2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

Improve Enhanced Fireworks Algorithm with Differential Mutation

Chao Yu, Junzhi Li, and Ying Tan

Abstract— **Fireworks algorithm (FWA) is a newly proposed swarm intelligence algorithm, which is used to solve optimization problems. However, the interaction of fireworks in FWA is not sufficient. In this paper, the differential mutation operator is introduced to improve the interaction mechanism of enhanced FWA (EFWA), which is the latest version of FWA. Extensive experiments on 30 benchmark functions were conducted to test the performance of the new algorithm named enhanced fireworks algorithm with differential mutation (FWA-DM). Experimental results have shown that differential mutation operator is able to improve EFWA.**

I. Introduction

In the past two decades, swarm intelligence had become a research trend. Many scholars studied swarm intelligence and proposed useful algorithms to solve real world problems. Dorigo, et al. observed the way ants searched for food and presented ant colony optimization (ACO) in 1991 [1]. Kennedy and Eberhart were inspired by flocking birds and put forth a particle swarm optimization (PSO) algorithm in 1995 [2]. In the same year, Storn and Price presented a simple and effective algorithm named differential evolution (DE) to deal with optimization problems [3]. The recently announced swarm intelligence algorithm, artificial bee colony algorithm (ABC) was proposed in 2008 [4]. In 2010, Tan and Zhu perpetuated the fireworks algorithm [5], which simulates groupings of sparks formed by fireworks exploding in the night sky, to find global optimum of functions. Each firework can then affect other fireworks, similar to a group of individuals acting and reacting to their surroundings accordingly. The explosion phenomena can be treated as a strategy for expanding across the adjacent area much like a group of individuals who populate a new location. As more fireworks explode, the algorithm will search the solution space in a defined structure and the optimal solution will gradually be formulated. Conventional fireworks algorithm has been applied to many fields, such as non-positive matrix factorization [6]–[8] and digital filters [9]. However, the conventional fireworks algorithm is not perfect and can be further improved. In 2013, Zheng, et al. put forward an enhanced fireworks algorithm to deal with the shortcomings of conventional fireworks algorithm [10]. Many scholars have studied FWA and made improvements, as in [**?**], [11]–[15]. Of all the published papers which were concerned with improvements of single objective fireworks algorithm, EFWA performed the best.

All the authors are with the Key Laboratory of Machine Perception and Intelligence (Peking University), Ministry of Education, Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, China (e-mail:chaoyu, ljz, ytan@pku.edu.cn). Ying Tan is the corresponding author.

This work was supported by the Natural Science Foundation of China with grant no. 61375119, 61170057 and 60875080.

However, EFWA has its own disadvantage. The interaction mechanism in EFWA is not sufficient whereas the interaction is vital in swarm intelligence algorithms. As a result, the performance of EFWA can be further improved by increasing the individuals' interaction within a population. Hence, differential mutation (DM) operator is introduced to enhance the interaction in EFWA and thus improve the performance of EFWA.

Section II introduces the enhanced fireworks algorithm, while Section III analyzes the interaction mechanism in EFWA. Section IV demonstrates the details of the new algorithm FWA-DM. Section V lists experimental design, experimental results and discussion. Lastly, the conclusions are presented in section VI.

II. Enhanced Fireworks Algorithm

EFWA is a swarm intelligence algorithm and adopts four parts to solve optimization problems. The first part is the explosion operator. Suppose the population of EFWA consists of *N* D-dimensional vectors as individuals, each individual 'explodes' according to explosion operator and generates sparks around it. The second part is the mutation operator. Sparks are generated under the effect of this operator in order to improve the diversity of the population. The third part, namely as the mapping rule, is used to map the sparks that are out of the feasible space into it. By applying this rule, the sparks are pulled back to feasible space. The last part is called as selection strategy. The individuals are selected from the whole population and pass down to next generation.

The details of EFWA are described as follows.

