in the field of optimization and thus is chosen to be the foundation of this work.

Figure 1 provides an illustration of the framework of LoFWA by demonstrating its basic features. Local search is conducted by fireworks through explosion, guiding and selection. Global coordination is realized by exchanging information among fireworks with LoT and resource allocation. The pseudocode of LoTFWA is shown as Algorithm 1, and its operations are described in detail in the following sections.

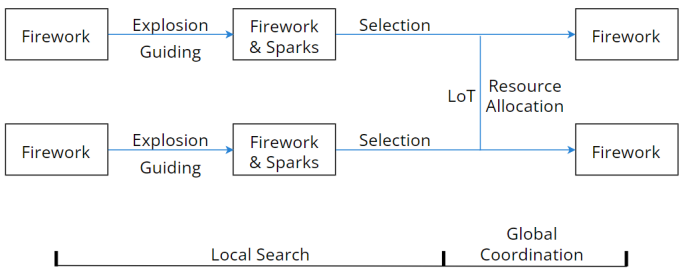

Figure 1. Basic features of LoTFWA.

Algorithm 1 LoTFWA

Randomly initialize μ fireworks in search space
Evaluate the fireworks' fitness
repeat
  for each firework **do**
    Generate and evaluate explosion sparks
    Generate and evaluate guiding sparks
    Select the best individual (including the firework, explosion sparks and guiding sparks) as the firework of next generation
  end for
  Perform LoT as described in section 2.1.4
until termination criterion is met
return the position and the fitness of the best individual

The fireworks are first randomly initialized in the search space, and then the explosion sparks are generated randomly in the explosion amplitudes of each firework. According to the explosion sparks, guiding sparks are generated for each firework. The relationship between firework and sparks is shown in Figure 2. After that, the next generation of fireworks are chosen using elitism selection. Finally, LoT is performed to provide cooperation among fireworks.

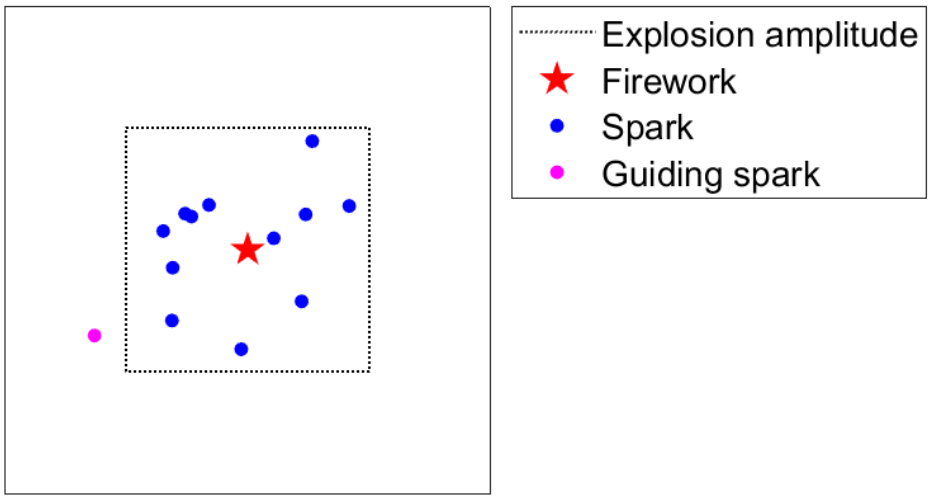

Figure 2. Illustration of fireworks explosions.

2.1.1. Explosion

The explosion sparks of each firework are random vectors uniformly distributed within the explosion amplitude of the corresponding firework. The generation process is as shown in Eq (2.1):

$$\boldsymbol{s}_{ij}^{g} = \boldsymbol{X}_{i}^{g} + \boldsymbol{\eta}_{j}^{g} \cdot A_{i}^{g} \tag{2.1}$$

where $\boldsymbol{s}_{ij}^{g}$ is the position of the jth spark generated by the ith firework in generation g, $\boldsymbol{X}_{i}^{g}$ is the position of the ith firework in generation g, $\boldsymbol{\eta}_{j}^{g}$ is a vector with each of its components independently sampled from a uniform distribution between -1 and 1, and A_{i}^{g} is the explosion amplitude of the ith firework in generation g.

The number of explosion sparks is allocated based on the performance of the firework:

$$\lambda_{r}^{g} = \begin{cases} \lambda_{r}^{1}, & g = 1 \\ \hat{\lambda} \cdot \frac{r^{-\alpha}}{\sum_{r=1}^{\mu} r^{-\alpha}}, & g \neq 1 \end{cases} \tag{2.2}$$

where r is the fitness ranking of the current firework, λ_{r}^{g} is the number of sparks that will be generated by the current firework in generation g, and $\hat{\lambda}$ is the total number of sparks that will be generated by all fireworks. μ is the total number of the fireworks, α is a parameter to control the shape of the distribution, and λ_{r}^{1} is a preset constant.

The size of explosion amplitude is controlled in a dynamic way:

$$A_{i}^{g} = \begin{cases} A_{i}^{1}, & g = 1 \\ C_{r} A_{i}^{g-1}, & f(\boldsymbol{X}_{i}^{g}) \geq f(\boldsymbol{X}_{i}^{g-1}) \\ C_{a} A_{i}^{g-1}, & f(\boldsymbol{X}_{i}^{g}) < f(\boldsymbol{X}_{i}^{g-1}) \end{cases} \tag{2.3}$$

where A_{i}^{g} is the explosion amplitude of the ith firework in generation g. A_{i}^{1}, C_{a}, C_{r} are preset constants so that $C_{a} > 1, C_{r} < 1$.

2.1.2. Guiding spark

The generation of the guiding spark in the LoTFWA uses the difference vector between the centroids of one group of explosion sparks with excellent fitness and one with inferior fitness. It is as indicated in Algorithm 2.

