Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Firefly algorithm with division of roles for complex optimal scheduling*

Jia ZHAO[1,2], Wenping CHEN[1], Renbin XIAO[‡3], Jun YE[1]

[1]*School of Information Engineering, Nanchang Institute of Technology, Nanchang 330099, China*

[2]*Jiangxi Province Key Laboratory of Water Information Cooperative Sensing and Intelligent Processing, Nanchang 330099, China*

[3]*School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China*

E-mail: zhaojia925@163.com; chen_9731@163.com; rbxiao@hust.edu.cn; yejun68@sina.com

Received Dec. 10, 2020; Revision accepted Mar. 9, 2021; Crosschecked Sept. 9, 2021

Abstract: A single strategy used in the firefly algorithm (FA) cannot effectively solve the complex optimal scheduling problem. Thus, we propose the FA with division of roles (DRFA). Herein, fireflies are divided into leaders, developers, and followers, while a learning strategy is assigned to each role: the leader chooses the greedy Cauchy mutation; the developer chooses two leaders randomly and uses the elite neighborhood search strategy for local development; the follower randomly selects two excellent particles for global exploration. To improve the efficiency of the fixed step size used in FA, a stepped variable step size strategy is proposed to meet different requirements of the algorithm for the step size at different stages. Role division can balance the development and exploration ability of the algorithm. The use of multiple strategies can greatly improve the versatility of the algorithm for complex optimization problems. The optimal performance of the proposed algorithm has been verified by three sets of test functions and a simulation of optimal scheduling of cascade reservoirs.

Key words: Firefly algorithm (FA); Division of roles; Cauchy mutation; Elite neighborhood search; Optimal scheduling
https://doi.org/10.1631/FITEE.2000691 **CLC number:** TP301.6

1 Introduction

Since the 1950s, the scheduling problem has attracted attention in such fields as applied mathematics, operation research, and engineering technology (Xu et al., 2004). The scheduling problem is usually defined as "allocating a group of resources to execute a group of tasks," and has been proved to be an NP complete problem (Cook, 1971; Wu N, 2020). With the development of the scheduling problem towards the direction of a large number of resources, diverse scheduling objectives, and coupled constraint conditions, the optimal scheduling problem becomes increasingly complex. Take the optimal scheduling of cascade reservoirs as an example. Because of the uneven spatial and temporal distribution of water resources and the urgent social need for energy, a number of large-scale cascade reservoir groups have been formed in China. There exists hydrological, hydraulic, and electric coupling among the cascade reservoir groups which have to meet the multiple constraints of domestic water consumption, water balancing equation, and water outflow. To achieve social, economic, ecological, and other multi-dimensional objectives, the optimal scheduling of cascade reservoirs has become a complex nonlinear scheduling problem with multiple objectives, multiple constraints, multiple stages, and strong coupling (Moeini and Babaei, 2020).

‡ Corresponding author

* Project supported by the National Science and Technology Innovation 2030 Major Project of the Ministry of Science and Technology of China (No. 2018AAA0101200), the National Natural Science Foundation of China (Nos. 52069014 and 51669014), and the Science Foundation for Distinguished Young Scholars of Jiangxi Province, China (No. 2018ACB21029)

ORCID: Jia ZHAO, https://orcid.org/0000-0002-3652-1903; Renbin XIAO, https://orcid.org/0000-0003-0951-2734

Dynamic programming (DP), as one of the most widely used methods to solve the scheduling problem, divides the problem into several stages to make decisions, and then obtains the optimal decision scheme for the whole system. However, DP features "curse of dimensionality" (Yu BH et al., 2004). When the amount of resources in the scheduling problem increases, the computer storage and computational complexity will increase dramatically. Because of the NP nature of the problem, it is impossible to solve it using only the scheduling techniques and methods based on analytic optimization. Hence, there have been attempts to use artificial, computational, and real-time intelligence to solve the problem (Xu et al., 2004). From the statistical results of a large number of available studies, the computational intelligence method is one of the most effective and promising scheduling methods (Xiao et al., 2015; Xiao and Wang, 2018; Tian et al., 2020). Common computational intelligence methods include QUATRE (Meng et al., 2016), fish migration optimization (FMO) (Guo et al., 2021), the Phasmatodea population evolution algorithm (PPE) (Song et al., 2020), the wolf pack algorithm (WPA) (Wu HS et al., 2020), particle swarm optimization (PSO) (Zhang HW et al., 2017; Zhao et al., 2017a, 2017b), the simulated annealing algorithm (SA) (Zou et al., 2016), surrogate-assisted hybrid optimization (SAHO) (Pan et al., 2020), and cuckoo search (CS) (Zhang MQ et al., 2018). The high performance of the computational intelligence method sees it being widely used in QR code (Pan et al., 2021), systematic optimization (Cui et al., 2019; Wang GG et al., 2020), clustering (Fan et al., 2021; Lv et al., 2021; Zhao et al., 2021), and other fields.

Firefly algorithm (FA) (Yang, 2008; Lv and Zhao, 2018; Lv et al., 2019), one of the computational intelligence algorithms, was inspired by the behavior of fireflies attracting mates by giving out light. FA regards the randomly generated solution as the firefly, and each solution is assigned a brightness according to its performance in the objective function. According to the rule that a firefly will be attracted by another brighter firefly, fireflies attract each other to achieve the optimization. FA is simple in concept and easy to implement, and combines the characteristics of PSO and SA, so that the motion of particles has more diversity (Fister I et al., 2013). The results show that FA can effectively solve the scheduling problem, enjoying good optimization performance (Gope et al., 2016; Kassandra et al., 2018).

However, FA uses a single learning strategy. According to the theorem of no free lunch, the average performance of any two solutions is the same when solving the scheduling problem (Fister I et al., 2013). That is, one method is efficient in solving one kind of problem, while the performance of solving another type of problem will be poorer. Hence, for complex optimal scheduling problems, the optimization performance of FA cannot be guaranteed.

In nature, the survival and reproduction of a population cannot be separated from a division of roles. For instance, an ant colony generally includes the queen ant, female ants, male ants, worker and soldier ants, and each type of ant is engaged in different work. A large project in human society is often subdivided into many subprojects, and each subproject is given to the people who are good at it in the team, and each person performs his or her own duties to better complete the project through cooperation. As shown by the above natural or social phenomena, division of roles can improve the working effect and efficiency of groups.

To solve the complex optimal scheduling problem effectively, in this paper, based on the idea of role division of labor, we integrate a variety of learning strategies into an FA, and propose an FA with division of roles (DRFA), which improves the versatility of the algorithm to various complex optimal scheduling problems. In addition, a new step attenuation method, i.e., stepped down step, is designed to meet the demand of firefly movement for the search step in different evolution periods, and to further improve the optimization performance of the algorithm.

To deal with the defect of a single learning mechanism of FA, at the core of DRFA are the role (hierarchical) division and labor division (multi-strategy) cooperation of fireflies. Role partition can balance the development and exploration ability of the algorithm, and the use of multiple strategies can greatly improve the versatility of the algorithm for different optimization problems. First, DRFA classifies the fireflies into three roles, leaders, developers, and followers, according to brightness; then it allocates three learning strategies to each role so that each role can perform its own duties. The leader takes the greedy Cauchy mutation strategy. Since the

scheduling problem is often multimodal, there are multiple local optimizations. Cauchy mutation enables the leader to jump out of the local optimum and to prevent the population evolution from falling into trouble. At the same time, the greedy strategy can retain a better scheduling scheme already found by the algorithm, and provide better guidance for population evolution. The developer adopts the elite neighborhood search strategy, regards the leader as an elite particle, conducts fine search around the leader, and adjusts the better scheduling scheme, which further improves the scheduling effect of the scheduling scheme and enhances the ability of algorithm development. The follower learns from two excellent particles at the same time, fuses the information of the two scheduling schemes, forms a new scheduling scheme, weakens the impact of a single scheduling scheme on the scheduling results, and enhances the exploration ability of the algorithm. Finally, after each generation of evolution is completed, the roles of fireflies are re-divided according to performance.

It has been verified by three sets of test functions and one simulation of optimal scheduling of cascade reservoirs that the role division of the DRFA algorithm can well balance the development and exploration ability of the algorithm, and the use of multiple strategies improves the applicability of the algorithm to different optimization problems.

2 Firefly algorithm

Let D be the dimension of the search space. The position of each firefly in the search space $X_i=\{x_{i1}, x_{i2}, \ldots, x_{iD}\}$ represents a feasible solution, and the fitness of the optimization problem $f(X_i)$ represents its brightness. The fireflies move to the brighter ones in turn, and the position updating equation for firefly i to be attracted by the brighter firefly j is defined as

$$\boldsymbol{x}_i(t+1)=\boldsymbol{x}_i(t)+\beta(\boldsymbol{x}_j(t)-\boldsymbol{x}_i(t))+\alpha\boldsymbol{\varepsilon}, \tag{1}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the positions of fireflies i and j, respectively, β is the attraction, step factor α=0.2, t is the current number of iterations, and $\boldsymbol{\varepsilon}$ is a vector with the value in each dimension being a random number obeying the uniform distribution in [−0.5, 0.5].

The brightness and attractiveness of the firefly are calculated as follows:

$$I=I_0\mathrm{e}^{-\gamma r_{ij}^2}, \tag{2}$$

$$\beta=\beta_0\mathrm{e}^{-\gamma r_{ij}^2}, \tag{3}$$

where I_0 and β_0 are the initial brightness and maximum attractiveness of the firefly respectively, β_0 is generally taken as 1, γ is the light absorption coefficient which is generally set as a constant 1, and r_{ij} is the Euclidean distance between fireflies i and j, determined by

$$r_{ij}=\left\|\boldsymbol{x}_i(t)-\boldsymbol{x}_j(t)\right\|=\sqrt{\sum_{d=1}^{D}\left(x_{id}(t)-x_{jd}(t)\right)^2}. \tag{4}$$

FA distinguishes two asymptotic behaviors. The former appears when $\gamma\to0$ and the latter appears when $\gamma\to\infty$. If $\gamma\to0$, the attractiveness becomes $\beta=\beta_0$. That is, the attractiveness is constant anywhere within the search space. This behavior is a special case of PSO. If $\gamma\to\infty$, the second term falls out of Eq. (1), and the firefly movement becomes a random walk. This behavior is essentially a parallel version of SA. In fact, each implementation of FA can be between these two asymptotic behaviors. As a result, the motion of firefly particles is more diverse (Fister I et al., 2013).

3 Firefly algorithm with division of roles

DRFA first divides firefly populations into leaders, developers, and followers, gives different learning strategies to each role, and integrates various learning strategies into the algorithm with the idea of role division. Second, a new variable step size method is proposed to meet the step size requirement of the algorithm in different periods.

The algorithm introduces some new symbols, as given in Table 1.

3.1 Division of roles of fireflies

Division of roles can be seen everywhere in human society and in nature. Division of labor exists in human society because of differences in individual abilities. In the agricultural era, men plowed the fields and women wove cloth, caused by the differences in men's and women's abilities. Men are strong and can undertake high-intensity physical work, while women are physically weak but delicate in mind, so they are more suitable for meticulous work. In nature, division

of roles also exists due to the difference of individual structure in the same population, and different roles do different jobs. For example, there is an ant queen, female ants, male ants, worker and soldier ants in the colony, and the bee colony is composed of the queen bee, drones, and worker bees. It can be seen that division of roles using advantages of different individuals can improve the working effect and efficiency of the colony.

Table 1 Description of symbols in the DRFA algorithm

Symbol	Description
cauchy()	The random number generated by the Cauchy distribution
gbest	Global optimal particle
r_1–r_5	The random numbers within (0, 1)
$\boldsymbol{S}$	The domain length of the search space
$\tau(i)$	The attenuation sequence of the step size factor and $\tau(i)=i$
c	The attenuation period of the step size factor

Division of roles also exists in the firefly population. The male fireflies have wings and often fly in the sky. Some species of female fireflies have no wings, and they usually move on the ground. When the firefly sends out the courtship signal, the female firefly will hover on the ground or be on low vegetation, while the male firefly will fly around in the sky searching for his favorite object (Lloyd, 1971; Ohba, 2004). Based on different roles, many improved gender-based FAs (Takeuchi et al., 2015; Ritthipakdee et al., 2017; Wang CF and Song, 2019; Alomoush et al., 2020) have been proposed. Takeuchi et al. (2015) made male fireflies be attracted by all fireflies and female fireflies attracted only by male ones. In Ritthipakdee et al. (2017), based on the idea of a genetic algorithm, a pair of fireflies with the highest mutual attraction value was selected as the parent, and the offspring was generated through crossover and mutation operations. Alomoush et al. (2020) set up a partner list mechanism to determine whether a firefly was attracted by another firefly. Wang CF and Song (2019) divided the firefly population into female and male, and the firefly was attracted only by the opposite sex. According to the different gender, different update equations were set to make the male firefly focus on global search, and the female firefly perform local development.