A. Explosion Operator

As the core operator in EFWA, explosion operator provides the basic way of fireworks explosion. Each of the *N* individuals in the population explodes and a shower of sparks surround them. Therefore, the number of sparks and the amplitude for each individual need to determined.

Assume the number of sparks is stated as S_i.

$$S_i = \hat{S} * \frac{Y_{\max} - f(x_i) + \varepsilon}{\sum_{i=1}^{N} (Y_{\max} - f(x_i)) + \varepsilon} \tag{1}$$

In the formula, $\hat{S}$ is set as a constant and Y_{max} means the fitness value of the worst individual among the *N* individuals. $f(x_i)$ represents the fitness for an individual x_i, while the last parameter ε is used to prevent the denominator from becoming zero.

A_i denotes the amplitude of each individual.

978-1-4799-3840-7/14/$31.00 ©2014 IEEE

$$A_i = \hat{A} * \frac{f(x_i) - Y_{\min} + \varepsilon}{\sum_{i=1}^{N} (f(x_i) - Y_{\min}) + \varepsilon} \quad (2)$$

In the formula, $\hat{A}$ is a constant relevant to amplitude, while Y_{min} means the fitness value of the best individual among the *N* individuals. The meaning of $f(x_i)$ and parameter ε are the same as aforementioned.

In the explosion, the amplitude might be too small, which leads to useless explosion since the new generated firework sparks are close and similar. Therefore, a new parameter A_{min} which prevents the amplitudes from being too small is proposed. There are two ways to calculate the parameter A_{min}, as linear and non-linear decrease, respectively. The value of A_{min} decreasing while the generation increasing.

$$(Linear)A_{\min} = A_{init} - (A_{init} - A_{final}) * Iter/MaxEval \quad (3)$$

$$(Non-linear)A_{\min} = A_{init} - (A_{init} - A_{final}) * \sqrt{(2 * MaxEval - Iter) * Iter/MaxEval} \quad (4)$$

In both formulae, A_{init} and A_{final} are constants, representing the initial and final amplitudes of the explosions. Parameter $Iter$ stands for the number of iterations so far and parameter $MaxEval$ is the maximum function evaluation times. From the empirical experiments in EFWA, A_{min} with the way of non-linear decreasing works better on most test functions. As a result, the non-linear decreasing way is used in this paper.

B. Mutation Operator

To keep the diversity of the populations, mutation operator is used in EFWA. Sparks explode around the selected individual and obey the Gaussian distribution.

$$x_i = x_i + (x_B - x_i) * Gaussian(0, 1) \quad (5)$$

Parameter x_i stands for an individual and $Gaussian(0, 1)$ means a normalized distribution with mean value 0 and standard deviation 1.

In both explosion and mutation operators, there is no guarantee that the generated sparks lie in the feasible space. To solve the problem, the mapping rule is introduced.

C. Mapping Rule

In conventional fireworks algorithm, the mapping rule trends to draw a spark that is out of boundary back to original point. Thus, a new mapping rule is proposed in EFWA. Once a spark flies out of the boundary, the spark becomes illegal and a random spark will generate to replace it. In this way, the diversity of the population is improved. Moreover, the running time of EFWA is reduced as no need to calculate the position of the new spark by any formula.

D. Selection Strategy

The selection strategy in EFWA is called as elite random selection. The best individual is always kept for next generation and the other $(N-1)$ individuals are selected randomly.

There are both advantages and disadvantages for the selection strategy. By selecting the other individuals randomly, the running time for the algorithm is greatly reduced compared with conventional fireworks algorithm, as it is unnecessary to calculate the distance between each individual or sort the individuals by their fitness values. However, since the sparks are distributed in several clusters and the selection strategy is random selection, there is chance for the selected individuals fall into a same cluster. In fact, in the elite random selection, the sparks in the best cluster have more chance to be selected, since at least one spark is selected from the best cluster before the other sparks are selected randomly. If the selected sparks fall into the same cluster, the diversity of the population may loss.

EFWA provides a new way to solve complex optimization problems and is easy to implement. As a matter of fact, it is the best fireworks algorithm published so far. However, EFWA can do better if it has sufficient interaction mechanism.