Algorithm 2 Generating the Guiding Spark for the ith Firework

Require: The ith firework's position $\boldsymbol{X}_{i}$, The ith firework's sparks' positions $\boldsymbol{s}_{ij}$, the ith firework's sparks' fitness $f(\boldsymbol{s}_{ij})$, the ith firework's fitness ranking r_{i}, the ith firework's spark number λ_{r_i} and a preset constant σ

Sort the sparks by their fitness values $f(\boldsymbol{s}_{ij})$ in ascending order

$\Delta_{i} \leftarrow \frac{1}{\sigma\lambda_{r_i}} (\sum_{j=1}^{\sigma\lambda_{r_i}} \boldsymbol{s}_{ij} - \sum_{j=\lambda_{r_i}-\sigma\lambda_{r_i}+1}^{\lambda_{r_i}} \boldsymbol{s}_{ij})$

$\boldsymbol{G}_{i} \leftarrow \boldsymbol{X}_{i} + \Delta_{i}$

return $\boldsymbol{G}_{i}$

2.1.3. Selection

Each firework in the next generation is selected from its own candidate pool using elitism selection mechanism:

$$\boldsymbol{X}_{i}^{g+1} = \arg\min\{f(\boldsymbol{X}_{i}^{g}), f(\boldsymbol{s}_{ij}), f(\boldsymbol{G}_{i})\}. \tag{2.4}$$

2.1.4. LoT

The LoT operation is designed for exchanging information among fireworks and giving those fireworks who get stuck in local optimum a chance to jump out.

Define the improvement of the ith firework $\boldsymbol{X}_i^g$ in generation g as

$$\delta_i^g = f(\boldsymbol{X}_i^{g-1}) - f(\boldsymbol{X}_i^g) \geq 0. \tag{2.5}$$

Then the prediction of its fitness in the final generation $g_{\max}$ can be made as

$$\widehat{f(\boldsymbol{X}_i^{g_{\max}})} = f(\boldsymbol{X}_i^g) - (g_{\max} - g)\delta_i^g \tag{2.6}$$

For each firework, if its prediction fitness is worse than the best firework in the current generation, the firework will be reinitialized.

2.1.5. Mapping rule

In optimization problems, the sparks might fly out of the search space in certain dimensions. If so, the decision variables in those dimensions will be uniformly randomly regenerated in the search space.

2.2. DE

In order to utilize the advantage of DE, multiple attempts have been made to combine DE with FWA. With fine-tuned parameters, DE operators can provide a new way of inter-firework communication and thus improve the performance of FWA in both exploration and exploitation. Consequently, DE operators become the possible choice of generating guiding sparks for LoTFWA.

The related DE mutation operator is described as Eq (2.7), where $\boldsymbol{v}_{i,g}$ is the ith mutated individual in generation g, $\boldsymbol{x}_{r0,g}, \boldsymbol{x}_{r1,g}, \boldsymbol{x}_{r2,g}$ are different individuals randomly chosen in generation g, F_i is a preset constant parameter, and the hybrid guiding strategy is presented in Section 3.

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{r0,g} + F_i \cdot (\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}) \tag{2.7}$$

2.3. ILoTFWA

In [22], a new loser-out tournament (LoT) mechanism is introduced and enhances the performance of LoTFWA greatly. Its procedure is presented in Algorithm 3.

After making prediction as Eq (2.6), the expected fitness improvement of each firework is divided into four degrees to follow different mechanisms to change their search pattern:

1) retains to ensure stability

2) migrates to the best fireworks population and update explosion amplitude to enhance exploitation ability

3) mutates in a certain range to enhance exploration ability

4) reinitialized

Algorithm 3 ILoT

Require: the fireworks' positions $\boldsymbol{X}_i^g$, a preset constant θ, the upper bound of the search space L and the lower bound of the search space U

for each firework $\boldsymbol{X}_i^g$ **do**

$\Delta_i = f(\boldsymbol{X}_i^g) - \min_j\{f(\boldsymbol{X}_j^g)\}$

$\widehat{f(\boldsymbol{X}_i^{g_{\max}})} \leftarrow$ make prediction according to Eq (2.6)

if $\widehat{f(\boldsymbol{X}_i^{g_{\max}})} > \min_j\{f(\boldsymbol{X}_j^g)\} + \frac{3\Delta_i}{4}$ **then**

reinitialize $\boldsymbol{X}_i^g$ as LoT

else if $\widehat{f(\boldsymbol{X}_i^{g_{\max}})} > \min_j\{f(\boldsymbol{X}_j^g)\} + \frac{\Delta_i}{2}$ **then**

$\boldsymbol{X}_i^g = \frac{rand(-1,1)\cdot(L-U)}{10} + \boldsymbol{X}_i^g$

else if $\widehat{f(\boldsymbol{X}_i^{g_{\max}})} > \min_j\{f(\boldsymbol{X}_j^g)\}$ **then**

$\boldsymbol{X}_i^g = \boldsymbol{X}_i^g + rand(0,1)\cdot(\boldsymbol{X}_{best}^g - \boldsymbol{X}_i^g)$ $\{\boldsymbol{X}_{best}^g$ is the best firework in generation $g\}$

$A_i = (1-\theta)A_i + \theta\|\boldsymbol{X}_{best}^g - \boldsymbol{X}_i^g\|$

end if

end for

The proposed guiding strategy can also cooperate with ILoTFWA and enhance its performance as shown later in experiments.

3. Proposed guiding strategy

There are two main problems with the original guiding spark strategy.

- The guiding spark's tendency to explore or exploit is unstable.
- The utilized information is limited in one single firework and its sparks.

In this section, we will try to settle these problems by introducing three strategies of generating guiding sparks for the fireworks and merge them together to propose TS guiding strategy.

3.1. Original guiding strategy

Li et al. [30] indicated that the original guiding spark of LoTFWA contributes greatly to both exploration and exploitation. It can also be observed from the later experiments that the original guiding spark can improve the algorithm's performance no matter what other guiding sparks it cooperates with.

Therefore, it is decided to retain the original guiding spark strategy as a part of the proposed strategy.

3.2. Barycenter guiding strategy

The guiding spark generated by the original guiding strategy is not stable enough because it may jump out of the explosion amplitude and lose the ability of exploiting the local region. Thus, we propose the barycenter guiding strategy to solve the problem.

As shown in Algorithm 4, the barycenter guiding strategy generates guiding spark with the central position of several best explosion sparks. It can be seen later in experiments that this guiding strategy

itself performs comparably with Algorithm 2 despite scarce exploration due to its strong exploitation ability.