In this study, the idea of division of roles is used to improve FA. Different from the above division methods based on gender difference, the division of labor in this study is based on the different performances of fireflies in the population. In FA, there are performance differences between different fireflies. Generally speaking, fireflies with strong light intensity store more dominant information, and hence can find a better position around them, so they are suitable for the development of the algorithm search space and improvement of the accuracy of algorithm optimization; the fireflies with weak light intensity are more discretely distributed, so they are suitable for the exploration of the algorithm search space to explore more unknown areas. Hence, the idea of division of labor can be used to assign different roles to fireflies with different performances, and each role uses different strategies to optimize the search space to improve the performance of the algorithm. In addition, in the gender-based division of roles, the roles and division of labor of each firefly are fixed at the beginning, and will not change. However, the performance of fireflies will change during the running of the algorithm, and fireflies with different performances have different working potentials. Therefore, in this study we divide the roles according to performance. After each generation is updated, the firefly performance is re-assessed and the roles and division of labor are assigned to the fireflies. The fireflies are divided into leaders, developers, and followers. The following details the specific division of each role in the algorithm:

1. Leader

In FA, the optimal particle is in the absolute leading position and attracts all the particles close to it. This mechanism is called the “single leader” mechanism, and makes the firefly movement appear as shown in Fig. 1.

After the division of labor, there are multiple leaders in the population. This is called the “multi-leader” mechanism. In the early stage of the algorithm, the firefly population is led by multiple leaders at the same time. This greatly increases the diversity of the firefly motion, and the firefly individuals have more opportunities to explore the extreme points in the space. When the firefly finds the best or the extreme point with significant advantages, according to the survival of the fittest mechanism, the leader will be replaced by the particle near the point. When multiple

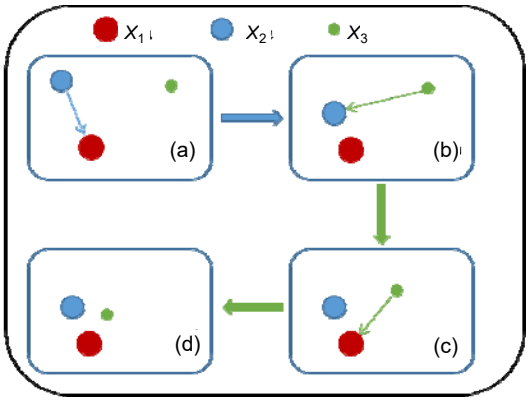

Fig. 1 Firefly motion analysis of the "single leader" mechanism

In (a), there are three fireflies, which are marked by their brightness as X_1, X_2, and X_3. Suppose that first X_2 moves towards X_1 to the position in (b). As shown in (b) and (c), X_3 moves towards X_2 and X_1 once each. At this time, X_2 moves to the neighborhood of X_1, so the two motions of X_3 can be approximately regarded moving towards X_1 twice. In this way, the situation as shown in (d) may occur, where all particles quickly gather together; that is, population evolution generation occurs, and a single particle approaches the optimal particle several times. In this case, the attractiveness of the optimal particle is too strong, which makes the population quickly gather together; the diversity of the population is rapidly lost, the algorithm easily falls into a local optimum, and the firefly individual is unable to jump out of the local optimum.

leaders are near the same extreme point, the population development of the extreme point will be accelerated. Hence, the "multi-leader" mechanism can increase the global exploration ability of the algorithm without affecting its local development ability. Compared with the "single leader" mechanism, the "multi-leader" mechanism weakens the leadership of the optimal particle, delays all particles in gathering near the optimal particle, and improves the ability of particles to jump out of the local optimum.

The existence of the leader ensures that the population evolves in a relatively correct direction. Here, the relative correctness means that the leader is in a relatively good position, that is, the superior extreme point (local optimum). The purpose of the optimization algorithm is to find the extreme point with the absolute advantage (global optimum). Hence, to prevent the leader from falling into the local optimum, it is necessary to give the leader a mechanism to jump out and prevent the population evolution from falling into trouble. The greedy Cauchy mutation strategy is used for the leader. The probability density function of the standard Cauchy distribution is shown in Fig. 2. Some studies (Wang H et al., 2017b; Sun et al., 2019) have shown that the long tail of the Cauchy distribution can help the trapped firefly jump to a better position, and the greedy strategy can save the dominant information of the population. The equation of the greedy mutation strategy is as follows:

$$\boldsymbol{x}_i(t+1) = \begin{cases} \boldsymbol{x}_i(t)+\textbf{cauchy}(), f(\boldsymbol{x}_i(t)+\textbf{cauchy}()) < f(\boldsymbol{x}_i(t)), \\ \boldsymbol{x}_i(t), f(\boldsymbol{x}_i(t)+\textbf{cauchy}()) \geq f(\boldsymbol{x}_i(t)), \end{cases} \tag{5}$$

where **cauchy**() represents the random number generated by the Cauchy distribution, $f(\boldsymbol{x}_i(t)+\textbf{cauchy}())$ represents the fitness value after Cauchy mutation by the firefly, and $f(\boldsymbol{x}_i(t))$ represents the current fitness value of the firefly.

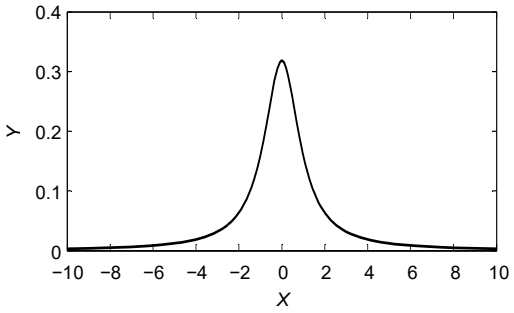

Fig. 2 The standard Cauchy distribution probability density function

With the greedy strategy, if the leader finds a better position after Cauchy mutation, he/she will jump to the position directly and lead the population out of the dilemma; if the Cauchy mutation finds the same or worse position, the leader will not move and the dominant information of the population is saved.

2. Developer

The developer undertakes the local development of the algorithm. To increase the convergence speed and development accuracy of the algorithm, we propose an elite neighborhood search strategy. The neighborhood search mechanism is to assign a neighborhood structure to the current individual to be optimized.

In the process of evolution, the algorithm evolves in the neighborhood rather than the individual. Wang H et al. (2013) introduced a global neighborhood search mechanism in PSO, and Zhou et al. (2017) integrated the search mechanism into the artificial bee colony (ABC) algorithm because of its high efficiency in PSO. The results showed that the search mechanism also has good performance in ABC. Then, based on the search mechanism, Wang H et al. (2017b) changed two particles randomly selected from the population to two excellent particles randomly selected from the k neighborhood of the current particle, so as to realize the local search of the current particle, greatly improving the FA performance.

Due to the excellent performance of this search mechanism in various algorithms, we take the leader as the neighborhood of the developer. This is called the elite neighborhood, and we propose an elite neighborhood search strategy to realize the local development of the leader. In addition, the unique step mechanism in FA is added to the search mechanism.

The search equation of the elite neighborhood is as follows: the first three terms in the right-hand side of Eq. (6) represent the global neighborhood search mechanism in PSO, introduced by Wang H et al. (2013):

$$\boldsymbol{x}_i(t)=r_1\boldsymbol{x}_i(t)+r_2\mathbf{gbest}+r_3(\boldsymbol{x}_j(t)-\boldsymbol{x}_k(t))+\alpha(t)\boldsymbol{S}\odot\boldsymbol{\varepsilon}, \tag{6}$$

where **gbest** is the global optimal particle, j and k are two particles randomly selected in the leader ($j\neq k$), r_1, r_2, and r_3 are three evenly distributed random numbers, with the values within (0, 1) and $r_1+r_2+r_3=1$, and S is the domain length of the search space.

3. Follower

The task of the follower is to explore more unknown space in the process of following the excellent particles, that is, to undertake the exploration work of the algorithm. To prevent the population from being bounded by the explored space, the follower's movement will be more diverse. In addition, to improve the probability of the population exploring the unknown excellent space, the number of followers in the population will be sufficient. Because of the large number of followers, we divide the followers into several layers. The better the fitness value is, the higher the layer of the firefly is. Each follower randomly selects two dominant particles from the leader, the developer, and the follower higher than itself for learning.

Since the followers of different levels have different learning objects, their learning ability is different. For example, the learning objects of the top-level followers are the leaders and the developers, whose learning objects are few but excellent, thus having strong development but weak exploration ability. The learning objects of the secondary high-level followers are the leaders, the developers, and the top-level followers. Compared with the top-level followers, the secondary high-level followers have more learning objects and more complex components, so their development ability becomes weaker and their exploration ability increases. The learning objects of the lowest-level followers are all the objects except those at their own level. Because of the diversity of their learning objects, the development ability is weak and the exploration ability is strong. Hence, layering of the followers can not only improve the diversity of the algorithm, but also help maintain the balance between the exploration and development of the algorithm.

The learning equation for the follower is

$$\begin{aligned}\boldsymbol{x}_i(t+1)=&\boldsymbol{x}_i(t)+r_4\beta_1(\boldsymbol{x}_j(t)-\boldsymbol{x}_i(t))\\&+r_5\beta_2(\boldsymbol{x}_k(t)-\boldsymbol{x}_i(t))+\alpha(t)\boldsymbol{S}\odot\boldsymbol{\varepsilon},\end{aligned} \tag{7}$$

where j and k are two particles randomly selected from the leader, the developer, and the layer higher than particle i, and r_4 and r_5 are random numbers within (0, 1), with $r_4+r_5=1$.

3.2 Stepped down step

In different stages of the algorithm, the firefly motion has different requirements for the step size. Generally speaking, in the early stage of the algorithm, the firefly needs a larger step size to detect as many unknown areas as possible in the search space. In the later stage of the algorithm, the firefly needs fine development. At this time, a small step size is needed to improve the optimization accuracy of the algorithm. In particular, when the distance r_{best} between the firefly and the global optimum is small enough, and the step factor α is much larger than r_{best} ($r_{\text{best}}/\alpha\to 0$), the random step size is equivalent to noise to the firefly, and interferes with firefly motion, making the firefly unable to approach the global optimum, and

the firefly population will stop evolving. In other words, if the search step size is too large, the local development ability of the algorithm will be restricted.

Wang et al. (2017b) verified by theoretical derivation that if FA is to converge, the step factor α needs to be close to 0. Many studies followed this principle in improving the step size, and a variety of step factors have been designed which decrease with the increase of the number of iterations. Based on this, we propose a stepped down type of step factor attenuation method, which first constructs a sequence τ=1, 2, …, and lets the step factor α be transformed as

$$\alpha(t+1)=\begin{cases}\alpha(t)/\tau(i), & \mathrm{mod}(t,c)=0,\\ \alpha(t), & \text{otherwise},\end{cases} \tag{8}$$

where t is the current number of iterations, c is the attenuation period (that is, α is attenuated once every time the algorithm iterates for c), and i is the current attenuation number of the step factor. As α=0.2 in the standard FA, $\alpha(0)$=0.2.

With the operation of the algorithm, the distribution of the firefly population in the search space is aggregated gradually, with accelerated development of local areas. Therefore, the attenuation speed of the search step is increased iteratively. Because the sequence τ is increasing, the attenuation range of the step factor α can be increased.

3.3 Procedure of the DRFA algorithm

Based on the concept described above, the pseudocode of DRFA is shown in Algorithm 1. Details that supplement the pseudo-code include:

Line 2: The initialized parameters include the problem dimension D, the firefly population size N, the maximum number of evaluation times MAX_FEs, the light absorption coefficient γ, the maximum attractiveness β_0, and the initial step size $\alpha(0)$.

Line 7: The fireflies are sorted according to brightness, and the sorted list is divided into groups of leaders, developers, and followers according to the proportion of l:d:h, where h is the number of layers divided in the followers, marked as k=l+d+h. p and q are the quotient and the remainder of N/k, respectively. Then the number of leaders is lp, the number of followers is dp, the number of followers in each layer except those at the lowest layer is p, and the number of followers at the lowest layer is p+q.

Algorithm 1 DRFA

```
1  Begin
2    initialize the algorithm parameters
3    initialize a population of fireflies randomly
4    calculate the fitness values of each firefly
5    FEs=N                    // N is the firefly population
6    while (FEs<MAX_FEs)
7      sort and divide firefly populations
8      update α according to Eq. (8)
9      for i=1 to lp            // leader
10       move firefly i according to Eq. (5)
11       FEs=FEs+1
12     end for
13     for i=lp+1 to (l+d)p       // developer
14       move firefly i according to Eq. (6)
15       FEs=FEs+1
16     end for
17     for i=(l+d)p+1 to N        // follower
18       pick two particles j and k at random from the high level
19       move firefly i according to Eq. (7)
20       FEs=FEs+1
21     end for
22   end while
23 End
```

4 Algorithm analysis

4.1 Experimental settings

In this section, DRFA is analyzed using 12 classical test functions, which were used in Wang H et al. (2017a). f_1–f_7 are single-mode functions with only one extreme point; these are used to test the local development ability of the algorithm. f_8–f_{12} are multi-mode functions with multiple extreme points in the definition domain; these are used to test the global exploration ability of the algorithm.

Some of the settings involved in the experiment are: the dimension of the optimization problem is set to be D=30, the maximum number of evaluation times MAX_ FEs=5×10^5, parameters β_0=1 and $\gamma=1/\Gamma^2$ (Γ is the domain length of the optimization function), and population size N=20. In addition, if the step factor is to be attenuated to 0 before the end of the algorithm, it needs to be attenuated 178 times, so the attenuation period c is set at c=MAX_FEs/178. The optimization result of the algorithm is the average value of 30 times of algorithm running.