III. The Interaction Mechanism in EFWA

Interaction mechanism is important for EFWA. The individuals in a population of EFWA exchange information and hence better finding the global optimum. In addition, the diversity of a population is enhanced by exchanging information among the individuals. However, the interaction mechanism of EFWA is not sufficient in the following two parts.

A. Interaction in Explosion Operator

To obtain the number of sparks, the fitness value of the worst firework is used at first. Then the fitness values of each firework are used to calculate the differences between the fitness values of the worst firework and the current firework. The sum of the differences is also needed to calculate the number of sparks. As a result, the number of sparks is determined by each firework in the population.

The fitness value of the best firework is take into consideration to determine the amplitude of each spark. The differences of the fitness values between the best firework and the current firework are obtained and the sum of the differences is used. Therefore, the amplitude for each firework is calculated and affected by each firework.

However, the interaction of fireworks is not sufficient when obtaining the number and amplitude of the fireworks. For any sparks, the information provided by the fireworks has little effect. As it can be seen from equations (1) and (2), the number and amplitude of a spark can easily be polarized. In other words, the best firework will generate overwhelming number of sparks, while the other fireworks can hardly produce any sparks.

Fig. 1 (a) represents the number of the sparks and Fig. 1 (b) shows the amplitude of the explosions when EFWA

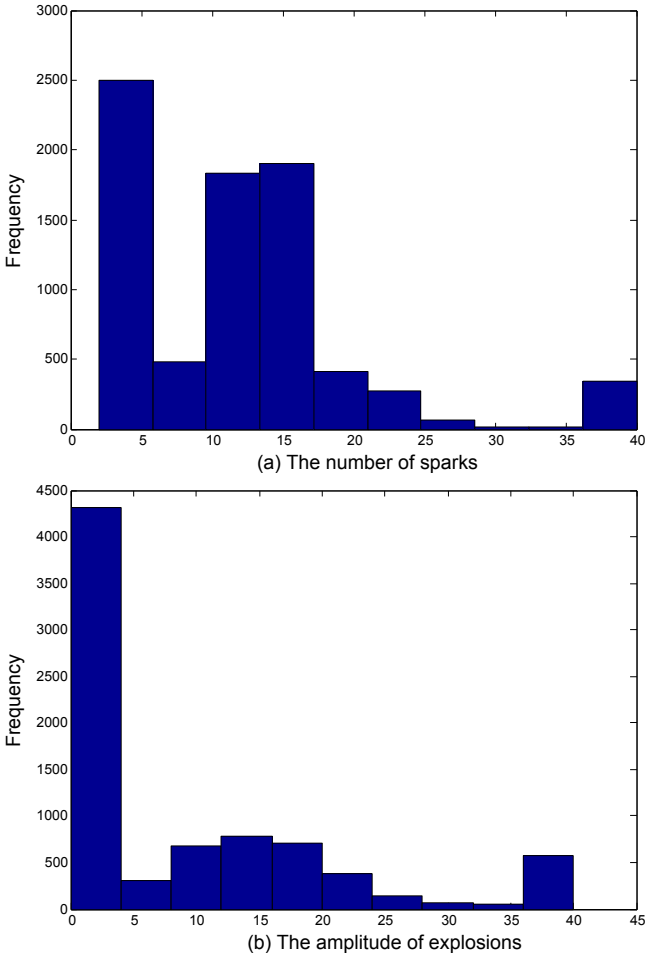

Fig. 1. The statistical of sparks on Sphere function within 100,000 evaluations.

evaluating function (titled: 'Sphere') is run for 100,000 times. Since each calculation of the numbers and amplitudes are followed by several function evaluations, the sum of the frequency for both numbers and amplitudes cycles are less than 100,000. Both the number of the sparks and the amplitude of the explosions are polarized.

To overcome the shortcomings of explosion operator, the number of sparks and the amplitude of explosion are not used in this way. The improvements will be given in section IV, part D.