Algorithm 4 Barycenter Guiding Strategy

Require: The ith firework's position $\boldsymbol{X}_i$, The ith firework's sparks' positions $\boldsymbol{s}_{ij}$, the ith firework's sparks' fitness $f(\boldsymbol{s}_{ij})$, the ith firework's fitness ranking r_i, the ith firework's spark number λ_{r_i} and a preset constant σ

Sort the sparks by their fitness values $f(\boldsymbol{s}_{ij})$ in ascending order

$\boldsymbol{G}_{i,2} \leftarrow \frac{1}{\sigma\lambda_{r_i}} \sum_{j=1}^{\sigma\lambda_{r_i}} \boldsymbol{s}_{ij}$

return $\boldsymbol{G}_{i,2}$

3.3. DE guiding strategy

DE guiding strategy generates guiding spark with DE operators as shown in Algorithm 5. It presents a way of communicating among fireworks at the guiding stage, which can enhance exploration ability of the algorithm.

In this strategy, DE mutation operator is applied to guiding sparks generated by original guiding strategy and barycenter guiding strategy. There is a little difference between the operator we use here and the one shown in Eq (2.7): We fix the first item to be the guiding spark of the current firework, so the new spark will not be too far away from the firework and its population. The other two items are randomly chosen and different from each other, but they are not necessarily different from $\boldsymbol{G}_{i,1}$. This is due to the fact that the number of fireworks in LoTFWA is not large. Such design helps increase diversity of information interaction among fireworks.

In the initial stage of our trial, we found that choosing crossover probability to be 1 performs no worse than other choices. At the same time, the greater the crossover rate is, the better it can explore, which matches our designing purpose of this strategy. So, the DE crossover operator is not applied in this work.

Though the same job of information exchanging and exploration has been done by LoT in LoTFWA [21], the expense of DE guiding strategy is much less than LoT. Random reinitialization costs much more than one evaluation only for it totally abandons the current information carried by the current firework.

Additionally, although the designing purpose of DE guiding strategy is to enhance exploration, it can actually demonstrate different abilities according to the stage of the algorithm and the properties of the objective function. In the initial stage of the algorithm, fireworks are spread randomly in the search space, and DE operator plays a role of exploration. In the convergence stage, the operator will show its ability of exploitation if the objective function is unimodal, since the fireworks are moving toward the same target, and their difference will be lessened. If the objective function is multimodal, fireworks are easily stuck in different local optimum and are still away from each other. Consequently, DE operator provides another exploring mechanism to help them get out of local optimum.

Algorithm 5 DE Guiding Strategy

Require: all fireworks' first guiding sparks' positions $\boldsymbol{G}_{j,1}$ and the current firework's second guiding spark's position $\boldsymbol{G}_{i,2}$

$\boldsymbol{G}_{i,3} \leftarrow \boldsymbol{G}_{i,2} + F \cdot (\boldsymbol{G}_{r1,1} - \boldsymbol{G}_{r2,1})$ {apply DE mutation identical to Eq (2.7)}

return $\boldsymbol{G}_{i,3}$

3.4. Triple-spark loser-out tournament-based fireworks algorithm

Combining the three guiding strategies mentioned above will result in the TS guiding strategy. The three strategies will be conducted sequentially, generating three different guiding sparks. The best one among them will be chosen to serve as the role of the original guiding spark in LoTFWA.

Replacing the original guiding strategy in LoTFWA with TS guiding strategy will yield TSLoTFWA. Algorithm 6 introduces the complete pseudocode for TSLoTFWA. Similar to LoTFWA, after initialization, explosion sparks will be generated for each firework, along with the conduction of original guiding strategy and barycenter guiding strategy. After the generation of original guiding sparks for all fireworks, DE guiding strategy will be conducted. The new generation of firework will be selected among its population. Finally, LoT will be performed. Note that the DE guiding strategy requires the original guiding sparks of all fireworks, so it should be conducted after all original guiding sparks are generated.

Algorithm 6 TSLoTFWA

```
Randomly initialize μ fireworks in search space
Evaluate the fireworks' fitness
repeat
  for each firework X_i do
    Generate and evaluate explosion sparks
    G_{i,2} ← generate and evaluate a GS for X_i according to Algorithm 4
    G_{i,1} ← generate and evaluate a GS for X_i according to Algorithm 2
  end for
  for each firework X_i do
    G_{i,3} ← generate and evaluate a GS for X_i according to Algorithm 5
    Select the best individual (including the firework, explosion sparks and guiding sparks) as the
    firework of next generation
  end for
  Perform LoT
until termination criterion is met
return the position and the fitness of the best individual
```

To validate that TS guiding strategy can bring promotion to other state-of-the-art FWA variants, we also apply TS guiding strategy to ILoTFWA, resulting in TSILoTFWA, which will be examined in detail in Section 4.5.

4. Experiments and analysis

4.1. Evaluate methodology

In this section, strategy validation tests and algorithm comparison tests are performed to verify the effectiveness and efficiency of the proposed strategy.

The following tests are all conducted on 28 benchmark functions from the CEC' 13 competition [31], which are shown in detail in Table 1.

Table 1. Test functions of CEC 2013 single objective optimization benchmark suite.

No.	Functions	$f_i^* = f_i(x^*)$
1	Sphere Function	−1400
2	Rotated High Conditioned Elliptic Function	−1300
3	Rotated Bent Cigar Function	−1200
4	Rotated Discus Function	−1100
5	Different Powers Function	−1000
6	Rotated Rosenbrock's Function	−900
7	Rotated Schaffers F7 Function	−800
8	Rotated Ackley's Function	−700
9	Rotated Weierstrass Function	−600
10	Rotated Griewank's Function	−500
11	Rastrigin's Function	−400
12	Rotated Rastrigin's Function	−300
13	Non-Continuous Rotated Rastrigin's Function	−200
14	Schwefel's Function	−100
15	Rotated Schwefel's Function	100
16	Rotated Katsuura Function	200
17	Lunacek Bi_Rastrigin Function	300
18	Rotated Lunacek Bi_Rastrigin Function	400
19	Expanded Griewank's plus Rosenbrock's Function	500
20	Expanded Schaffer's F6 Function	600
21	Composition Function 1 (n=5,Rotated)	700
22	Composition Function 2 (n=3,Unrotated)	800
23	Composition Function 3 (n=3,Rotated)	900
24	Composition Function 4 (n=3,Rotated)	1000
25	Composition Function 5 (n=3,Rotated)	1100
26	Composition Function 6 (n=5,Rotated)	1200
27	Composition Function 7 (n=5,Rotated)	1300
28	Composition Function 8 (n=5,Rotated)	1400
	Search Range: $[-100,100]^D$	

The benchmark dimension is D=30, and the maximal number of function evaluations is 10000*D. For each algorithm and test function, the test runs for 51 times, and the mean errors of optimal solutions are recorded. To evaluate the performance of the proposed algorithm, the mean errors are ranked on each function, and the averaged rankings (AR) over all functions for each algorithm are calculated. Additionally, two-sided Wilcoxon rank sum tests [32] (with confidence level 95%) are conducted to validate the performance improvement of the proposed strategy.