4.2 Analysis of role matching in firefly population

Leaders, developers, and followers in the firefly population are divided according to the proportion of l:d:h. The work of the leader is to guide the direction of population evolution. The developer completes the algorithm development by local search of the leader. These two tasks require only a small number of fireflies, and there is no layer division inside developers and leaders. Since only a few fireflies are needed to maintain and tune a good scheduling scheme, many are used to find new scheduling schemes. Hence, for the convenience of the follow-up discussion, in this study we set l=d=1, and then the role proportion of the fireflies is leader:developer:follower=1:1:h.

In this subsection, the effect of different values of h on the performance of the DRFA algorithm is discussed through the experiment. Since the subtle interval has little influence on the algorithm result, one is taken as the interval, set up h=1, 2, …, 8, and the optimized average results are shown in Table 2. To compare the performances of the algorithms, the Friedman test is conducted on the optimization results of the algorithms, as shown in the last line of Table 2. The smaller the test value is, the better the algorithm performance is.

As shown in Table 2, different proportions do not affect the optimization performance of the algorithm on functions f_1, f_3, f_6, and f_9–f_{11}. On f_2 and f_4, when the proportion of leaders, developers, and followers is 1:1:1, the algorithm performs poorly, and the global optimum of the function can be found for other proportions. The reason is that the firefly population size is N=20, and when the ratio of leaders, developers, and followers is 1:1:1, the number of fireflies in each role is 6, 6, and 8. At this time, there are many particles in the elite neighborhood learned by the developer, which leads to the poor quality of the elite neighborhood and the decline of the development ability of the algorithm. Therefore, the optimization accuracy of the algorithm is lower than that of other proportions. On function f_5, the optimization performance of the algorithm decreases with the increase of h, and note that there is a large difference when h=6. On function f_7, the optimization results of different proportions are almost the same, and the performance is the best when h=2. On function f_8, the optimization performance of the algorithm decreases with the increase of h, and the performance is optimal when h=2. On function f_{12}, the optimization performance of the algorithm first increases and then decreases with the increase of h, the performance is optimal when h=4, and there is a turning point at h=6.

With respect to the stability of the algorithm with different role proportions, since the optimization results of different proportions running on functions f_1, f_3, f_6, and f_9–f_{11} for 30 times are the same, and the global optimum can be found for other values except h=1 on functions f_2 and f_4, the stability of different proportions on the eight functions is not discussed. Fig. 3 shows the box diagram of the optimization results of the algorithm running on functions f_5, f_7, f_8, and f_{12} 30 times. To show the difference more intuitively, the natural logarithm is used for f_5, f_7, and f_{12}.

Table 2 Optimization results of the DRFA algorithm with different role proportioning

Function	Optimization result							
	h=1	2	3	4	5	6	7	8
f_1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_2	7.83E-246	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_3	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_4	2.22E-246	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_5	**6.53E-06**	1.26E-05	2.20E-05	2.75E-05	7.53E-05	7.93E-01	5.17E-05	6.20E-05
f_6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	1.06E-06	**9.09E-07**	2.12E-06	2.15E-06	1.35E-06	1.46E-06	3.42E-06	1.11E-06
f_8	−1.21E+04	**−1.22E+04**	−1.20E+04	−1.19E+04	−1.16E+04	−1.17E+04	−1.16E+04	−1.16E+04
f_9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{10}	**5.89E-16**	**5.89E-16**	**5.89E-16**	**5.89E-16**	**5.89E-16**	**5.89E-16**	**5.89E-16**	**5.89E-16**
f_{11}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{12}	9.44E-12	4.10E-12	1.45E-13	**2.26E-14**	6.91E-14	1.07E-13	7.36E-14	7.80E-14
Rank	4.67 (5)	**3.83 (1)**	4.42 (3)	4.25 (2)	4.58 (4)	4.83 (8)	4.75 (7)	4.67 (5)

The best results are in bold

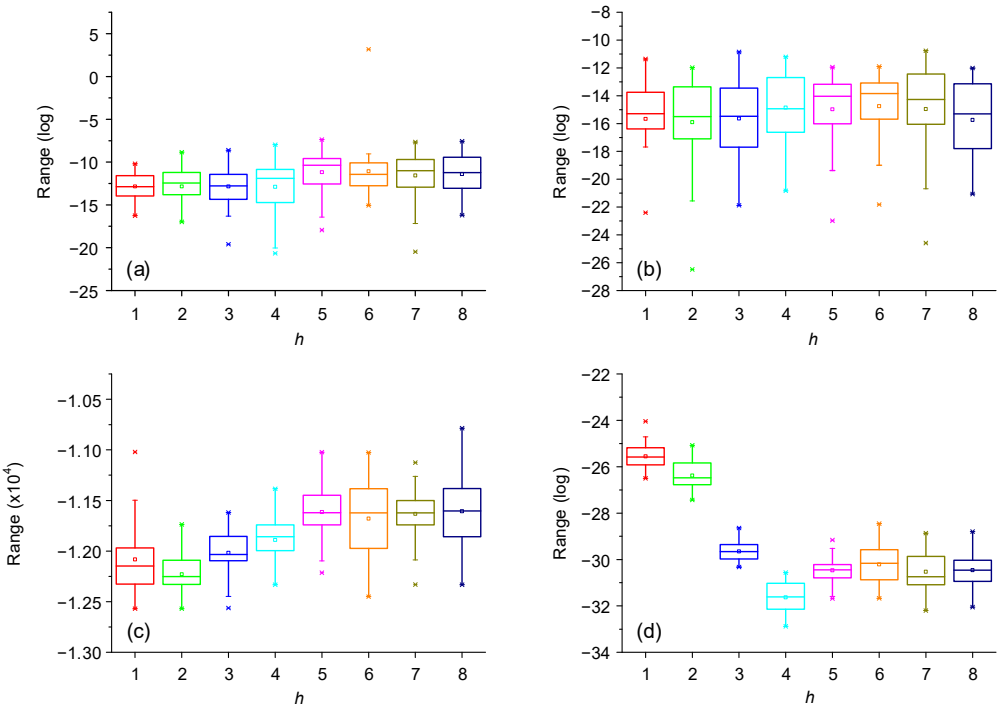

Fig. 3 Box diagram of the optimization results of the DRFA algorithm with different role proportions: (a) Rosenbrock (f_5); (b) quartic with noise (f_7); (c) Schwefel 2.26 (f_8); (d) penalized (f_{12})

As shown in Fig. 3, there are lower outliers when h=3, 4, 5, 7 on function f_5. This indicates that the algorithm has a lower probability to achieve higher optimization accuracy, while there is a higher outlier when h=6 and the algorithm has a lower probability of obtaining a poor optimization result. It is this poor optimization result that makes DRFA show an average optimization result with a large difference from other proportions when h=6. Therefore, on function f_5, the algorithm is unstable when h=6. On function f_7, all proportions have lower outliers, the outlier is the lowest when h=2, and the algorithm has the opportunity to obtain higher optimization accuracy. On function f_8, the stability of all proportioning algorithms is poor. On function f_{12}, the stability of the algorithm is poor when h=6, so the optimized average value does not conform to the trend of first increasing and then decreasing. In general, the increase or decrease of h is not necessarily related to the stability of the algorithm, and the stability of the algorithm is poor when h=6.

As shown by the Friedman test results, the comprehensive optimization performance of the algorithm is the best when h=2. Hence, in the follow-up experiments, the proportion of leaders, developers, and followers of fireflies is set to be 1:1:2.

4.3 Effect of different roles

DRFA allocates various strategies to leaders, developers, and followers in the way of division of labor and cooperation. To explore the influence of roles and their division of labor on the algorithm, in this subsection we combine the roles and their division of labor. For clearer expression, each combination is named by the initials of each role name (Table 3). Because the optical attraction coefficient $\gamma=1/\Gamma^2$ of DRFA uses what is set in version FA2010 (Yang, 2010), for fair comparison, the same parameters are used in the standard FA. The first set of test functions is used to test the optimization performance of each combination.

The corresponding algorithms of each combination are run independently 30 times, and the average results are shown in Table 4. Fig. 4 shows the convergence curves of each algorithm on two single-mode functions f_1 and f_5 and two multi-mode functions f_8 and f_{12}.

As shown in Table 4, compared with FA, if the population is full of followers, the optimization accuracy of the algorithm on functions f_1, f_6, and f_{10} is greatly improved, while the performance on f_3 and f_8 is significantly reduced, and there is no significant

Table 3 Combination of roles in the DRFA algorithm

Combination strategy	Algorithm name
FA2010	FA2010
FA2010+Follower	F
FA2010+Developer	D
FA2010+Leader	L
FA2010+Follower+Developer	FD
FA2010+Follower+Leader	FL
FA2010+Developer+Leader	DL
FA2010+Follower+Developer+Leader	DRFA

difference on other functions. As shown by the test results, the strategy used by F has improved the performance of FA, but not by all that much. As indicated in Fig. 4, the F-combination converges in advance on every function. This suggests that the strategy used by the F-combination and the diversity of firefly learning objects can effectively improve the global search ability of the algorithm, but no strategy can improve the local development ability of the firefly, which prevents the optimization performance of the algorithm from being high. If the population is full of developers, the optimization accuracy of the algorithm on 12 functions is greatly improved, showing that the strategy used by the D-combination can improve the development ability of the algorithm. As shown in Fig. 4, the strategy used by the D-combination can also improve the convergence speed of the algorithm. If the population is full of leaders, all the fireflies take advantage of the greedy Cauchy mutation to change the position, and there is no information communication between fireflies, so the optimization efficiency of the algorithm is low, and the optimization performance is weaker than that of FA.

Compared with the single F, the FD-combination composed of follower-F and developer-D effectively addresses the problem of insufficient local development capacity of F, and greatly improves the optimization accuracy on 12 functions. Compared with the single D, although the optimization performance of FD on function f_8 is reduced, the performance on f_5 and f_{12} is improved, and the global search ability of FD is improved. As shown in Table 4, the performance of FD is better than that of F and D. Compared with F alone, the performance of the algorithm is significantly improved on f_8 using FL-combination, which shows that the strategy used by L can enhance the firefly's ability to jump out of the local optimum. Similarly, compared with single D, DL can significantly improve the optimization performance of the algorithm on function f_8.

In the three roles, if L is absent, the performance of the algorithm using the FD-combination is poor on function f_8. As shown in Fig. 4c, the algorithm falls into the local optimum prematurely and the population stops evolving. However, the combination of L, DL, FL with L and DRFA can find better points. This further proves that the strategy of L can improve the firefly's ability to jump out of the local optimum. If D is absent, the optimization accuracy of the algorithm using the FL-combination on 10 functions of f_1–f_5, f_7, and f_9–f_{12} is not high, which indicates that the algorithm lacks the ability of local development. In the absence of F, the performance of the algorithm using DL-combination is poor on functions f_5 and f_{12}, and the global search ability of the algorithm is insufficient. The DRFA algorithm composed of three role combinations can effectively make up for the

Table 4 Optimization results of the FA algorithm of different combinations of division of roles

Function	Optimization result							
	FA2010	F	D	L	FD	FL	DL	DRFA
f_1	1.84E-06	6.06E-72	**0.00E+00**	5.65E+01	**0.00E+00**	1.62E-70	**0.00E+00**	**0.00E+00**
f_2	1.65E-03	3.65E-05	**0.00E+00**	4.50E+01	**0.00E+00**	2.14E-04	**0.00E+00**	**0.00E+00**
f_3	6.19E-06	3.47E+01	**0.00E+00**	3.50E+02	**0.00E+00**	2.22E+01	**0.00E+00**	**0.00E+00**
f_4	5.94E-04	1.00E-02	**0.00E+00**	3.67E+00	**0.00E+00**	7.77E-03	**0.00E+00**	**0.00E+00**
f_5	4.38E+01	2.21E+02	3.33E-03	3.70E+04	**1.05E-05**	6.62E+01	4.24E-03	1.26E-05
f_6	2.67E-01	**0.00E+00**	**0.00E+00**	3.40E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	9.86E-02	1.85E-02	**2.21E-07**	1.17E+02	9.46E-07	1.95E-02	3.55E-07	9.09E-07
f_8	−8.39E+03	−6.40E+03	−1.04E+04	−1.17E+04	−8.79E+03	−1.21E+04	**−1.22E+04**	**−1.22E+04**
f_9	4.57E+01	3.50E+01	**0.00E+00**	3.13E+02	**0.00E+00**	4.03E+01	**0.00E+00**	**0.00E+00**
f_{10}	3.11E-04	2.21E-14	**5.89E-16**	6.76E+00	**5.89E-16**	2.10E-14	**5.89E-16**	**5.89E-16**
f_{11}	9.68E-03	4.35E-03	**0.00E+00**	9.31E-01	**0.00E+00**	5.50E-03	**0.00E+00**	**0.00E+00**
f_{12}	3.46E-03	7.14E-03	1.27E-06	3.37E+00	**9.69E-13**	1.73E-02	1.14E-06	4.10E-12
Rank	6.33 (7)	5.79 (6)	2.83 (4)	7.67 (8)	2.75 (3)	5.54 (5)	2.63 (2)	**2.46 (1)**