B. Interaction in Mutation Operator

Gaussian mutation (GM) operator is used in EFWA. The information of the fireworks is used to generate new sparks that follow the Gaussian distribution. However, GM works poor in EFWA. In one hand, GM only effects the fireworks, ignoring the sparks that are generated by the explosion operator and thus narrowing the interaction between the sparks. On the other hand, the sparks that are generated by GM can hardly be passed down to next generation. In addition, when a spark is generated by GM, it can close to the selected firework, or close to the best firework, or distance to both of them but on the line between the selected firework and the best firework. If the spark is close to any fireworks, it is similar to that firework. If not, the spark can be treated as generated by an explosion with a large amplitude, which is always a bad spark. Hence, the mutation operator is unable to provide much interaction between the fireworks.

IV. EFWA with Differential Mutation

The interaction mechanism in EFWA is not sufficient, but the DM operator can compensate this disadvantage. The main idea of DM operator is to find the solution of a problem by utilizing the difference between individuals. In this way, the individuals in a population communicate with each other and help to find the solution, improving the interaction in an algorithm. Therefore, the idea of differential mutation can be introduced into the EFWA to help with increasing the information exchange of the individuals as well.

A. Differential Mutation Operator

Operator: DM/best/1/exp. In this operator, DM means the differential mutation operator and the word best indicates that the best one is kept for the mutation. Number one means the number of difference vectors used and the abbreviation 'exp' stands for an exponent recombination. The formula for this operator is as follow.

$$X_{i1}^k = X_B^k + F * (X_{i2}^k - X_{i3}^k) \tag{6}$$

In the formula, X_{i1}^k means the k dimension of the target individual and F is the scale factor generally between 0 and 2 (Storn and Price, 1995). X_B^k is the k dimension of the current best individual, while X_{i2}^k and X_{i2}^k are two distinguish random individuals on their k dimensions.

B. The interaction mechanism in DM

First, the DM operator uses the information from the best individual. Therefore, the best information is spread among future individuals. Second, the other two distinguished individuals are chosen and the information of the difference between the two individuals are also used. Third, the two individuals are selected randomly, letting all individuals in a population get the chance to be selected and an equal opportunity for its information to be used. Due to DM, the information of any population is fully utilized.

C. The comparison between GM and DM

In EFWA, the GM operator is adopted to generate sparks. The sparks are calculated according with $X_i^k = X_i^k + (X_B^k - X_i^k) * e$, where X_i is the current firework, X_B is the current best firework, k stands for dimension and $e = Gaussian(0, 1)$. The left part of Fig.2 shows how the sparks are generated by the GM operator. The sparks are distributed in a line across the best firework and the selected firework, utilizing the information of the best firework to improve the current firework.

The right part of Fig.2 represents the way of generating sparks by the DM operator. The sparks are calculated as $X_{i1}^k = X_B^k + F * (X_{i2}^k - X_{i3}^k)$, where X_{i1}, X_{i2} and X_{i3} are distinguish fireworks and X_B is the best firework. F is

a scale factor and k stands for dimension. The best firework searches the space by adding a distance that is the difference between the selected fireworks NO.1 and NO.2. Since the fireworks are selected randomly, each firework has an equal chance to donate its information. In this way, the information of all the fireworks is used to help the best firework finding the optimum. Hence, DM operator is better than GM operator when utilizing the information and keeping the diversity of the population.

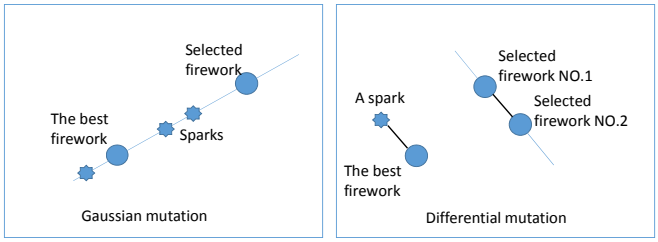

Fig. 2. The difference between GM and DM.