4.2. Parameter setting

Parameters inheriting from LoTFWA are shown in Table 2 and set according to the suggestion in [21].

In order to study the impact of mutation rate F in Algorithm 5 and verify if the proposed strategy is sensitive to the choice of the parameter, five different choices for F are conducted as the group of $F \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. The simulations are conducted on all 28 benchmark functions, and the results are shown in Table 3. Best results are marked in bold.

Table 2. Parameter settings in LoTFWA.

Parameter	Description	Value
μ	Number of fireworks	5
$\hat{\lambda}$	Total number of explosion sparks in one generation	300
C_a	Amplification coefficient that controls the amplitude	1.2
C_r	Reduction coefficient that controls the amplitude	0.9
σ	Parameter that controls the proportion of adopted explosion sparks	0.2
α	Parameter that controls the shape of the distribution of explosion sparks	0

Table 3 indicates that assigning F too close to 0 or 1 will lead to relatively poor performances. With F close to 0, the guiding spark generated by DE guiding strategy will have no significant difference compared with the guiding spark generated by barycenter guiding strategy, which will waste computation resource. With F close to 1, the guiding spark generated by DE guiding strategy will be too far from the current firework, leading to inadequate utilization of the known information. The choices of $F = 0.3, F = 0.5, F = 0.7$ do not suffer from these problems and thus have similar performances, indicating that the algorithm is relatively insensitive to $F \in (0.3, 0.7)$. As Table 3 shows, $F = 0.3$ has a small advantage over the other two choices, so it will be used in the following experiments.

Table 3. Comparison among 5 sets of parameters.

value	F=0.1	F=0.3	F=0.5	F=0.7	F=0.9
AR.	3.14	**2.54**	2.64	2.79	3.46

4.3. Strategy validation

In order to verify that each algorithmic component of the TS guiding strategy is absolutely necessary, experiments are conducted on all combinations of the components based on LoTFWA.

The results are shown in Table 4, with $\boldsymbol{G}_1$ representing generating guiding spark according to Algorithm 2, $\boldsymbol{G}_2$ representing generating guiding spark according to Algorithm 4, $\boldsymbol{G}_3$ representing generating guiding spark according to Algorithm 5.

Table 4. Comparison among proposed strategy and other simplified versions.

func	$\boldsymbol{G}_1$	$\boldsymbol{G}_2$	$\boldsymbol{G}_3$	$\boldsymbol{G}_1\&\boldsymbol{G}_2$	$\boldsymbol{G}_1\&\boldsymbol{G}_3$	$\boldsymbol{G}_2\&\boldsymbol{G}_3$	$\boldsymbol{G}_1\&\boldsymbol{G}_2\&\boldsymbol{G}_3$
1	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$	$\mathbf{0.00\times10^{0}}$
2	1.21×10^{6}	1.50×10^{6}	$\mathbf{8.81\times10^{5}}$	1.51×10^{6}	9.91×10^{5}	1.06×10^{6}	1.02×10^{6}
3	2.39×10^{7}	8.80×10^{6}	1.30×10^{7}	8.58×10^{6}	4.13×10^{6}	1.88×10^{6}	$\mathbf{9.35\times10^{5}}$
4	1.93×10^{3}	9.43×10^{2}	1.82×10^{3}	8.08×10^{2}	1.98×10^{3}	$\mathbf{3.81\times10^{2}}$	4.47×10^{2}
5	3.58×10^{-3}	5.74×10^{-3}	2.09×10^{-3}	5.31×10^{-3}	$\mathbf{1.80\times10^{-3}}$	2.52×10^{-3}	2.41×10^{-3}
6	$\mathbf{1.31\times10^{1}}$	1.59×10^{1}	1.45×10^{1}	1.46×10^{1}	1.32×10^{1}	1.51×10^{1}	1.48×10^{1}
7	5.02×10^{1}	4.22×10^{1}	4.21×10^{1}	4.16×10^{1}	4.15×10^{1}	3.20×10^{1}	$\mathbf{3.02\times10^{1}}$
8	2.09×10^{1}	2.09×10^{1}	2.08×10^{1}	2.08×10^{1}	2.09×10^{1}	2.09×10^{1}	$\mathbf{2.08\times10^{1}}$
9	1.45×10^{1}	1.30×10^{1}	1.74×10^{1}	1.29×10^{1}	1.43×10^{1}	1.33×10^{1}	$\mathbf{1.22\times10^{1}}$
10	4.04×10^{-2}	2.95×10^{-2}	5.81×10^{-2}	3.41×10^{-2}	2.98×10^{-2}	2.57×10^{-2}	$\mathbf{2.54\times10^{-2}}$
11	6.40×10^{1}	4.33×10^{1}	5.41×10^{1}	3.63×10^{1}	3.86×10^{1}	3.71×10^{1}	$\mathbf{3.20\times10^{1}}$
12	6.96×10^{1}	4.31×10^{1}	5.70×10^{1}	3.40×10^{1}	4.20×10^{1}	3.71×10^{1}	$\mathbf{2.75\times10^{1}}$
13	1.31×10^{2}	8.88×10^{1}	1.13×10^{2}	7.72×10^{1}	9.59×10^{1}	7.84×10^{1}	$\mathbf{7.10\times10^{1}}$
14	2.42×10^{3}	2.45×10^{3}	2.79×10^{3}	2.47×10^{3}	2.46×10^{3}	2.52×10^{3}	$\mathbf{2.31\times10^{3}}$
15	2.56×10^{3}	2.58×10^{3}	2.64×10^{3}	2.54×10^{3}	2.55×10^{3}	2.53×10^{3}	$\mathbf{2.49\times10^{3}}$
16	5.74×10^{2}	6.81×10^{2}	1.59×10^{1}	5.64×10^{2}	5.85×10^{2}	7.46×10^{2}	$\mathbf{5.57\times10^{2}}$
17	6.31×10^{1}	6.77×10^{1}	7.56×10^{1}	5.91×10^{1}	6.10×10^{1}	6.27×10^{1}	$\mathbf{5.63\times10^{1}}$
18	6.33×10^{1}	6.65×10^{1}	7.60×10^{1}	6.25×10^{1}	6.08×10^{1}	6.41×10^{1}	$\mathbf{5.79\times10^{1}}$
19	3.17×10^{0}	2.85×10^{0}	3.64×10^{0}	2.55×10^{0}	2.93×10^{0}	2.73×10^{0}	$\mathbf{2.50\times10^{0}}$
20	1.34×10^{1}	1.32×10^{1}	1.34×10^{1}	1.33×10^{1}	1.35×10^{1}	1.33×10^{1}	$\mathbf{1.31\times10^{1}}$
21	$\mathbf{2.00\times10^{2}}$	2.28×10^{2}	2.69×10^{2}	2.30×10^{2}	2.69×10^{2}	2.69×10^{2}	2.65×10^{2}
22	2.84×10^{3}	2.96×10^{3}	3.28×10^{3}	$\mathbf{2.84\times10^{3}}$	2.96×10^{3}	3.02×10^{3}	2.93×10^{3}
23	3.11×10^{3}	3.18×10^{3}	3.33×10^{3}	3.03×10^{3}	3.12×10^{3}	$\mathbf{3.03\times10^{3}}$	3.14×10^{3}
24	2.40×10^{2}	2.10×10^{2}	2.32×10^{2}	2.07×10^{2}	2.12×10^{2}	2.03×10^{2}	$\mathbf{2.01\times10^{2}}$
25	2.76×10^{2}	2.57×10^{2}	2.75×10^{2}	2.52×10^{2}	2.69×10^{2}	$\mathbf{2.41\times10^{2}}$	2.51×10^{2}
26	2.00×10^{2}	2.00×10^{2}	2.00×10^{2}	2.00×10^{2}	2.00×10^{2}	$\mathbf{2.00\times10^{2}}$	2.00×10^{2}
27	6.96×10^{2}	4.23×10^{2}	6.57×10^{2}	3.65×10^{2}	4.36×10^{2}	3.29×10^{2}	$\mathbf{3.19\times10^{2}}$
28	$\mathbf{2.69\times10^{2}}$	2.96×10^{2}	2.92×10^{2}	2.92×10^{2}	2.96×10^{2}	2.96×10^{2}	2.96×10^{2}
AR.	4.86	4.68	5.39	3.04	4.14	3.21	**1.82**