The best results are in bold

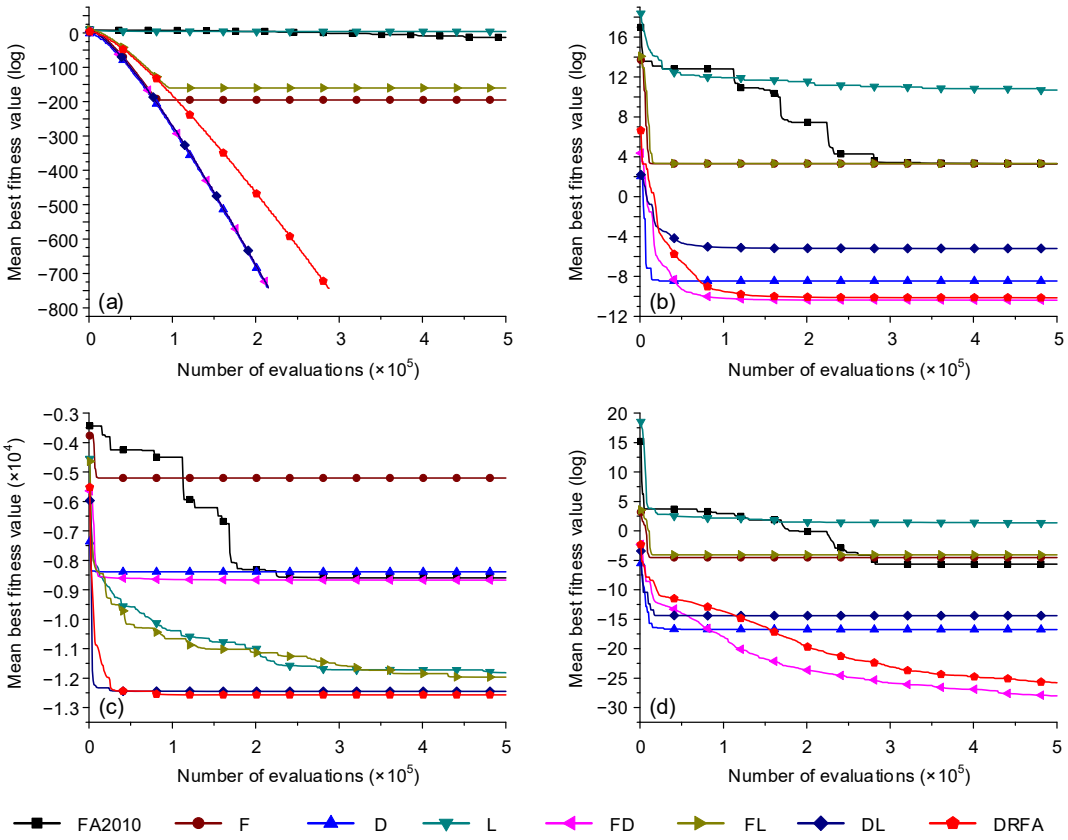

Fig. 4 Convergence curve of the FA algorithm with different combinations of division of roles: (a) sphere (f_1); (b) Rosenbrock (f_5); (c) Schwefel 2.26 (f_8); (d) penalized (f_{12})

deficiency of pairwise combination of roles, and ideal optimization results are achieved on 12 functions. As shown in Fig. 4, the convergence rate of DRFA on f_1, f_5, and f_{12} is lower than that of D, FD, and DL combination. This is because the leader uses the Cauchy mutation to perform the chaos searches around the leader, but such searches are low in efficiency and most of them are useless and a waste of the computing resources of the algorithm. However, considering the optimization performance, the Cauchy mutation does have a chance to make the algorithm jump out of a local optimum, so this kind of "waste" of computing resources is acceptable. As suggested by the Friedman test results, the comprehensive performance of the algorithm is the best when the combination of three roles is used.

5 Comparison of algorithms

In this section three sets of test functions are used to compare DRFA with related algorithms.

5.1 Comparison with FA and its improved algorithm

5.1.1 Experimental results and analysis

In this subsection, 12 test functions from the previous section are used to compare DRFA with FA (Yang, 2008), wise step strategy FA (WSSFA) (Yu SH et al., 2014), variable step size FA (VSSFA) (Yu SH et al., 2015), memetic FA (MFA) (Fister IJr et al., 2012), FA with random attraction (RaFA) (Wang H et al., 2016), FA with adaptive control parameters (ApFA) (Wang H et al., 2017a), randomly attracted FA with neighborhood search and dynamic parameter adjustment mechanism (NSRaFA) (Wang H et al., 2017b), FA with deep learning (DLFA) (Zhao et al., 2018), and FA based on level-based attracting and variable step size (LVFA) (Zhao et al., 2020).

The experimental conditions and DRFA parameters are consistent with those in the previous section, and other parameters of each algorithm are the same as those in corresponding references. The average

values and standard deviations of the optimization results are shown in Table 5.

In Table 5, the best optimization results are shown in bold. DRFA has the best optimization effect on 10 functions and finds the global optimum on 7 functions. NSRaFA has the best optimization effect on 6 functions and finds the global optimum on 3 functions. DLFA, at the third position, has the best optimization effect on 5 functions and finds the global optimum on 5 functions. As shown in Table 5, the results of DRFA are better than those of FA, WSSFA, and VSSFA on all test functions. Compared with MFA, DRFA has better performance on 11 functions, and the optimization results are the same only on function f_6. This is because f_6 is a step function and its global optimum is easy to find. Compared with RaFA, the performance of DRFA is better on 10 functions and equivalent on f_6 and f_{11}. The performance of DRFA is better on 10 functions than ApFA and LVFA, and equivalent on f_6; they all find the global optimum, but the performance of DRFA is poor on f_{12}. better on 6 functions, equivalent on f_6, f_9, f_{10}, and Compared with NSRaFA, the performance of DRFA is f_{11}, and worse on f_7 and f_{12}. Compared with DLFA, the performance of DRFA is better on 6 functions, equivalent on f_1, f_2, f_4, f_6, and f_9, and they both find the global optimum, but the performance of DRFA is worse on f_{12}. In general, the performance of DRFA is better on f_1–f_{11}, ranking first or second in all algorithms, but poor on f_{12}, ranking fifth among all algorithms, inferior to ApFA, NSRaFA, LVFA, and DLFA. This shows that the performance of DRFA in dealing with such problems needs to be improved. From the Wilcoxon test results, DRFA is significantly better than other algorithms except NSRaFA.

To intuitively compare the optimization performance of each algorithm on different functions, Table 6 shows the Friedman test results of each algorithm on single-mode, multi-mode, and all functions. In Table 6, the comprehensive performance of DRFA in the single-mode function is the best, but in the multi-mode function, DRFA is inferior to NSRaFA,

Table 5 Optimization results of 10 algorithms on 12 test functions

Function	Type	FA	WSSFA	VSSFA	MFA	RaFA	ApFA	NSRaFA	DLFA	LVFA	DRFA
f_1	Mean	6.67E+04	6.34E+04	5.84E+04	1.56E-05	5.36E-184	2.02E-44	4.11E-110	**0.00E+00**	7.65E-204	**0.00E+00**
	Std.	1.83E+04	4.91E+04	1.17E+04	2.31E-05	6.82E-184	2.85E-44	9.05E-110	**0.00E+00**	0.00E+00	**0.00E+00**
f_2	Mean	5.19E+02	1.35E+02	1.13E+02	1.85E-03	8.76E-05	1.83E-12	1.35E-55	**0.00E+00**	1.24E-102	**0.00E+00**
	Std.	1.42E+02	5.66E+02	3.93E+01	3.57E-03	7.58E-05	4.01E-12	6.22E-56	**0.00E+00**	6.24E-103	**0.00E+00**
f_3	Mean	2.43E+05	1.10E+05	1.16E+05	5.89E-05	4.91E+02	1.01E+01	1.59E-109	1.58E-09	1.53E-21	**0.00E+00**
	Std.	4.85E+04	4.60E+05	3.64E+04	4.52E-05	1.06E+02	5.92E+00	4.95E-109	1.10E-08	4.51E-20	**0.00E+00**
f_4	Mean	8.35E+01	7.59E+01	8.18E+01	1.73E-03	2.43E+00	1.30E-07	1.88E-55	**0.00E+00**	1.00E-96	**0.00E+00**
	Std.	3.16E+01	1.88E+01	2.32E+01	3.86E-03	1.87E+00	8.85E-08	5.87E-56	**0.00E+00**	2.70E-95	**0.00E+00**
f_5	Mean	2.69E+08	2.49E+08	2.16E+08	2.29E+01	2.92E+01	2.81E+01	2.85E+01	1.99E+00	3.82E+01	**1.26E-05**
	Std.	6.21E+07	2.76E+08	5.79E+07	3.25E+00	4.19E+00	3.98E-01	2.03E-02	1.62E+01	3.61E+02	**1.50E-04**
f_6	Mean	7.69E+04	6.18E+04	5.48E+04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	3.38E+03	5.12E+04	2.16E+03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	Mean	5.16E+01	3.24E-01	4.43E+01	1.30E-01	5.47E-02	2.76E-03	**4.41E-16**	1.26E-02	1.96E-03	9.09E-07
	Std.	2.46E+01	4.80E+01	1.72E+01	2.36E-01	4.65E-02	1.10E-02	**1.25E-16**	1.86E-02	4.54E-03	7.44E-06
f_8	Mean	−1.56E+03	−2.01E+03	−1.85E+03	−7.63E+03	−1.21E+04	−6.25E+03	−1.20E+04	−8.86E+03	−7.38E+03	**−1.22E-04**
	Std.	3.77E+03	3.23E+03	8.56E+02	8.72E+02	3.61E+02	1.26E+02	2.25E+02	2.04E+04	4.48E+03	**1.09E+03**
f_9	Mean	3.33E+02	3.61E+02	3.12E+02	6.47E+02	2.69E+01	1.21E+01	**0.00E+00**	**0.00E+00**	3.19E+01	**0.00E+00**
	Std.	6.28E+01	8.56E+01	4.18E+01	2.53E+01	1.52E+01	2.77E+00	**0.00E+00**	**0.00E+00**	5.07E+01	**0.00E+00**
f_{10}	Mean	2.03E+01	2.05E+01	2.03E+01	4.23E-04	3.61E-14	2.55E-14	**5.89E-16**	5.68E-15	5.09E-15	**5.89E-16**
	Std.	2.23E-01	5.56E-01	2.46E-01	3.35E-04	5.98E-14	6.48E-15	**0.00E+00**	9.64E-15	8.61E-15	**0.00E+00**
f_{11}	Mean	6.54E+02	6.09E+02	5.47E+02	9.86E-03	**0.00E+00**	3.33E-16	**0.00E+00**	2.99E-02	2.38E-03	**0.00E+00**
	Std.	1.69E+02	4.19E+02	1.29E+02	6.81E-03	**0.00E+00**	2.22E-16	**0.00E+00**	1.52E-01	2.45E-02	**0.00E+00**
f_{12}	Mean	7.16E+08	6.18E+08	3.99E+08	5.04E-08	4.50E-05	1.23E-16	**1.57E-32**	1.57E-32	1.57E-32	4.10E-12
	Std.	1.82E+08	8.38E+08	1.05E+08	3.27E-08	6.28E-04	1.59E-16	**0.00E+00**	1.50E-47	1.49E-47	1.38E-11
p		**0.002**	**0.002**	**0.002**	**0.003**	**0.005**	**0.013**	0.327	**0.043**	**0.016**	
w/*t*/*l*		12/0/0	12/0/0	12/0/0	11/1/0	10/2/0	10/1/1	6/4/2	6/5/1	10/1/1	

The best optimization results are in bold. The *p* values are obtained by the Wilcoxon test (García et al., 2009) for DRFA and other FAs. The *p* value less than 0.05 is in bold, indicating that DRFA is clearly superior to the compared algorithm. *w*/*t*/*l* indicates that compared with the algorithm, the performance of DRFA is better on *w* functions, equivalent on *t* functions, and worse on *l* functions

indicating that DRFA has strong local development capability. The rank mean value of DRFA is the smallest, indicating that the comprehensive performance of DRFA is the best among the 10 algorithms.