D. Apply DM to EFWA

After introduced EFWA in section II and DM in section IV, it is important to apply DM to EFWA and thus compensate the shortcomings of EFWA.

N denotes the number of individuals in a population for EFWA, which does not change during the optimization process. At first, N individuals are selected randomly and should lie in the feasible space. The N individuals form a population and the population is marked as POP1. Next, for each individual, a spark is produced around it within a certain amplitude.

$$A = A_{min} * rand(0, 1) \tag{7}$$

In the formula, A means the amplitude of each firework, while $Amin$ decreases with the way of non-linear. $rand(0, 1)$ generates random number from 0 to 1.

Then, each new generated spark is compared with its corresponding individual. The one with a better fitness value is kept and used to form a new population with N individuals marked as POP2. Finally, the DM operator is applied to POP2 and a new population is generated as POP3.

To select the individuals for next generation and continue the evolutionary process, the individuals in population POP3 are compared with individuals at the correspondence places in population POP2. The better ones are selected and passed down to the next generation, forming a new population POP1. The iteration of FWA-DM continues till the terminate condition is met.

The process of applying DM to EFWA is drawn in Fig.3. The first row represents population POP1 with N individuals. The second row shows the generated explosion sparks after applying EFWA to POP1. After comparing the sparks in the first row with explosion sparks in the second row, better sparks are chosen and displayed in the third row. Then DM

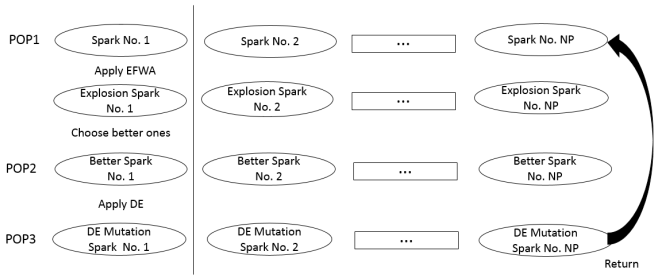

Fig. 3. The process of applying DM to EFWA.

operator is used and population POP3 is produced. The better individuals between population POP2 and POP3 are selected for next iteration as a new population POP1. It is obvious that since DM is introduced, the communication of individuals is enhanced. As a result, the diversity of the population is guaranteed. In summary, the Algorithm for FWA-DM is given below.

Algorithm 1 The process of FWA-DM

```
randomly generate N individuals as POP1
while FuncEval ≤ MaxEval do
    generate N sparks from POP1 as explosion sparks
    choose better individuals as POP2
    apply DM operator and generate POP3
    choose better individuals between POP2 and POP3 as
    a new POP1
end while
```

In Algorithm 1, $MaxEval$ stands for the maximum function evaluation times. Parameter $FuncEval$ represents the current function evaluation times. It can be seen from Algorithm 1 that FWA-DM is simple. Hence, the new algorithm is easy to implement.

V. Experiments and Discussion

To test the performance of the new algorithm, experimental environment is given, extensive experiments are conducted and the experimental results are discussed.

A. Design of Experiments

To make the experimental results more convincing, 30 standard benchmark functions are chosen from the competition of CEC 2014 [16] to verify the effectiveness of FWA-DM. Also, the latest version of SPSO is used as a baseline [17].

EFWA and FWA-DM were wrote in C language, while SPSO2011 was wrote in Matlab language. The experimental platform are Visual Studio 2012 and Matlab R2013b. All the programs are running on 64-bit Window 8 operating system with an Intel Core E8400 with 3.00GHz and 6GB RAM. Each experiment runs 51 times and during each run, the fitness functions are evaluated 300,000 times for SPSO2011 and FWA-DM and just over 300,000 times for EFWA, according to the rules in [16]. The function evaluation times

for EFWA cannot be equal to 300,000 because the number of sparks is not fixed in each generation. Therefore, once the number of function evaluations exceeds 300,000 at the end of a generation, there will be no more generations.