As the results indicate, the more strategies we use, the better result can be achieved. All forms of cooperation among the three strategies outperform the corresponding non-cooperative ones. Compared with algorithms without barycenter guiding strategy, the algorithms that take use of barycenter guiding strategy always perform better. This means that both the original guiding strategy and DE guiding strategy are good at exploration while relatively poor in exploitation, and the barycenter guiding strategy can remarkably make up for this shortcoming. Though DE guiding strategy itself performs worse than the original guiding strategy, the performance of the combination of all three strategies is significantly better than the cooperation of original guiding strategy and barycenter guiding strategy. This indicates that DE guiding strategy and original guiding strategy can complement each other in the ability of exploration.

In conclusion, the combination of all three strategies outperforms other combinations greatly. Thus, the three components of TS guiding strategy are all indispensable.

4.4. Performance comparison

In this section, the significance of TS guiding algorithm is presented by combining it with LoTFWA to form TSLoTFWA. Comparison is conducted between TSLoTFWA, LoTFWA, ILoTFWA and DE algorithm.

The results are shown in Table 5. In the table, the $+, -, \approx$ indicate whether a given algorithm performs significantly better (+), significantly worse (−), or not significantly different (≈) compared to TSLoTFWA according to Wilcoxon rank-sum test.

As can be seen from the result, TSLoTFWA performs significantly better than LoTFWA and DE on most benchmark functions. Over the total 28 benchmark functions, TSLoTFWA is significantly better than DE on 22 of them and is significantly better than LoTFWA on 17 of them. At the same time, the number of functions TSLoTFWA performs significantly worse is small, which indicates that the improvement from the TS guiding strategy has little side effect.

Furthermore, though the average ranking of TSLoTFWA is a little worse than ILoTFWA, it still holds the advantage from the Wilcoxon rank-sum test's point of view. On the whole, the result of TSLoTFWA is comparable to ILoTFWA.

In short, TS guiding strategy helps LoTFWA achieve better results with low cost, indicating that our design enhances the ability of both exploration and exploitation. The improvement it provides is similar to ILoT.

Table 5. Comparison among the proposed algorithm and other optimization algorithms.