Table 6 Friedman test results of 10 algorithms on 12 test functions

Algorithm	Rank		
	Single-mode f_1–f_7	Multi-mode f_8–f_{12}	All f_1–f_{12}
FA	10.00 (10)	9.30 (10)	9.71 (10)
WSSFA	8.43 (8)	8.10 (8)	8.29 (8)
VSSFA	8.57 (9)	9.00 (9)	8.75 (9)
MFA	4.86 (5)	7.00 (7)	5.75 (7)
RaFA	5.86 (7)	4.40 (4)	5.25 (6)
ApFA	5.00 (6)	4.80 (6)	4.92 (5)
NSRaFA	3.71 (3)	**2.10 (1)**	3.04 (2)
DLFA	2.86 (2)	3.60 (3)	3.17 (3)
LVFA	3.86 (4)	4.40 (4)	4.08 (4)
DRFA	**1.86 (1)**	2.30 (2)	**2.04 (1)**

The best results are in bold

Fig. 5 shows the convergence process curves of 10 algorithms on 12 test functions. As shown in the figure, DRFA has faster convergence on functions f_1–f_{11}. On function f_{12}, although DRFA has faster convergence in the early stage, the convergence rate in the middle stage decreases continuously, lower than that of ApFA, NSRaFA, and DLFA. In the later stage, the convergence is premature and the optimization accuracy is no longer improved. On the single-mode test function, both DLFA and DRFA can find the global optimum on f_1, f_2, and f_4. However, the difference is that DLFA shows a jump phenomenon, and it suddenly jumps to 0 when the convergence accuracy is still very high, while DRFA continues to refine the search and the optimal value is constantly approaching the global optimum. Although DRFA has faster convergence, it cannot find the optimal value of the function in the early stage as can DLFA. On the multi-mode test function, there are several algorithms that can find the optimal value of f_9 and f_{11}, but DRFA has the smallest number of evaluations. Generally speaking, DRFA has high performance in terms of both optimization accuracy and optimization speed.

5.1.2 Time complexity analysis

For a given function f, the time complexity for calculating the function value is f, and $G_{\max}$ is the maximum number of iterations. For FA, WSSFA, VSSFA, MFA, and ApFA, because these algorithms use the full attraction model, there are two loop nests to traverse all fireflies, and the time complexity is $G_{\max}N^2(D+f)$. RaFA improves the attraction model of FA, and each firefly needs only to move once in an iteration, so the complexity is $G_{\max}N(D+f)$. NSRaFA adds a judgment mechanism based on the random attraction. If the learning object randomly selected by the firefly is better than itself, the firefly moves according to the standard FA formula; if the learning object is inferior, three neighborhood learning formulas are used to generate three candidate solutions, from which the optimal solution is selected to update the position. Therefore, the time complexity is $G_{\max}N(D+f)$ to $3G_{\max}N(D+f)$. DLFA first updates the firefly position using the random attraction model, and then constructs the generalized central particle so that it can carry out deep learning; after deep learning it is used to lead the population evolution, and its time complexity is $G_{\max}(2N+\text{DL\_count})(D+f)$ (DL_count is the number of deep learning times). LVFA uses a hierarchical learning model to ensure that the excellent position information found by the population is not lost, and the best particles in the population do not move, so its time complexity is $G_{\max}N(k-1)/k(D+f)$ (k is the number of layers).

As for DRFA, each particle in the population needs only to move once, and the time complexity is the same as that of RaFA, i.e., $G_{\max}N(D+f)$. Table 7 lists the time complexity of all algorithms. The time complexity of DRFA is lower than that of FA, WSSFA, VSSFA, MFA, ApFA, NSRaFA, and DLFA, equivalent to that of RaFA, but inferior to that of LVFA. Although the time complexity of DRFA is higher than that of LVFA, LVFA does not use the best part of the particle information for further mining. However, DRFA takes the greedy Cauchy mutation

Table 7 Time complexity of 10 algorithms

Algorithm	Time complexity
FA WSSFA VSSFA MFA ApFA	$G_{\max}N^2(D+f)$
RaFA	$G_{\max}N(D+f)$
NSRaFA	$G_{\max}N(D+f)$ to $3G_{\max}N(D+f)$
DLFA	$G_{\max}(2N+\text{DL\_count})(D+f)$
LVFA	$G_{\max}N(k-1)/k(D+f)$
DRFA	$G_{\max}N(D+f)$

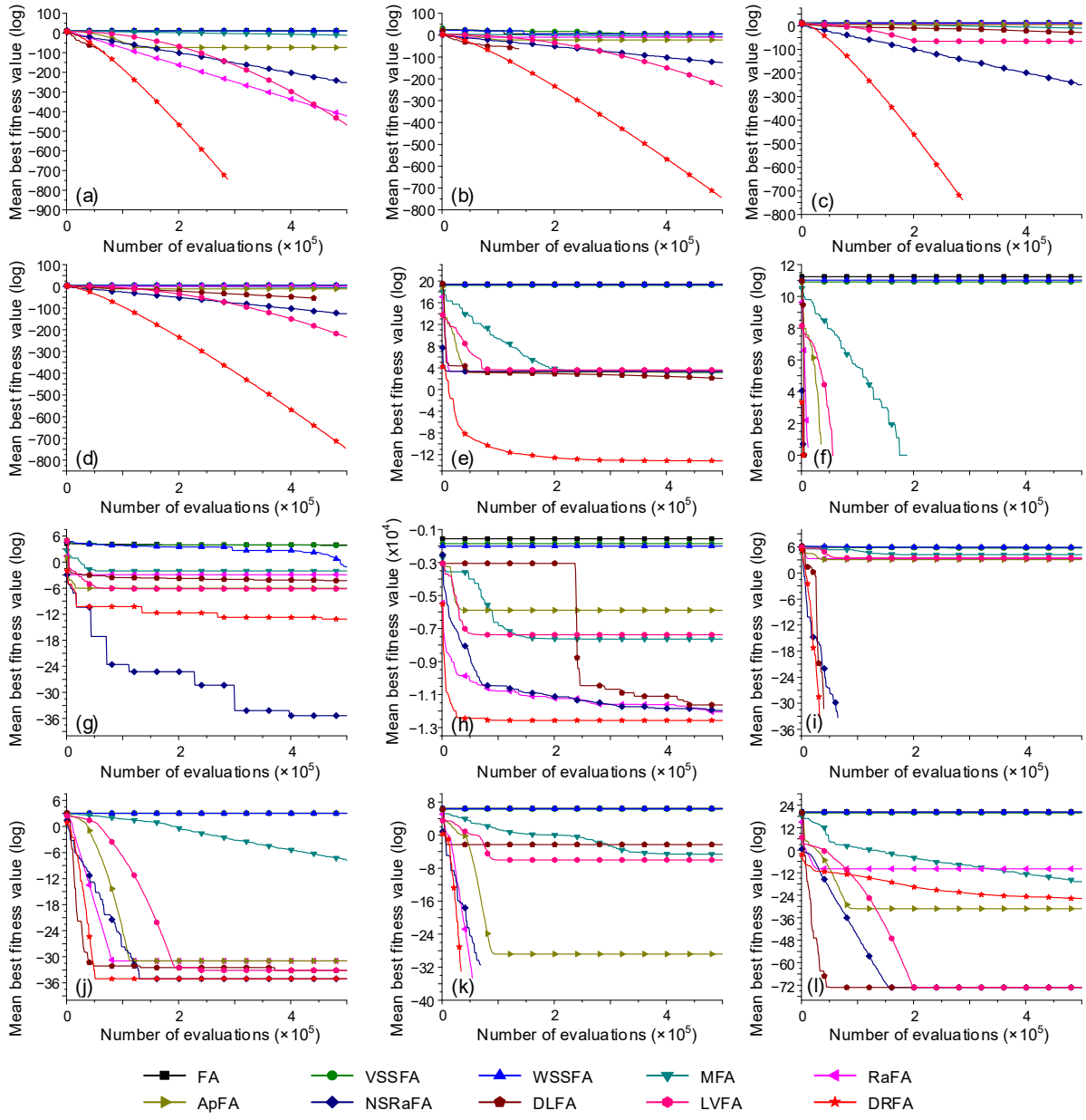

Fig. 5 Convergence process curves of 10 algorithms on 12 test functions: (a) sphere (f_1); (b) Schwefel 2.22 (f_2); (c) Schwefel 1.2 (f_3); (d) Schwefel 2.21 (f_4); (e) Rosenbrock (f_5); (f) step (f_6); (g) quartic with noise (f_7); (h) Schwefel 2.26 (f_8); (i) Rastrigin (f_9); (j) Ackley (f_{10}); (k) Griewank (f_{11}); (l) penalized (f_{12})

strategy for the leader, so that the population has the potential to jump out of the local optimum, which is better than LVFA.

5.2 Comparison with other algorithms

5.2.1 Comparison on 22 test functions

To further test the performance of DRFA, in this subsection 22 test functions are used (Sun et al., 2019) to compare DRFA with some classical and recent algorithms in the swarm intelligence field. Among them, f_1–f_9 are the single-mode functions; f_{10} is the single-mode function when in low dimension and the multi-mode function when in high dimension; f_{11}–f_{22} are multi-mode functions. The number of local optimal values increases exponentially with the increase of the dimension. The compared algorithms include improved differential evolution algorithms CoDE (Wang Y et al., 2011), JADE (Zhang JQ and Sanderson, 2009), and jDEscop (Brest and Maučec, 2011),

improved particle swarm optimization algorithms CLPSO (Liang et al., 2006) and HCOPSO (Sun et al., 2019), improved ABC algorithms AABCLS (Jadon et al., 2015), ABCVSS (Kiran et al., 2015), and BABC (Gao et al., 2015).

The dimension of the test function is set at D=50, and the maximum number of evaluation times of the algorithm is MAX_FEs=5000D. Refer to the corresponding references for specific parameters of each algorithm. The algorithm runs independently on each function 25 times, and its mean and standard deviation are shown in Table 8.

Table 8 Optimization results of 9 algorithms on 22 test functions

Function	Type	CoDE	JADE	jDEscop	CLPSO	HCOPSO	AABCLS	ABCVSS	BABC	DRFA
f_1	Mean	1.16E-37	2.50E-87	1.15E-36	3.59E-13	**0.00E+00**	4.60E-35	6.68E-23	1.01E-14	**0.00E+00**
	Std.	2.69E-37	8.62E-87	5.75E-36	1.50E-13	**0.00E+00**	1.67E-35	3.16E-32	5.07E-14	**0.00E+00**
f_2	Mean	1.25E-34	4.43E-78	4.77E-36	5.99E-11	**0.00E+00**	4.48E-33	1.11E-23	3.14E-29	**0.00E+00**
	Std.	1.50E-34	2.21E-77	1.85E-35	2.37E-11	**0.00E+00**	2.16E-33	4.17E-23	4.95E-29	**0.00E+00**
f_3	Mean	1.94E-38	2.27E-85	1.80E-41	4.78E-14	**0.00E+00**	2.54E-35	1.77E-33	1.23E-13	**0.00E+00**
	Std.	4.08E-38	1.14E-82	1.47E-16	1.33E-14	**0.00E+00**	1.24E-35	7.88E-33	6.16E-13	**0.00E+00**
f_4	Mean	6.20E-143	3.01E-100	3.58E-124	1.70E-65	**0.00E+00**	1.52E-50	7.21E-41	2.96E-92	**0.00E+00**
	Std.	3.15E-142	1.00E-99	1.79E-123	2.62E-65	**0.00E+00**	4.88E-50	3.56E-40	0.00E+00	**0.00E+00**
f_5	Mean	1.21E-20	1.89E-41	1.71E-22	6.75E-09	**0.00E+00**	4.98E-18	2.89E-18	2.18E-06	**0.00E+00**
	Std.	8.94E-21	8.32E-41	8.46E-22	1.61E-09	**0.00E+00**	1.34E-18	6.52E-18	1.06E-05	**0.00E+00**
f_6	Mean	6.00E-05	8.20E-10	2.52E+00	1.04E+01	**0.00E+00**	1.16E-01	2.06E+00	7.20E+00	**0.00E+00**
	Std.	9.51E-05	9.51E-10	5.61E-01	7.69E-01	**0.00E+00**	1.24E-01	3.77E-01	2.78E+00	**0.00E+00**
f_7	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_8	Mean	**2.67E-109**	**2.67E-109**	**2.67E-109**	**2.67E-109**	6.00E-106	**2.67E-109**	**2.67E-109**	**2.67E-109**	**2.67E-109**
	Std.	**9.67E-125**	**9.60E-125**	**9.65E-125**	**7.60E-116**	1.02E-104	**2.77E-119**	**3.08E-120**	**2.62E-116**	**2.59E-124**
f_9	Mean	8.17E-03	2.50E-03	3.91E-03	1.32E-02	3.20E-05	1.63E-02	6.14E-02	5.74E-02	**4.84E-06**
	Std.	2.79E-03	1.54E-03	1.47E-03	2.50E-03	1.74E-04	4.68E-03	1.38E-02	1.11E-02	**3.27E-05**
f_{10}	Mean	3.32E+01	3.19E-01	2.58E+01	5.44E+01	4.10E+01	3.09E+00	1.09E-01	6.29E-02	**5.27E-04**
	Std.	2.26E+01	1.10E+00	2.68E+01	2.37E+01	3.51E+00	1.36E+01	2.52E-01	1.24E-01	**1.37E-03**
f_{11}	Mean	7.34E-01	1.78E-11	1.03E-14	2.00E-04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	8.82E-01	1.74E-11	1.31E-14	1.31E-04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{12}	Mean	2.28E+01	2.74E-08	8.02E-02	4.24E-03	**0.00E+00**	8.00E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	4.68E+00	2.07E-08	2.77E-01	2.67E-03	**0.00E+00**	2.77E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{13}	Mean	2.96E-04	7.88E-04	1.47E-16	1.23E-09	**0.00E+00**	4.44E-18	3.67E-14	**0.00E+00**	**0.00E+00**
	Std.	1.48E-03	2.82E-03	2.60E-16	2.48E-09	**0.00E+00**	2.22E-17	1.84E-13	**0.00E+00**	**0.00E+00**
f_{14}	Mean	4.74E+00	4.75E+00	6.18E+01	9.48E+00	7.19E+02	1.41E-11	4.66E-12	**3.78E-12**	2.21E+03
	Std.	2.37E+01	2.32E+01	1.69E+02	4.74E+01	2.08E+03	3.53E-12	2.47E-12	**7.28E-13**	3.09E+03
f_{15}	Mean	2.81E-15	6.22E-15	1.43E+01	7.32E-06	**5.89E-16**	2.65E-14	1.70E-14	7.50E-15	**5.89E-16**
	Std.	7.11E-16	0.00E+00	6.50E+00	4.28E-06	**0.00E+00**	3.48E-15	5.75E-33	2.49E-15	**0.00E+00**
f_{16}	Mean	**9.42E-33**	2.49E-03	1.91E-31	1.20E-13	3.25E-31	**9.42E-33**	1.07E-32	1.22E-13	3.96E-06
	Std.	**1.40E-48**	1.24E-02	5.28E-31	5.37E-14	4.92E-30	**1.40E-48**	5.74E-33	6.09E-13	1.46E-05
f_{17}	Mean	**1.55E-33**	1.01E-04	5.02E-32	1.87E-13	8.89E-31	1.50E-33	1.80E-33	2.50E-15	7.76E-05
	Std.	**2.47E-34**	8.05E-05	5.95E-32	6.87E-14	2.14E-29	0.00E+00	1.08E-33	1.25E-14	5.96E-04
f_{18}	Mean	4.39E-03	1.01E-04	2.95E-05	1.13E-03	**0.00E+00**	5.32E-11	2.14E-16	2.07E-16	9.19E-05
	Std.	6.44E-03	8.05E-05	3.09E-05	2.87E-04	**0.00E+00**	9.68E-11	7.43E-16	7.79E-16	1.73E-03
f_{19}	Mean	**1.35E-31**	**1.35E-31**	1.30E-30	1.84E-14	1.76E-02	**1.35E-31**	**1.35E-31**	**1.35E-31**	4.86E-03
	Std.	**2.47E-33**	**2.23E-47**	3.69E-30	6.58E-15	2.01E-01	**2.23E-47**	**2.23E-47**	**2.23E-47**	8.07E-03
f_{20}	Mean	3.45E+00	3.33E-01	**0.00E+00**	2.79E-01	**0.00E+00**	2.56E-06	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	2.94E-01	4.45E-02	**0.00E+00**	4.51E-02	**0.00E+00**	7.39E-06	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{21}	Mean	**−7.83E+01**	**−7.83E+01**	**−7.83E+01**	**−7.83E+01**	−7.07E+01	**−7.83E+01**	**−7.83E+01**	**−7.83E+01**	−7.70E+01
	Std.	**4.10E-15**	**2.26E-01**	**1.12E-13**	**7.67E-15**	8.04E+00	**4.10E-15**	**1.00E-14**	**1.16E-14**	2.27E+01
f_{22}	Mean	−4.86E+01	−4.98E+01	**−5.00E+01**	−4.97E+01	−4.12E+01	**−5.00E+01**	**−5.00E+01**	**−5.00E+01**	−2.03E+01
	Std.	4.45E-01	3.86E-02	**9.23E-03**	7.12E-02	1.03E+01	**9.13E-04**	**3.82E-07**	**6.79E-06**	9.93E+00
p		0.370	0.218	0.841	0.411	0.878	1.000	0.687	0.605	
$w/t/l$		14/2/6	16/2/4	12/3/7	14/2/6	5/12/5	12/3/7	10/5/7	9/6/7	