The parameters for FWA-DM are setting as follows. Parameters A_{init} and A_{final} are set as 20 and 0.001, while the population size is set as 5 times of the dimension and the parameters F and CR are set as 0.5 and 0.9, respectively, as in [3]. The dimension in the experiments is set as 30 for all functions. The parameters for EFWA and SPSO2011 are all set as their default settings.

The way to calculate algorithm complexity is according with [16] and describes as follows.

a) Run the test program below:

```
for i = 1:1000000
    x = 0.55 + (double)i;
    x = x + x; x = x / 2; x = x * x; x = sqrt(x);
    x = log(x); x = exp(x); x = x / (x + 2);
end
```

The computing time is set as T0;

b) Evaluate function 18 without an algorithm for 200,000 times and mark the time as T1;

c) Use an algorithm to evaluate function 18 for 200,000 times and mark the time as T2;

d) Run step c) for five times and $\hat{T}2$ denotes the average time.

The complexity of an algorithm is indicated by ($\hat{T}2$ - T1) / T0;

B. Experimental Results

The mean and standard deviation for the algorithms on 30 functions are given in Table I. The best result for each function is shown in bold. The last row means the experiment results of three algorithms. Sign + stands for the number of functions which an algorithm performs the best, while sign - means the number of functions that the algorithm is not the best.

Table II gives the *t-test* results for FWA-DM versus EFWA. *T-test* is used to discriminate whether the two sets are significantly different from each other when the two sets follow a normal distribution. The null hypothesis is that the two sets are the same from each other when follow a normal distribution. Then a *p-value* is calculated and shown in Table II. The null hypothesis is rejected if the *p-value* is lower than a threshold. The threshold chosen for statistical significance is 0.05. As a result, the bold numbers in Table II mean that the experimental results of new algorithms are significantly better than EFWA.

The computational complexity of the algorithms is given in Table III.

C. Discussion

The selected benchmark functions contain unimodal, multimodal, hybrid and composition functions. Therefore, the experimental results are objective and do not skew to any kind of benchmark functions.

TABLE II
T-TEST RESULTS FOR FWA-DM VERSUS EFWA ON 30 BENCHMARK FUNCTIONS

NO.	FWA-DM	NO.	FWA-DM
1	**0.0000000929**	16	**0**
2	**0**	17	**0**
3	**0**	18	**0.0000000002**
4	**0**	19	**0.0006685466**
5	0	20	**0**
6	**0**	21	**0**
7	**0.0151942843**	22	**0**
8	**0**	23	**0**
9	**0**	24	**0.0000000064**
10	**0**	25	**0**
11	**0**	26	**0.0076079049**
12	0.0000000018	27	**0**
13	**0**	28	**0**
14	0.0424872565	29	**0.0000023067**
15	**0**	30	**0**

TABLE III
TIME COMPLEXITY OF THE THREE ALGORITHMS

	EFWA	FWA-DM
T0	0.097	0.097
T1	1.591	1.591
$\hat{T}2$	2.1778	3.7782
($\hat{T}2$-T1)/T0	6.049485	22.54845

It can be seen from the mean values that FWA-DM performs the best. For the 30 functions, FWA-DM performs best on 23 functions. After applying DM operator to EFWA, the performance of the new algorithm FWA-DM improved. When comparing FWA-DM with EFWA and SPSO2011 separately, FWA-DM defeats EFWA on 27 functions and outperforms SPSO2011 on 25 functions. The t-test results show that the better results of FWA-DM are significantly different from the results of EFWA.

However, the running times for the three algorithms cannot be compared, as the algorithms are not running with the same language. But the computational complexity can be compared. It can be seen that the computational complexity of FWA-DM is similar with FWA.

From the convergence curves, it can be seen that EFWA convergences quicker than the other algorithms on only 2 functions and FWA-DM defeats the other algorithms on 17 functions. Therefore, FWA-DM convergences very quickly and is able to find better solutions in feasible space. By applying the DM operator to EFWA, the new algorithm achieves much better on benchmark functions with computational complexity similar to EFWA.