func	TSLoTFWA	DE		LoTFWA		ILoTFWA	
1	$\mathbf{0.00 \times 10^{0}}$	$\mathbf{0.00 \times 10^{0}}$	≈	$\mathbf{0.00 \times 10^{0}}$	≈	$\mathbf{0.00 \times 10^{0}}$	≈
2	1.02×10^{6}	1.02×10^{8}	−	1.21×10^{6}	−	$\mathbf{7.64 \times 10^{5}}$	+
3	9.35×10^{5}	$\mathbf{5.37 \times 10^{5}}$	+	2.39×10^{7}	−	4.49×10^{6}	−
4	$\mathbf{4.47 \times 10^{2}}$	5.49×10^{4}	−	1.93×10^{3}	−	4.71×10^{2}	≈
5	2.41×10^{-3}	$\mathbf{0.00 \times 10^{0}}$	+	3.58×10^{-3}	−	2.39×10^{-3}	≈
6	1.48×10^{1}	1.62×10^{1}	−	$\mathbf{1.31 \times 10^{1}}$	≈	1.54×10^{1}	≈
7	3.02×10^{1}	$\mathbf{2.29 \times 10^{1}}$	+	5.02×10^{1}	−	3.47×10^{1}	≈
8	2.08×10^{1}	2.09×10^{1}	−	2.09×10^{1}	≈	$\mathbf{2.08 \times 10^{1}}$	≈
9	1.22×10^{1}	3.93×10^{1}	−	1.45×10^{1}	−	$\mathbf{1.21 \times 10^{1}}$	≈
10	$\mathbf{2.54 \times 10^{-2}}$	7.70×10^{0}	−	4.04×10^{-2}	−	3.67×10^{-2}	−
11	$\mathbf{3.20 \times 10^{1}}$	1.10×10^{2}	−	6.40×10^{1}	−	3.89×10^{1}	−
12	$\mathbf{2.75 \times 10^{1}}$	1.96×10^{2}	−	6.96×10^{1}	−	3.51×10^{1}	−
13	$\mathbf{7.10 \times 10^{1}}$	1.93×10^{2}	−	1.31×10^{2}	−	8.32×10^{1}	−
14	2.31×10^{3}	4.29×10^{3}	−	2.42×10^{3}	≈	$\mathbf{2.29 \times 10^{3}}$	≈
15	2.49×10^{3}	7.34×10^{3}	−	2.56×10^{3}	≈	$\mathbf{2.38 \times 10^{3}}$	≈
16	5.57×10^{-2}	2.41×10^{0}	−	5.74×10^{-2}	≈	$\mathbf{5.51 \times 10^{-2}}$	≈
17	5.63×10^{1}	1.41×10^{2}	−	6.31×10^{1}	−	$\mathbf{5.61 \times 10^{1}}$	≈
18	$\mathbf{5.79 \times 10^{1}}$	2.23×10^{2}	−	6.33×10^{1}	−	6.10×10^{1}	≈
19	$\mathbf{2.50 \times 10^{0}}$	1.37×10^{1}	−	3.17×10^{0}	−	2.70×10^{0}	≈
20	1.31×10^{1}	1.28×10^{1}	≈	1.34×10^{1}	≈	$\mathbf{1.25 \times 10^{1}}$	+
21	2.65×10^{2}	2.94×10^{2}	−	$\mathbf{2.00 \times 10^{2}}$	+	2.04×10^{2}	+
22	2.93×10^{3}	5.26×10^{3}	−	2.84×10^{3}	≈	$\mathbf{2.77 \times 10^{3}}$	+
23	3.14×10^{3}	7.88×10^{3}	−	3.11×10^{3}	≈	$\mathbf{2.78 \times 10^{3}}$	+
24	$\mathbf{2.01 \times 10^{2}}$	2.16×10^{2}	−	2.40×10^{2}	−	2.13×10^{2}	−
25	$\mathbf{2.51 \times 10^{2}}$	3.00×10^{2}	−	2.76×10^{2}	−	2.65×10^{2}	−
26	$\mathbf{2.00 \times 10^{2}}$	2.08×10^{2}	−	2.00×10^{2}	−	2.00×10^{2}	−
27	$\mathbf{3.19 \times 10^{2}}$	1.10×10^{3}	−	6.96×10^{2}	−	5.74×10^{2}	−
28	2.96×10^{2}	3.00×10^{2}	≈	$\mathbf{2.69 \times 10^{2}}$	+	2.76×10^{2}	≈
AR.	1.79	3.46		2.82		**1.71**	
	+	3		2		5	
	−	22		17		9	
	≈	3		9		14	

4.5. Combining with ILoTFWA

As mentioned above, the overall improvements provided by the TS guiding strategy have no obvious advantage versus ILoT. However, the main focuses of TS guiding strategy and ILoTFWA are different from each other. At the same time, ILoTFWA does not change other parts of the algorithm significantly, which makes it possible for them to combine and achieve better results.

The combination of ILoTFWA and TS guiding strategy is very simple: substituting the original

strategy used for generating guiding sparks in ILoTFWA with TS guiding strategy. Related experiment results are displayed in Table 6. Notations in this table have similar meanings as Table 5.

Table 6. Comparison among TSILoTFWA and other LoTFWA.

func	TSILoTFWA	LoTFWA		TSLoTFWA		ILoTFWA	
1	$\mathbf{0.00 \times 10^{0}}$	$\mathbf{0.00 \times 10^{0}}$	≈	$\mathbf{0.00 \times 10^{0}}$	≈	$\mathbf{0.00 \times 10^{0}}$	≈
2	8.91×10^{5}	1.21×10^{6}	−	1.02×10^{6}	−	$\mathbf{7.64 \times 10^{5}}$	+
3	3.22×10^{6}	2.39×10^{7}	−	$\mathbf{9.35 \times 10^{5}}$	+	4.49×10^{6}	−
4	$\mathbf{1.11 \times 10^{2}}$	1.93×10^{3}	−	4.47×10^{2}	−	4.71×10^{2}	−
5	$\mathbf{1.99 \times 10^{-3}}$	3.58×10^{-3}	−	2.41×10^{-3}	−	2.39×10^{-3}	−
6	1.45×10^{1}	$\mathbf{1.31 \times 10^{1}}$	≈	1.48×10^{1}	−	1.54×10^{1}	≈
7	$\mathbf{2.74 \times 10^{1}}$	5.02×10^{1}	−	3.02×10^{1}	≈	3.47×10^{1}	−
8	2.08×10^{1}	2.09×10^{1}	≈	2.08×10^{1}	≈	$\mathbf{2.08 \times 10^{1}}$	≈
9	$\mathbf{1.03 \times 10^{1}}$	1.45×10^{1}	−	1.22×10^{1}	−	1.21×10^{1}	−
10	$\mathbf{2.26 \times 10^{-2}}$	4.04×10^{-2}	−	2.54×10^{-2}	≈	3.67×10^{-2}	−
11	$\mathbf{3.09 \times 10^{1}}$	6.40×10^{1}	−	3.20×10^{1}	≈	3.89×10^{1}	−
12	2.80×10^{1}	6.96×10^{1}	−	$\mathbf{2.75 \times 10^{1}}$	≈	3.51×10^{1}	−
13	$\mathbf{6.23 \times 10^{1}}$	1.31×10^{2}	−	7.10×10^{1}	−	8.32×10^{1}	−
14	2.38×10^{3}	2.42×10^{3}	≈	2.31×10^{3}	≈	$\mathbf{2.29 \times 10^{3}}$	≈
15	$\mathbf{2.30 \times 10^{3}}$	2.56×10^{3}	−	2.49×10^{3}	−	2.38×10^{3}	≈
16	$\mathbf{5.34 \times 10^{-2}}$	5.74×10^{-2}	≈	5.57×10^{-2}	≈	5.51×10^{-2}	≈
17	$\mathbf{5.34 \times 10^{1}}$	6.31×10^{1}	−	5.63×10^{1}	−	5.61×10^{1}	−
18	$\mathbf{5.36 \times 10^{1}}$	6.33×10^{1}	−	5.79×10^{1}	−	6.10×10^{1}	−
19	$\mathbf{2.46 \times 10^{0}}$	3.17×10^{0}	−	2.50×10^{0}	≈	2.70×10^{0}	≈
20	$\mathbf{1.23 \times 10^{1}}$	1.34×10^{1}	−	1.31×10^{1}	−	1.25×10^{1}	≈
21	2.73×10^{2}	$\mathbf{2.00 \times 10^{2}}$	+	2.65×10^{2}	≈	2.04×10^{2}	+
22	$\mathbf{2.72 \times 10^{3}}$	2.84×10^{3}	≈	2.93×10^{3}	−	2.77×10^{3}	≈
23	2.89×10^{3}	3.11×10^{3}	−	3.14×10^{3}	−	$\mathbf{2.78 \times 10^{3}}$	≈
24	$\mathbf{2.01 \times 10^{2}}$	2.40×10^{2}	−	2.01×10^{2}	−	2.13×10^{2}	−
25	2.59×10^{2}	2.76×10^{2}	−	$\mathbf{2.51 \times 10^{2}}$	≈	2.65×10^{2}	−
26	2.00×10^{2}	2.00×10^{2}	−	$\mathbf{2.00 \times 10^{2}}$	≈	2.00×10^{2}	−
27	3.31×10^{2}	6.96×10^{2}	−	$\mathbf{3.19 \times 10^{2}}$	+	5.74×10^{2}	−
28	2.96×10^{2}	$\mathbf{2.69 \times 10^{2}}$	+	2.96×10^{2}	≈	2.76×10^{2}	≈
AR.	**1.57**	3.50		2.32		2.36	
	+	2		2		2	
	−	20		13		15	
	≈	6		13		11	