The best optimization results are in bold. The p values are obtained by the Wilcoxon test for DRFA and other algorithms. $w/t/l$ indicates that compared with the algorithm, the performance of DRFA is better on w functions, equivalent on t functions, and worse on l functions

As shown in Table 8, DRFA has the best performance on 15 functions, followed by HCOPSO which has the best performance on 13 test functions, and then BABC which has the best performance on 10 functions. From this point of view, DRFA has the best performance among all the algorithms. In specific analysis, compared with the improved differential evolution algorithms CoDE, JADE, and jDEscop, the optimization performance of DRFA is superior on 14, 16, and 12 functions, and inferior on 6, 4, and 7 functions respectively, and its performance is worse on functions f_{14}, f_{16}, f_{17}, f_{19}, f_{21}, and f_{22}. Compared with the improved PSO algorithms CLPSO and HCOPSO, the performance of DRFA is superior on 14 and 5 functions, and is inferior on 6 and 5 functions respectively. Compared with HCOPSO, DRFA has the same number of advantages and disadvantages in terms of performance. Compared with the improved ABC algorithms AABCLS, ABCVSS, and BABC, the performance of DRFA is superior on 12, 10, and 9 functions respectively, and inferior on seven functions, and the seven functions in the inferior position are the same, namely f_{14}, f_{16}, f_{17}, f_{18}, f_{19}, f_{21}, and f_{22}. The results show that the ABC algorithm is more suitable for dealing with these problems. In general, DRFA has higher performance on functions f_1–f_{13}, f_{15}, and f_{20}, but the capacity is not enough on functions f_{14}, f_{16}, f_{17}, f_{18}, f_{19}, f_{21}, and f_{22}; in particular, its worst performance occurs on function f_{14}. The ability of DRFA to deal with such problems needs to be improved. According to the results of the Wilcoxon test, DRFA does not achieve a significant performance improvement compared with the other algorithms.

As can be seen from the Friedman test results in Table 8, DRFA has the best performance on the single-mode function, followed by HCOPSO. In terms of the multi-mode test function, the improved ABC algorithm has more advantages, followed by HCOPSO and DRFA, which indicates that the ABC algorithm has strong global exploration ability but weak local development ability. Among all the test functions, DRFA leads to the best overall optimization. As can be seen from Table 8, the performance of DRFA is similar to that of HCOPSO. As can be seen from Table 9, DRFA has stronger development ability, while HCOPSO has stronger exploration ability. However, the overall optimization performance of DRFA is better on the 22 test functions than that of HCOPSO.

Table 9 Friedman test results of 9 algorithms on 22 test functions

Algorithm	Rank		
	Single-mode f_1–f_9	Multi-mode f_{10}–f_{22}	All f_1–f_{22}
CoDE	4.44 (4)	5.85 (7)	5.30 (8)
JADE	3.69 (3)	6.12 (8)	5.09 (7)
jDEscop	4.69 (5)	5.38 (6)	5.07 (6)
CLPSO	7.44 (9)	6.62 (9)	6.89 (9)
HCOPSO	2.88 (2)	5.00 (4)	4.09 (2)
AABCLS	5.94 (6)	3.96 (3)	4.82 (3)
ABCVSS	6.56 (7)	3.42 (2)	4.82 (3)
BABC	7.06 (8)	**3.35 (1)**	4.91 (5)
DRFA	**2.31 (1)**	5.31 (5)	**4.02 (1)**

The best results are in bold

5.2.2 Comparison on CEC 2015 test functions

The first and second groups of test functions are the earlier version of test functions, and most of the improved algorithms have good optimization results. However, CEC 2015 test functions are a group of newly proposed test functions, which are more difficult to optimize (Chen et al., 2015). They contain 15 functions divided into four groups: single-mode functions f_1 and f_2, multi-mode functions f_3–f_5, mixed functions f_6–f_8, and composite functions f_9–f_{15}.

To further verify the optimization performance of DRFA, in this subsection CEC 2015 test functions are used to compare DRFA and FA (Yang, 2008), PSO (Kennedy and Eberhart, 1995), and the new hybrid algorithms FA and PSO, i.e., HPSOFF (Arunachalam et al., 2014), FFPSO (Kora and Krishna, 2016), and HFPSO (Aydilek, 2018).

The dimension of the optimization problem is set at D=30, and the maximum number of evaluation times MAX_FEs=1500. Since the number of evaluations is smaller, the step attenuation period of DRFA is set at c=1. Other parameters of DRFA are the same as those in the section above. For parameter settings of each algorithm, please refer to Aydilek (2018). The algorithm runs independently on each function 30 times, and its mean and standard deviation are shown in Table 10. Because the number of each class of functions in this group is small, there is no grouping test when the Friedman test is carried out.

As shown in Table 10, DRFA has the highest optimization accuracy on 11 functions, followed by HFPSO which has the best performance on 4 functions. Specifically, DRFA outperforms FA and FFPSO on 15 functions. Compared with PSO, DRFA wins on 12 functions and has poor performance on f_1, f_8, and f_{10}. Compared with HPSOFF, DRFA is superior on 13 functions, and inferior on f_8 and f_{10}. DRFA is better than HFPSO on 11 functions and worse on f_1, f_4, f_8, and f_{10}. From the Wilcoxon test results, DRFA is significantly better than FA and FFPSO. Compared with PSO, HPSOFF, and HFPSO, DRFA does not achieve significant performance improvement. The results of the Friedman test show that DRFA performs best among the six algorithms.

6 Optimal scheduling of cascade reservoirs

A series of stepped reservoirs are built from upstream to downstream of a river or reach; i.e., they are "cascade" reservoirs. The optimal scheduling of cascade reservoirs is characterized by being multi-dimensional, high-order, non-convex, and non-linear. At the same time, there are complicated hydrological, hydraulic, and electric power connections among cascade reservoirs. This makes the problem more difficult to solve. It is a typical complex optimal scheduling problem with many resources, various scheduling objectives, and coupling constraints.

Here, the water level of cascade reservoirs at the end of the month is taken as the scheduling object, the

Table 10 Optimization results of six algorithms on CEC 2015 test functions

Function	Type	PSO	FA	FFPSO	HPSOFF	HFPSO	DRFA
f_1	Mean	3.90E+09	2.89E+10	9.33E+10	4.75E+09	**1.18E+09**	4.22E+09
	Std.	1.60E+09	5.85E+09	1.28E+10	1.34E+09	**6.24E+08**	1.65E+09
f_2	Mean	9.98E+04	1.34E+05	6.94E+06	9.74E+04	8.57E+04	**8.39E+04**
	Std.	2.58E+04	2.63E+04	1.45E+07	2.08E+04	2.09E+04	**1.71E+04**
f_3	Mean	3.31E+02	3.39E+02	3.48E+02	3.31E+02	3.26E+02	**3.47E+01**
	Std.	2.83E+00	1.96E+00	1.92E+00	2.62E+00	4.14E+00	**3.25E+00**
f_4	Mean	7.79E+03	8.03E+03	9.67E+03	6.82E+03	**5.12E+03**	5.31E+03
	Std.	5.24E+02	4.26E+02	4.03E+02	8.57E+02	**6.79E+02**	6.76E+02
f_5	Mean	5.04E+02	5.04E+02	5.06E+02	5.04E+02	5.04E+02	**3.99E+00**
	Std.	7.22E-01	5.19E-01	1.20E+00	6.16E-01	8.13E-01	**5.92E-01**
f_6	Mean	6.01E+02	6.04E+02	6.08E+02	6.01E+02	6.01E+02	**7.63E-01**
	Std.	1.88E-01	2.29E-01	6.70E-01	1.31E-01	9.53E-02	**1.09E-01**
f_7	Mean	7.04E+02	7.70E+02	8.94E+02	7.08E+02	7.01E+02	**4.07E+00**
	Std.	2.01E+00	1.22E+01	2.69E+01	4.27E+00	3.33E-01	**2.94E+00**
f_8	Mean	4.00E+03	2.35E+06	1.60E+08	1.72E+04	**2.64E+03**	4.41E+04
	Std.	3.35E+03	1.45E+06	1.16E+08	3.33E+04	**4.49E+03**	5.00E+04
f_9	Mean	9.14E+02	9.14E+02	9.14E+02	9.14E+02	9.13E+02	**1.33E+01**
	Std.	2.34E-01	2.81E-01	1.85E-01	3.17E-01	2.27E-01	**3.19E-01**
f_{10}	Mean	7.56E+06	2.99E+07	3.94E+08	1.13E+07	**5.47E+06**	2.09E+07
	Std.	3.12E+06	1.45E+07	1.68E+08	6.35E+06	**3.35E+06**	1.05E+07
f_{11}	Mean	1.16E+03	1.29E+03	2.11E+03	1.16E+03	1.13E+03	**9.75E+01**
	Std.	4.10E+01	3.76E+01	4.66E+02	2.85E+01	2.24E+01	**3.64E+01**
f_{12}	Mean	2.24E+03	2.97E+03	4.99E+05	2.06E+03	1.78E+03	**8.28E+02**
	Std.	1.86E+02	2.80E+02	6.16E+05	2.07E+02	1.45E+02	**3.06E+02**
f_{13}	Mean	1.77E+03	1.96E+03	3.63E+03	1.74E+03	1.69E+03	**4.71E+02**
	Std.	2.65E+01	8.11E+01	7.74E+02	2.34E+01	1.65E+01	**4.65E+01**
f_{14}	Mean	1.66E+03	1.75E+03	2.17E+03	1.67E+03	1.65E+03	**3.12E+02**
	Std.	1.86E+01	2.61E+01	1.46E+02	2.85E+01	9.74E+00	**2.91E+01**
f_{15}	Mean	2.55E+03	2.83E+03	3.90E+03	2.65E+03	2.45E+03	**1.32E+03**
	Std.	1.71E+02	5.55E+01	4.47E+02	1.69E+02	2.10E+02	**1.30E+02**
Rank		3.27 (3)	4.87 (5)	5.90 (6)	3.33 (4)	1.97 (2)	**1.67 (1)**
p		0.307	**0.001**	**0.001**	0.211	0.334	
$w/t/l$		12/0/3	15/0/0	15/0/0	13/0/2	11/0/4	

The best optimization results are in bold. The p values are obtained by the Wilcoxon test for DRFA and other algorithms. The p value less than 0.05 is in bold, indicating that DRFA is clearly superior to the compared algorithm. $w/t/l$ indicates that compared with the algorithm, the performance of DRFA is better on w functions, equivalent on t functions, and worse on l functions

maximum total power generation of cascade reservoirs is taken as the scheduling objective, and FA is selected as the scheduling algorithm. The optimal scheduling model of cascade reservoirs based on FA is established and compared with many improved FAs to verify the performance of DRFA in solving the optimal scheduling problem of cascade reservoirs.