VI. CONCLUSIONS

DM operator was introduced to improve the performance of EFWA, which was the best fireworks algorithm published so far. The new algorithm named FWA-DM outperformed EFWA on most functions. FWA-DM provided a brand new way to solve function optimization problems. Experimental results on 30 benchmark functions of CEC 2014 proved that FWA-DM could solve many function optimization problems

TABLE I

MEANS AND STANDARD DEVIATION OF THREE ALGORITHMS

Function NO.	EFWA	FWA-DM	SPSO2011
1	6.590564117e+005(2.624037120e+005)	3.793203067e+005(2.202342113e+005)	**2.737560700e+005(1.125073475e+005)**
2	1.868574831e+004(9.784967004e+003)	**7.526010749e-017(1.541127529e-016)**	9.579735978e+003(6.065698971e+003)
3	1.945009593e-001(1.095034039e-001)	**3.927807415e-016(5.474012943e-016)**	4.019159568e+003(1.357287539e+003)
4	5.796178581e+001(3.038229108e+001)	**1.977635781e+001(1.400829774e+001)**	2.608423676e+001(3.901368080e+001)
5	**2.000002158e+001(8.155149242e-006)**	2.055422287e+001(4.578315564e-002)	2.068163978e+001(1.066970080e-001)
6	3.700792250e+001(3.373283698e+000)	1.309931690e+001(5.083891761e+000)	**1.236312207e+001(2.676969612e+000)**
7	1.678555789e-002(2.192370498e-002)	**8.450100065e-003(8.854007130e-003)**	9.261812314e-003(1.235762978e-002)
8	2.247331504e+002(4.326519626e+001)	**1.365667057e-001(3.950259268e-001)**	5.086607469e+001(1.743150326e+001)
9	4.160323778e+002(8.035818516e+001)	**4.207809461e+001(6.467525810e+000)**	4.993291136e+001(1.168056952e+001)
10	1.308465212e+003(4.385391599e+002)	**1.710415369e+001(4.497146171e+000)**	2.953981277e+003(6.156040662e+002)
11	4.345088333e+003(5.199235800e+002)	**2.484430278e+003(3.145416563e+002)**	3.477008186e+003(5.797380597e+002)
12	**3.774070492e-001(1.897832294e-001)**	5.901572072e-001(1.164365705e-001)	1.413664305e+000(3.171655824e-001)
13	5.325835951e-001(1.047093864e-001)	3.403325614e-001(5.149146350e-002)	**2.069957129e-001(3.954360096e-002)**
14	2.459293029e-001(4.892486060e-002)	2.645424312e-001(4.129537169e-002)	**2.117601654e-001(3.684560643e-002)**
15	2.672492717e+001(7.168595705e+000)	**8.388664834e+000(8.986544742e-001)**	8.699160153e+000(3.406189454e+000)
16	1.321105867e+001(3.186589268e-001)	1.101664329e+001(3.091815462e-001)	**1.098119576e+001(7.522169675e-001)**
17	4.831268324e+004(2.785520740e+004)	**1.044383444e+004(1.590957303e+004)**	2.203756781e+004(1.542807708e+004)
18	8.218886367e+003(7.219891022e+003)	**7.323271041e+001(3.427624162e+001)**	1.771030243e+003(1.968208038e+003)
19	2.214596081e+001(2.344459161e+001)	**1.007507018e+001(2.211424352e+000)**	1.437701769e+001(2.164483115e+000)
20	3.515671014e+002(8.670052970e+001)	**3.940122574e+001(2.125750166e+001)**	8.448833903e+002(4.330910035e+002)
21	2.779738728e+004(1.513081973e+004)	**1.626857725e+003(4.964980830e+003)**	1.724438179e+004(9.939286743e+003)
22	6.15686411e+002(2.336785114e+002)	**1.229373020e+002(6.707902254e+001)**	2.438925571e+002(9.378050917e+001)
23	3.152441980e+002(7.781090161e-005)	**3.140128864e+002(8.791743683e-014)**	3.152489463e+002(8.522262075e-004)
24	2.544955227e+002(2.712396424e+001)	**2.275619815e+002(5.544356207e+000)**	2.333281159e+002(6.886763518e+000)
25	2.260048229e+002(1.070018094e+001)	**2.005711369e+002(1.619895866e-001)**	2.125671817e+002(2.283292420e+000)
26	1.258075881e+002(6.474927775e+001)	**1.003362428e+002(5.048526362e-002)**	1.158373307e+002(3.667114340e+001)
27	1.065977291e+003(4.997850261e+002)	**3.991034349e+002(1.191595361e+001)**	6.072160053e+002(1.006545625e+002)
28	3.508693257e+003(4.359844290e+002)	**3.948886456e+002(1.331627806e+001)**	1.011782719e+003(1.274191389e+002)
29	2.075360359e+007(2.750139861e+007)	**2.085394766e+002(3.396449789e+000)**	9.533548601e+005(3.831579048e+006)
30	4.052305573e+003(1.506881516e+003)	**4.288160423e+002(2.179562019e+002)**	5.367027370e+003(1.576129601e+003)
Results	(2+,28-)	(23+,7-)	(5+,25-)