By combining with ILoTFWA, TSILoTFWA shows significantly better results compared to both TSLoTFWA and ILoTFWA. At the same time, the difference between the performance of TSILoTFWA and ILoTFWA is quite similar to the difference between the performance of TSLoTFWA and LoTFWA. This means that TS guiding strategy can have similar effects on LoTFWA and ILoTFWA and provide robust performance improvements for FWA variants.

According to the average ranking values and the rank-sum test results, the newly proposed TSILoTFWA achieves very competitive performance when comparing with the most well-known competitors. It takes the performance of LoTFWA to a brand-new level.

4.6. Computational cost

With promising performance, the computational cost of TS guiding strategy is low according to the experimental results shown in Table 7. For each algorithm, the average running time on CEC' 13 benchmark functions with maximal function evaluation time equal to 300,000 is recorded as $T2$. $T1$ stands for the average running time of 300,000 evaluations of the benchmark functions without any other operations.

Table 7. Computational cost of TS guiding strategy and other algorithms.

Algorithm	$T1$(s)	$T2$(s)	$(T2-T1)/T1$
DE		6.958	0.762
$\boldsymbol{G}_1$(LoTFWA)		4.437	0.124
$\boldsymbol{G}_2$		4.420	0.119
$\boldsymbol{G}_3$	3.949	4.503	0.140
TSLoTFWA		4.504	0.140
ILoTFWA		4.384	0.110
TSILoTFWA		4.444	0.125

Among the three component strategies, barycenter guiding strategy has the lowest cost, while DE guiding strategy has the greatest cost. This is because the computation required by barycenter guiding strategy is included in the original guiding strategy, and the DE guiding strategy requires the generation of the original guiding spark and the barycenter guiding spark. At the same time, as shown in the table, the computation complexity of DE is high due to its structure, which is another reason for the relatively high cost of DE guiding strategy.

As for the complete TS guiding strategy, its computational cost is not simple summation of the three component strategies. This is because, as mentioned above, certain computation steps are exactly the same in different strategies and thus can be simplified. Specifically, all extra computations before function evaluation in TS guiding strategy are included in DE guiding strategy. As a result, the computational cost of TSLoTFWA is similar to LoTFWA with DE guiding strategy.

As the result indicates, when cooperating with LoTFWA, the TS guiding strategy results in an increase of about 12.9% in the computational cost, 13.6% when cooperating with ILoTFWA, which is a reasonable trade-off between cost and performance. As for absolute running time, it takes about 0.07 seconds to perform TS guiding strategy when solving CEC' 13 benchmark functions, which is less than 2% of $T1$. Therefore, the computational cost of TS guiding strategy is low.

5. Conclusions

This paper proposes a new guiding strategy for LoTFWA which contains three different kinds of guiding spark generation methods. Experimental results on the CEC' 13 benchmark suite indicate

that the proposed strategy has low computational cost and can enhance both LoTFWA and ILoTFWA. Therefore, it is believed that the proposed strategy is efficient and effective for optimization. Future works can be done to integrate the TS guiding strategy with other variants of FWA to attain possible improvements.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61973042), Beijing Natural Science Foundation (1202020) and BUPT Excellent Ph.D. Students Foundation (CX2022157).

Conflict of interest

The authors declare no conflict of interest.

References

1. Y. Tan, Y. Zhu, Fireworks algorithm for optimization, in *Advances in Swarm Intelligence*, (2010), 355–364. https://doi.org/10.1007/978-3-642-13495-1_44
2. W. Luo, H. Jin, H. Li, X. Fang, R. Zhou, Optimal performance and application for firework algorithm using a novel chaotic approach, *IEEE Access*, **8** (2020), 120798–120817. https://doi.org/10.1109/ACCESS.2020.3004430
3. S. An, S. Xiao, G. Zou, D. Lin, Gradient-based fireworks algorithm for solving high dimensional electromagnetic optimization design, in *2022 IEEE 20th Biennial Conference on Electromagnetic Field Computation (CEFC)*, (2022), 1–2. https://doi.org/10.1109/CEFC55061.2022.9940703
4. X. Li, T. Zhang, X. Zhao, S. Li, Region selection with discrete fireworks algorithm for person re-identification, in *Advances in Swarm Intelligence*, (2021), 433–440. https://doi.org/10.1007/978-3-030-78743-1_39
5. X. Liu, X. Qin, A neighborhood information utilization fireworks algorithm and its application to traffic flow prediction, *Expert Syst. Appl.*, **183** (2021), 115189, https://doi.org/10.1016/j.eswa.2021.115189
6. Y. Hu, M. Li, X. Liu, Y. Tan, Multi-source, multi-object and multi-domain (m-sod) electromagnetic interference system optimised by intelligent optimisation approaches, *Nat. Comput.*, **19** (2020), 713–732. https://doi.org/10.1007/s11047-019-09728-8
7. D. Roch Dupré, T. Gonsalves, *Increasing Energy Efficiency by Optimizing the Electrical Infrastructure of a Railway Line Using Fireworks Algorithm*, IGI Global, Pennsylvania, 2020. https://doi.org/10.4018/978-1-7998-1659-1.ch012
8. X. Zhou, Q. Zhao, D. Zhang, Discrete fireworks algorithm for welding robot path planning, *J. Phys.: Conf. Ser.*, **1267** (2019), 012003. https://doi.org/10.1088/1742-6596/1267/1/012003
9. A. M. Yadav, K. N. Tripathi, S. C. Sharma, An enhanced multi-objective fireworks algorithm for task scheduling in fog computing environment, *Cluster Comput.*, **25** (2022), 983–998. https://doi.org/10.1007/s10586-021-03481-3