6.1 Objective function

In the optimal scheduling model of cascade reservoirs based on FA, the firefly position is determined by the water level of each reservoir at the end of each month, and the firefly brightness is determined by the total power generation of each reservoir. The greater the total power generation is, the brighter the firefly is. The total power generation of each reservoir is shown in Eq. (9):

$$\begin{aligned}\max E &= \sum_{t=1}^{T}\sum_{i=1}^{N} N_{i,t}\Delta t \\ &= \sum_{t=1}^{T}\sum_{i=1}^{N} \eta_i Q_{i,t} H_{i,t}\Delta t,\end{aligned} \tag{9}$$

where t=1, 2, …, T is the number of time periods, i=1, 2, …, N represents the number of power stations, $N_{i,t}$ is the output of power station i in period t, Δt is a single period, η_i is the comprehensive output coefficient of power station i, $Q_{i,t}$ represents the average discharge flow, $H_{i,t}$ is the generation head, and E is the total power generation of cascade power stations.

Eq. (9) needs to satisfy four constraints listed in Eq. (10). These are the constraints of water balance, reservoir water level, output, and discharge flow.

$$\begin{cases} V_{i,t} = V_{i,t-1} + (I_{i,t} - Q_{i,t})\Delta t, \\ Z_{\min_{i,t}} \le Z_{i,t} \le Z_{\max_{i,t}}, \\ N_{\min_{i,t}} \le N_{i,t} \le N_{\max_{i,t}}, \\ Q_{\min_{i,t}} \le Q_{i,t} \le Q_{\max_{i,t}}, \end{cases} \tag{10}$$

where $I_{i,t}$ represents the average water inflow of reservoir i in time period t, and $V_{i,t}$ represents the storage capacity of reservoir i at the end of period t. $Z_{\min_{i,t}}$ and $Z_{\max_{i,t}}$ are the minimum and maximum limited water levels of the reservoir, respectively. Usually, $Z_{\min_{i,t}}$ = dead water level and $Z_{\max_{i,t}}$ = normal water level. $N_{\max_{i,t}}$ and $N_{\min_{i,t}}$ are the installed capacity and guaranteed output respectively. $Q_{\min_{i,t}}$ and $Q_{\max_{i,t}}$ are the minimum and maximum discharge flow of the reservoir, respectively. All variables satisfy the nonnegative constraints.

6.2 Constraint processing and solving steps

Of the four constraints mentioned above, the water balance constraint is essentially the water yield connection of each reservoir. This plays the role of connecting the variables in the function, and does not have to be dealt with. In the reservoir water level constraint, since the firefly position is determined by the water level, $Z_{\min_{i,t}}$ and $Z_{\max_{i,t}}$ represent the boundary of the firefly search space. In FA, there is a corresponding mechanism for handling the boundary crossing. Hence, the reservoir water level constraint does not have to be processed additionally. For the output constraint, the output is related to the power of the power station, and the output processing will directly affect the fitness function. The flow constraint shall consider not only the overflow capacity of the unit, but also the downstream irrigation and shipping demand. In view of the above problem, a penalty function is adopted to deal with the output constraint and discharge flow constraint. See Eq. (11):

$$L_{i,t} = \begin{cases} 0, & N_{\min_{i,t}} \le N^*_{i,t} \le N_{\max_{i,t}}, \\ 1, & N^*_{i,t} < N_{\min_{i,t}} \text{ or } N^*_{i,t} > N_{\max_{i,t}}, \end{cases} \tag{11}$$

where $L_{i,t}$ records the default situation of the output of reservoir i at time period t, and $N^*_{i,t}$ is the output value. In the same way, the penalty function is used to deal with the discharge constraint. The total power after the penalty function is added:

$$E = E^* - u_N \sum_{i=1}^{N}\sum_{t=1}^{T} L_{i,t}, \tag{12}$$

where E^* is the calculated value of the total power, and u_N is the penalty coefficient. As shown in this formula, the intensity of punishment is directly proportional to the number of defaults, i.e., the more the number of defaults, the greater the punishment.

The steps of solving the optimal scheduling model of cascade reservoirs based on FA are:

Step 1: Initialization. The number of fireflies is N. Within the water level constraint range of each reservoir period, the random water level is generated to represent the firefly position, where the dimension D=number of reservoirs×number of time periods.

Step 2: Calculate the power generation of cascade reservoirs at the current water level, that is, the fitness value of the firefly particles.

Step 3: Through the mutual attraction movement between fireflies, update the position of each firefly particle and recalculate the fitness value.

Step 4: Judge whether the end condition of the algorithm is reached. If so, the optimal solution will be outputted; otherwise, return to step 3.

6.3 Case study

The Qingjiang cascade reservoirs are taken as the scheduling object, and the annual scheduling period is used for monthly scheduling. The characteristic parameters of each reservoir are shown in Table 11. The scheduling period is from the beginning of January to the end of December. The starting and ending water levels of each reservoir are 370 m in Shuibuya, 170 m in Geheyan, and 79 m in Gaobazhou. June and July are the flood seasons, and the maximum limited water level comes in the flood season.

Due to the poor performance of WSSFA, VSSFA, MFA, and RaFA, we compare only DRFA with the classical FA and ApFA, NSRaFA, DLFA, and LVFA with higher performance. The number of fireflies is set at 20, the dimension as D=36, and the maximum number of evaluation times as 100 000. The results of the algorithm are shown in Table 12.

As shown in Table 12, DRFA has the highest power generation in Shuibuya and Geheyan hydropower stations. In the Gaobazhou hydropower station, the power generation of DRFA is lower than that of NSRaFA, DLFA, and ApFA, where DLFA has the highest power generation. In terms of total power generation, the total power generated by the algorithm in this study is the highest. Fig. 6 shows the convergence curves of all algorithms. When the evaluation number is 10 000, the convergence speed of each algorithm is approximately the same except for FA. However, the convergence of DRFA and LVFA is significantly faster than that of other algorithms between 10 000 and 20 000 times, among which DRFA has the fastest convergence. After 20 000 times, the

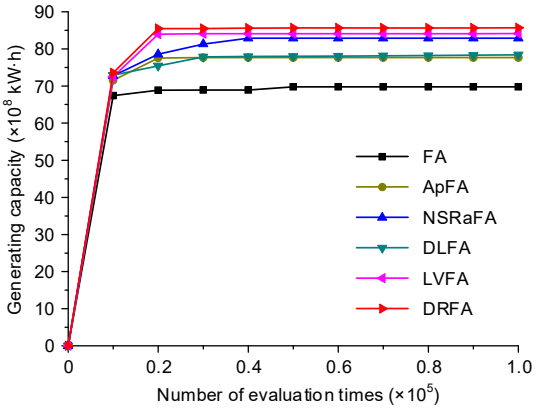

Fig. 6 Convergence diagram of power generation and the number of evaluation times of six algorithms

Table 11 Characteristic parameters of each hydropower station in Qingjiang cascade reservoirs

Reservoir	Normal water level (m)	Flood water level (m)	Dead water level (m)	Installed capacity (MW)	Firm output (MW)	Maximum head (m)	Minimum head (m)	Comprehensive output coefficient
Shuibuya	400.0	391.8	350.0	1600.0	310.0	203.0	147.0	8.5
Geheyan	200.0	193.6	160.0	1200.0	241.5	121.6	80.7	8.5
Gaobazhou	80.0	78.5	78.0	270.0	77.3	40.0	22.3	8.4

Table 12 Maximum power generation after optimal scheduling of six algorithms

Algorithm	Maximum power generating ability (×10^8 kW·h)			
	Shuibuya	Geheyan	Gaobazhou	Total
FA	33.552 02	26.359 89	9.849 58	69.761 49
ApFA	37.453 11	29.624 85	10.577 57	77.655 53
NSRaFA	38.984 70	32.549 24	11.321 09	82.855 03
DLFA	38.038 87	29.005 23	**11.396 26**	78.440 36
LVFA	41.095 58	32.551 07	10.468 05	84.114 70
DRFA	**41.215 95**	**33.494 23**	10.499 12	**85.209 30**

optimization speed of each algorithm is clearly slower. DRFA, ApFA, and LVFA basically stop evolving. DRFA can achieve higher power generation with a smaller number of evaluation times.

Table 13 shows the surplus water amount of each algorithm. Specifically, surplus water refers to the amount of water available for power generation under the power generation capacity of hydropower stations, but not actually used for power generation for various reasons, so the lower the better. The smaller the surplus water amount is, the higher the utilization of water resources is during the scheduling of cascade reservoirs. As shown in Table 13, in the Shuibuya hydropower station and the Geheyan hydropower station, the surplus water amount is 0. In the Gaobazhou hydropower station, the surplus water amount of FA is the largest, followed by NSRaFA and DLFA. The surplus water amount of ApFA, LVFA, and DRFA is 0 in each case. By comprehensive comparison, this algorithm is characterized by the largest power generation capacity and the smallest surplus water amount. Hence, when the cascade reservoir scheduling problem is solved, the performance of DRFA is the best among the six algorithms.

Table 13 Surplus water amount after optimal scheduling of six algorithms

Algorithm	Surplus water amount (m^3/s)			
	Shuibuya	Geheyan	Gaobazhou	Average
FA	0	0	206.049	68.683
ApFA	0	0	0	0
NSRaFA	0	0	72.986	24.329
DLFA	0	0	16.730	5.577
LVFA	0	0	0	0
DRFA	0	0	0	0

7 Conclusions

1. Taking the scheduling of cascade reservoirs as an example, this paper determines that the current optimal operation problem is a complex problem featuring multi-objective, multi-constraint, multi-stage, and strong coupling.

2. Given that FA cannot solve the complex optimal scheduling problem, DRFA is proposed by integrating various learning strategies into FA with the idea of division of roles, which makes up for the deficiency of FA using a single learning strategy. DRFA takes full account of the optimization characteristics of the complex optimal scheduling problem, and divides the fireflies into leaders, developers, and followers, where the leader uses the greedy Cauchy mutation strategy to save better scheduling schemes and find new ones; the developer conducts fine search around the leader to adjust for a better scheduling scheme to further improve the scheduling effect; the follower integrates the information of the two scheduling schemes to form a new one, so as to weaken the impact of the single scheduling scheme on the scheduling.

3. Experimental results show that DRFA can effectively improve the adaptability of an algorithm to complex optimization problems. In the next step, more targeted learning strategies will be developed and applied to different roles of fireflies. At the same time, DRFA will be applied to other complex optimal scheduling problems to expand its application scope.

Contributors

Renbin XIAO designed the research. Jia ZHAO and Jun YE processed the data. Wenping CHEN drafted the manuscript. Renbin XIAO helped organize the manuscript. Wenping CHEN and Jia ZHAO revised and finalized the paper.

Compliance with ethics guidelines

Jia ZHAO, Wenping CHEN, Renbin XIAO, and Jun YE declare that they have no conflict of interest.

References

Alomoush W, Omar K, Alrosan A, et al., 2020. Firefly photinus search algorithm. *J King Univ-Comput Inform Sci*, 32(5):599-607. https://doi.org/10.1016/j.jksuci.2018.06.010

Arunachalam S, AgnesBhomila T, Ramesh Babu M, 2014. Hybrid particle swarm optimization algorithm and firefly algorithm based combined economic and emission dispatch including valve point effect. Proc 5th Int Conf on Swarm, Evolutionary, and Memetic Computing, p.647-660. https://doi.org/10.1007/978-3-319-20294-5_56

Aydilek IB, 2018. A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. *Appl Soft Comput*, 66:232-249. https://doi.org/10.1016/j.asoc.2018.02.025

Brest J, Maučec MS, 2011. Self-adaptive differential evolution algorithm using population size reduction and three strategies. *Soft Comput*, 15(11):2157-2174. https://doi.org/10.1007/s00500-010-0644-5

Chen Q, Liu B, Zhang Q, et al., 2015. Problem definitions and evaluation criteria for CEC 2015 Special Session on

Bound Constrained Single-Objective Computationally Expensive Numerical Optimization. Proc IEEE Congress on Evolutionary Computation, p.84-88.