effectively. No matter for theoretical or practical researches, FWA-DM is worth researching and can bring benefits on both scientific and economic fields.

REFERENCES

[1] A. Colorni, M. Dorigo, V. Maniezzo, *et al.*, "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, vol. 142, pp. 134–142, Paris, France, 1991.

[2] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pp. 39–43, IEEE, 1995.

[3] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[4] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (abc) algorithm," *Applied soft computing*, vol. 8, no. 1, pp. 687–697, 2008.

[5] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Advances in Swarm Intelligence*, pp. 355–364, Springer, 2010.

[6] A. Janecek and Y. Tan, "Swarm intelligence for non-negative matrix factorization," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 2, no. 4, pp. 12–34, 2011.

[7] A. Janecek and Y. Tan, "Using population based algorithms for initializing nonnegative matrix factorization," in *Advances in Swarm Intelligence*, pp. 307–316, Springer, 2011.

[8] A. Janecek and Y. Tan, "Iterative improvement of the multiplicative update nmf algorithm using nature-inspired optimization," in *Natural Computation (ICNC), 2011 Seventh International Conference on*, vol. 3, pp. 1668–1672, IEEE, 2011.

[9] H. Gao and M. Diao, "Cultural firework algorithm and its application for digital filters design," *International Journal of Modelling, Identification and Control*, vol. 14, no. 4, pp. 324–331, 2011.

[10] S. Zheng, A. Janecek, and Y. Tan, "Enhanced fireworks algorithm," *IEEE Congress on Evolutionary Computation*, pp. 2069–2077, 2013.

[11] Y. Zheng, X. Xu, H. Ling, and S. Chen, "A hybrid fireworks optimization method with differential evolution operators," *Neurocomputing*, 2012.

[12] Y. Pei, S. Zheng, Y. Tan, and H. Takagi, "An empirical study on influence of approximation approaches on enhancing fireworks algorithm," in *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, pp. 1322–1327, IEEE, 2012.

[13] J. Liu, S. Zheng, and Y. Tan, "The improvement on controlling exploration and exploitation of firework algorithm," in *Advances in Swarm Intelligence*, pp. 11–23, Springer, 2013.

[14] Y. Zheng, Q. Song, and S. Chen, "Multiobjective fireworks optimization for variable-rate fertilization in oil crop production," *Applied Soft Computing*, vol. 13, no. 11, pp. 4253–4263, 2013.

[15] K. Ding, S. Zheng, and Y. Tan, "A gpu-based parallel fireworks algorithm for optimization," in *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, pp. 9–16, ACM, 2013.

[16] J. Liang, B. Qu, and P. Suganthan, "Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization," tech. rep., Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 2013.

[17] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas, "Standard particle swarm optimisation 2011 at cec-2013: A baseline for future pso improvements," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pp. 2337–2344, IEEE, 2013.