10. Y. Chen, F. He, X. Zeng, H. Li, Y. Liang, The explosion operation of fireworks algorithm boosts the coral reef optimization for multimodal medical image registration, *Eng. Appl. Artif. Intell.*, **102** (2021), 104252, https://doi.org/10.1016/j.engappai.2021.104252
11. M. Li, Y. Tan, Economic dispatch optimization for microgrid based on fireworks algorithm with momentum, in *Advances in Swarm Intelligence* (2022), 339–353. https://doi.org/10.1007/978-3-031-09677-8_29
12. R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, **11** (1997), 341–359. https://doi.org/10.1023/A:1008202821328
13. X. Yu, N. Jiang, X. Wang, M. Li, A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning, *Expert Syst. Appl.*, **215** (2022), 119327. https://doi.org/10.1016/j.eswa.2022.119327
14. Y. Li, T. Han, S. Tang, C. Huang, H. Zhou, Y. Wang, An improved differential evolution by hybridizing with estimation-of-distribution algorithm, *Inf. Sci.*, **619** (2023), 439–456. https://doi.org/10.1016/j.ins.2022.11.029
15. J. Tvrdík, I. Křivý, Hybrid differential evolution algorithm for optimal clustering, *Appl. Soft Comput.*, **35** (2015), 502–512, https://doi.org/10.1016/j.asoc.2015.06.032
16. W. Gong, Z. Cai, C. Ling, De/bbo: A hybrid differential evolution with biogeography-based optimization for global numerical optimization, *Soft Comput.*, **15** (2010), 645–665. https://doi.org/10.1007/s00500-010-0591-1
17. C. Yu, L. Kelley, S. Zheng, Y. Tan, Fireworks algorithm with differential mutation for solving the cec 2014 competition problems, in *2014 IEEE Congress on Evolutionary Computation (CEC)*, (2014), 3238–3245. https://doi.org/10.1109/CEC.2014.6900590
18. J. Ji, H. Xiao, C. Yang, Hfade-fmd: a hybrid approach of fireworks algorithm and differential evolution strategies for functional module detection in protein-protein interaction networks, *Appl. Intell.*, **51** (2021), 1118–1132. https://doi.org/10.1007/s10489-020-01791-4
19. Y. Zheng, X. Xu, H. Ling, S. Chen, A hybrid fireworks optimization method with differential evolution operators, *Neurocomputing*, **148** (2015), 75–82, https://doi.org/10.1016/j.neucom.2012.08.075
20. X. Zhang, X. Zhang, UAV path planning based on hybrid differential evolution with fireworks algorithm, in *Advances in Swarm Intelligence*, (2022), 354–364. https://doi.org/10.1007/978-3-031-09677-8_30
21. J. Li, Y. Tan, Loser-out tournament-based fireworks algorithm for multimodal function optimization, *IEEE Trans. Evol. Comput.*, **22** (2018), 679–691. https://doi.org/10.1109/TEVC.2017.2787042
22. P. Hong, J. Zhang, Using population migration and mutation to improve loser-out tournament-based fireworks algorithm, in *Advances in Swarm Intelligence*, (2021), 423–432. https://doi.org/10.1007/978-3-030-78743-1_38
23. Y. Li, Y. Tan, Multi-scale collaborative fireworks algorithm, in *2020 IEEE Congress on Evolutionary Computation (CEC)*, (2020), 1–8. https://doi.org/10.1109/CEC48606.2020.9185563

24. S. Li, F. Liu, Adaptive niche radius fireworks algorithm for multi-modal function optimization, in *2021 4th International Conference on Intelligent Autonomous Systems (ICoIAS)*, (2021), 205–210. https://doi.org/10.1109/ICoIAS53694.2021.00044

25. J. Zhang, W. Li, Last-position elimination-based fireworks algorithm for function optimization, in *Advances in Swarm Intelligence*, (2019), 267–275. https://doi.org/10.1007/978-3-030-26369-0_25

26. M. Chen, Y. Tan, Exponentially decaying explosion in fireworks algorithm, in *2021 IEEE Congress on Evolutionary Computation (CEC)*, (2021),1406–1413. https://doi.org/10.1109/CEC45853.2021.9504974

27. X. Shen, Q. Wang, Y. Huang, Y. Xuan, An enhanced multi-modal function optimization fireworks algorithm based on loser-out tournament, *J. Syst. Simul.*, **32** (2020), 9–19. https://doi.org/10.16182/j.issn1004731x.joss.19-0225

28. Y. Li, Y. Tan, Enhancing fireworks algorithm in local adaptation and global collaboration, in *Advances in Swarm Intelligence*, (2021), 451–465. https://doi.org/10.1007/978-3-030-78743-1_41

29. I. Tuba, I. Strumberger, E. Tuba, N. Bacanin, M. Tuba, Performance analysis of the fireworks algorithm versions, in *Advances in Swarm Intelligence*, (2021), 415–422. https://doi.org/10.1007/978-3-030-78743-1_37

30. J. Li, S. Zheng, Y. Tan, The effect of information utilization: Introducing a novel guiding spark in the fireworks algorithm, *IEEE Trans. Evol. Comput.*, **21** (2017), 153–166. https://doi.org/10.1109/TEVC.2016.2589821

31. J. Liang, B. Qu, P. Suganthan, A. Hernández-Díaz, Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization, (2013).

32. F. Wilcoxon, *Individual Comparisons by Ranking Methods*, Springer, New York, 1992.

AIMS Press