Cook SA, 1971. The complexity of theorem-proving procedures. Proc 3rd Annual ACM Symp on Theory of Computing, p.151-158. https://doi.org/10.1145/800157.805047

Cui ZH, Cao Y, Cai XJ, et al., 2019. Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things. *J Parall Distrib Comput*, 132:217-229. https://doi.org/10.1016/j.jpdc.2017.12.014

Fan TH, Yao ZF, Han LZ, et al., 2021. Density peaks clustering based on k-nearest neighbors sharing. *Concurr Comput Pract Exp*, 33(5):e5993. https://doi.org/10.1002/cpe.5993

Fister I, Fister IJr, Yang XS, et al., 2013. A comprehensive review of firefly algorithms. *Swarm Evol Comput*, 13:34-46. https://doi.org/10.1016/j.swevo.2013.06.001

Fister IJr, Yang XS, Fister I, et al., 2012. Memetic firefly algorithm for combinatorial optimization. *Mathematics*, 2012:75-86.

Gao WF, Chan FTS, Huang LL, et al., 2015. Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood. *Inform Sci*, 316:180-200. https://doi.org/10.1016/j.ins.2015.04.006

García S, Molina D, Lozano M, et al., 2009. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *J Heurist*, 15(6):617-644. https://doi.org/10.1007/s10732-008-9080-4

Gope S, Goswami AK, Tiwari PK, et al., 2016. Rescheduling of real power for congestion management with integration of pumped storage hydro unit using firefly algorithm. *Int J Electr Power Energy Syst*, 83:434-442. https://doi.org/10.1016/j.ijepes.2016.04.048

Guo BY, Zhuang ZJ, Pan JS, et al., 2021. Optimal design and simulation for PID controller using fractional-order fish migration optimization algorithm. *IEEE Access*, 9:8808-8819. https://doi.org/10.1109/ACCESS.2021.3049421

Jadon SS, Bansal JC, Tiwari R, et al., 2015. Accelerating artificial bee colony algorithm with adaptive local search. *Memet Comput*, 7(3):215-230. https://doi.org/10.1007/s12293-015-0158-x

Kassandra T, Rojali, Suhartono D, 2018. Resource-constrained project scheduling problem using firefly algorithm. *Proc Comput Sci*, 135:534-543. https://doi.org/10.1016/j.procs.2018.08.206

Kennedy J, Eberhart R, 1995. Particle swarm optimization. Proc Int Conf on Neural Networks, p.1942-1948. https://doi.org/10.1109/ICNN.1995.488968

Kiran MS, Hakli H, Gunduz M, et al., 2015. Artificial bee colony algorithm with variable search strategy for continuous optimization. *Inform Sci*, 300:140-157. https://doi.org/10.1016/j.ins.2014.12.043

Kora P, Krishna KSR, 2016. Hybrid firefly and particle swarm optimization algorithm for the detection of bundle branch block. *Int J Cardiov Acad*, 2(1):44-48. https://doi.org/10.1016/j.ijcac.2015.12.001

Liang JJ, Qin AK, Suganthan PN, et al., 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans Evol Comput*, 10(3):281-295. https://doi.org/10.1109/TEVC.2005.857610

Lloyd JE, 1971. Bioluminescent communication in insects. *Ann Rev Entomol*, 16:97-122. https://doi.org/10.1146/annurev.en.16.010171.000525

Lv L, Zhao J, 2018. The firefly algorithm with Gaussian disturbance and local search. *J Signal Process Syst*, 90(8-9):1123-1131. https://doi.org/10.1007/s11265-017-1278-y

Lv L, Zhao J, Wang JY, et al., 2019. Multi-objective firefly algorithm based on compensation factor and elite learning. *Fut Gener Comput Syst*, 91:37-47. https://doi.org/10.1016/j.future.2018.07.047

Lv L, Wang JY, Wu RX, et al., 2021. Density peaks clustering based on geodetic distance and dynamic neighbourhood. *Int J Bio-Inspir Comput*, 17(1):24-33. https://doi.org/10.1504/IJBIC.2021.113363

Meng ZY, Pan JS, Xu HR, 2016. QUasi-Affine TRansformation Evolutionary (QUATRE) algorithm: a cooperative swarm based algorithm for global optimization. *Knowl-Based Syst*, 109:104-121. https://doi.org/10.1016/j.knosys.2016.06.029

Moeini R, Babaei M, 2020. Hybrid SVM-CIPSO methods for optimal operation of reservoir considering unknown future condition. *Appl Soft Comput*, 95:106572. https://doi.org/10.1016/j.asoc.2020.106572

Ohba N, 2004. Flash communication systems of Japanese fireflies. *Integr Comp Biol*, 44(3):225-233. https://doi.org/10.1093/icb/44.3.225

Pan JS, Liu NX, Chu SC, et al., 2020. An efficient surrogate-assisted hybrid optimization algorithm for expensive optimization problems. *Inform Sci*, 561:304-325. https://doi.org/10.1016/j.ins.2020.11.056

Pan JS, Sun XX, Chu SC, et al., 2021. Digital watermarking with improved SMS applied for QR code. *Eng Appl Artif Intell*, 97:104049. https://doi.org/10.1016/j.engappai.2020.104049

Ritthipakdee A, Thammano A, Premasathian N, et al., 2017. Firefly mating algorithm for continuous optimization problems. *Comput Intell Neurosci*, 2017:8034573. https://doi.org/10.1155/2017/8034573

Song PC, Chu SC, Pan JS, et al., 2020. Phasmatodea population evolution algorithm and its application in length-changeable incremental extreme learning machine. Proc 2nd Int Conf on Industrial Artificial Intelligence, p.1-5. https://doi.org/10.1109/IAI50351.2020.9262236

Sun H, Deng ZC, Zhao J, et al., 2019. Hybrid mean center opposition-based learning particle swarm optimization. *Acta Electron Sin*, 47(9):1809-1818 (in Chinese). https://doi.org/10.3969/j.issn.0372-2112.2019.09.001

Takeuchi M, Matsushita H, Uwate Y, et al., 2015. Firefly algorithm distinguishing between males and females for

minimum optimization problems. Proc IEEE Workshop on Nonlinear Circuit Networks, p.50-51.

Tian AQ, Chu SC, Pan JS, et al., 2020. A compact pigeon-inspired optimization for maximum short-term generation mode in cascade hydroelectric power station. *Sustainability*, 12(3):767. https://doi.org/10.3390/su12030767

Wang CF, Song WX, 2019. A novel firefly algorithm based on gender difference and its convergence. *Appl Soft Comput*, 80:107-124. https://doi.org/10.1016/j.asoc.2019.03.010

Wang GG, Cai XJ, Cui ZH, et al., 2020. High performance computing for cyber physical social systems by using evolutionary multi-objective optimization algorithm. *IEEE Trans Emerg Top Comput*, 8(1):20-30. https://doi.org/10.1109/TETC.2017.2703784

Wang H, Sun H, Li CH, et al., 2013. Diversity enhanced particle swarm optimization with neighborhood search. *Inform Sci*, 223:119-135. https://doi.org/10.1016/j.ins.2012.10.012

Wang H, Wang WJ, Sun H, et al., 2016. Firefly algorithm with random attraction. *Int J Bio-Inspir Comput*, 8(1):33-41. https://doi.org/10.1504/IJBIC.2016.074630

Wang H, Zhou XY, Sun H, et al., 2017a. Firefly algorithm with adaptive control parameters. *Soft Comput*, 21(17):5091-5102. https://doi.org/10.1007/s00500-016-2104-3

Wang H, Cui ZH, Sun H, et al., 2017b. Randomly attracted firefly algorithm with neighborhood search and dynamic parameter adjustment mechanism. *Soft Comput*, 21(18):5325-5339. https://doi.org/10.1007/s00500-016-2116-z

Wang Y, Cai ZX, Zhang QF, 2011. Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Trans Evol Comput*, 15(1):55-66. https://doi.org/10.1109/TEVC.2010.2087271

Wu HS, Xue JJ, Xiao RB, et al., 2020. Uncertain bilevel knapsack problem based on an improved binary wolf pack algorithm. *Front Inform Technol Electron Eng*, 21(9):1356-1368. https://doi.org/10.1631/FITEE.1900437

Wu N, 2020. Research on Scheduling Optimization Models and Corresponding Algorithms for Container Terminal under Abnormal Working Conditions. PhD Thesis, Dalian Maritime University, Dalian, China (in Chinese).

Xiao RB, Wang YC, 2018. Labour division in swarm intelligence for allocation problems: a survey. *Int J Bio-Inspir Comput*, 12(2):71-86. https://doi.org/10.1504/IJBIC.2018.094186

Xiao RB, Zhang YF, Huang ZD, 2015. Emergent computation of complex systems: a comprehensive review. *Int J Bio-Inspir Comput*, 7(2):75-97. https://doi.org/10.1504/IJBIC.2015.069292

Xu JG, Dai GZ, Wang HA, 2004. An overview of theories and methods of production scheduling. *J Comput Res Dev*, 41(2):257-267 (in Chinese).

Yang XS, 2008. Nature-Inspired Metaheuristic Algorithms. Luniver Press, Frome, UK.

Yang XS, 2010. Engineering Optimization: an Introduction with Metaheuristic Applications. John Wiley & Sons, Hoboken, US.

Yu BH, Wang JW, Li CL, et al., 2004. DP with successive approximation for solving hydropower unit commitment problem. *Centr China Electr Power*, 17(6):1-3 (in Chinese). https://doi.org/10.3969/j.issn.1006-6519.2004.06.001

Yu SH, Su SB, Lu QP, et al., 2014. A novel wise step strategy for firefly algorithm. *Int J Comput Math*, 91(12):2507-2513. https://doi.org/10.1080/00207160.2014.907405

Yu SH, Zhu SL, Ma Y, et al., 2015. A variable step size firefly algorithm for numerical optimization. *Appl Math Comput*, 263:214-220. https://doi.org/10.1016/j.amc.2015.04.065

Zhang HW, Xie JW, Lu WL, et al., 2017. A scheduling method based on a hybrid genetic particle swarm algorithm for multifunction phased array radar. *Front Inform Technol Electron Eng*, 18(11):1806-1816. https://doi.org/10.1631/FITEE.1601358

Zhang JQ, Sanderson AC, 2009. JADE: adaptive differential evolution with optional external archive. *IEEE Trans Evol Comput*, 13(5):945-958. https://doi.org/10.1109/TEVC.2009.2014613

Zhang MQ, Wang H, Cui ZH, et al., 2018. Hybrid multi-objective cuckoo search with dynamical local search. *Memet Comput*, 10(2):199-208. https://doi.org/10.1007/s12293-017-0237-2

Zhao J, Fan TH, Lü L, et al., 2017a. Adaptive intelligent single particle optimizer based image de-noising in shearlet domain. *Intell Autom Soft Comput*, 23(4):661-666. https://doi.org/10.1080/10798587.2017.1316069

Zhao J, Lv L, Wang H, et al., 2017b. Particle swarm optimization based on vector Gaussian learning. *KSII Trans Intern Inform Syst*, 11(4):2038-2057. https://doi.org/10.3837/TIIS.2017.04.012

Zhao J, Xie ZF, Lü L, et al., 2018. Firefly algorithm with deep learning. *Acta Electron Sin*, 46(11):2633-2641 (in Chinese). https://doi.org/10.3969/j.issn.0372-2112.2018.11.010

Zhao J, Chen WP, Ye J, et al., 2020. Firefly algorithm based on level-based attracting and variable step size. *IEEE Access*, 8:58700-58716. https://doi.org/10.1109/ACCESS.2020.2981656

Zhao J, Yao ZF, Lü L, et al., 2021. Density peaks clustering based on mutual neighbor degree. *Contr Dec*, 36(3):543-552. https://doi.org/10.13195/j.kzyjc.2019.0795

Zhou XY, Wang H, Wang MW, et al., 2017. Enhancing the modified artificial bee colony algorithm with neighborhood search. *Soft Comput*, 21(10):2733-2743. https://doi.org/10.1007/s00500-015-1977-x

Zou DX, Wang GG, Pan G, et al., 2016. A modified simulated annealing algorithm and an excessive area model for floorplanning using fixed-outline constraints. *Front Inform Technol Electron Eng*, 17(11):1228-1244. https://doi.org/10.1631/FITEE.1500386

Jia ZHAO, first author of this invited paper, received his ME degree in computer application technology from Nanchang Hangkong University, Nanchang, China, in 2011, and PhD degree in information and communication engineering from Hohai University, Nanjing, China, in 2020. He is currently a full professor with the School of Information Engineering, Nanchang Institute of Technology, Nanchang, China. He is Director of the Nanchang Key Laboratory of Big Data and Computational Intelligence. His research interests include big data analysis and artificial intelligence theory.

Renbin XIAO, corresponding author of this invited paper, received his PhD degree in systems engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1993. He is currently a full professor with the School of Artificial Intelligence and Automation, HUST. He has coauthored more than 10 books and published over 300 academic papers. He received 10 projects from the National Natural Science Foundation of China and won five science and technology awards from the Ministry of Education and Hubei Province, China. His research interests include swarm intelligence, emergent computation, intelligent manufacturing, and innovation design of complex products.