J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

The Variable Amplitude Coefficient Fireworks Algorithm with Uniform Local Search Operator

Lixian Li[1,2] and Jaewan Lee[2*]

ABSTRACT

Fireworks Algorithm (FWA) is a relatively novel swarm-based metaheuristic algorithm for global optimization. To solve the low-efficient local searching problem and convergence of the FWA, this paper presents a Variable Amplitude Coefficient Fireworks Algorithm with Uniform Local Search Operator (namely VACUFWA). Firstly, the explosive amplitude is used to adjust improving the convergence speed dynamically . Secondly, Uniform Local Search (ULS) enhances exploitation capability of the FWA. Finally, the ULS and Variable Amplitude Coefficient operator are used in the VACUFWA. The comprehensive experiment carried out on 13 benchmark functions. Its results indicate that the performance of VACUFWA is significantly improved compared with the FWA, Differential Evolution, and Particle Swarm Optimization.

☞ keyword : Fireworks Algorithm; Variable Amplitude Coefficiency; Uniform Local Search; Global Optimization

1. Introduction

The fireworks algorithm (FWA) is a novel swarm-based metaheuristic intelligence algorithm [1]. Inspired by the natural phenomenon of fireworks explosion, Tan and Zhu [1] proposed the FWA in 2010. Fireworks are regarded as a feasible solution in the solution space of the optimization problem in the FWA, and the process of producing a certain amount of sparks can be treatd as the searching neighbor area around fireworks. The FWA has the local search capability and global search capability for keeping diversity of sparks [1]. It is more effective than other swarm intelligent optimization algorithm in finding optimal global values, such as differential evolution (DE) and particle swarm optimization (PSO) algorithm [2, 3]. The FWA has been applied to various practical problems, such as spam detection [4], image-recognition [5], and distribution network reconstruction and optimization [6].

1 School of Information Science and Technology, Jiujiang University, Jiujiang, 332005, P.R.China
2 Department of Information and Communication Engineering, Kunsan National University, Kunsan, 54150, Korea
* Corresponding author: Jaewan Lee(jwlee@kunsan.ac.kr)
[Received 18 December 2019, Reviewed 26 December 2019, Accepted 04 April 2020]
☆ This research is supported by Institute of Information and Telecommunication Technology of kunsan National University, South Korea

Recently, many researchers have studied in-depth and made various improvements to the FWA. There are two categories: one is the operator analysis and improvement; another is the study of the mixed algorithm with other heuristic algorithms.

2. Fireworks Algorithm

FWA has the capability of balancing exploration and exploitation. Superior fireworks with better fitness have a larger quantity of explosive sparks and smaller explosion amplitude than inferior fireworks with worse fitness [10]. Suppose that the dimension is *dm* and the quantity of fireworks is *N*, explosion spark quantity *sn* (Eq.2) of each fireworks x_i, and the explosion amplitude *AE* (Eq.4) is calculated. The calculation manner is as follows.

2.1 The Main Principal of the Standard FWA

2.1.1 Quantity of Sparks

Assume that the FWA was designed for routine optimization problems:

$$Minimize\ f(x) \in \Omega \quad x_{\min} \le x \le x_{\max} \tag{1}$$

$x = x_1, x_2, \ldots, x_n$ represents a position in the potential space of sparks, $f(x)$ is an objective function, x_{max} and x_{min} represent the upper and the lower boundary of the potential space of sparks [1,11].

Eq.2 defines the quantity of sparks generated by each firework x_i. sm is a parameter used to adjust the number of sparks, which are generated by the N fireworks. y_{max} is the maximum fitness(worst) in the current fireworks population, δ is employed to avoid zero-division-error, in addition, it is the smallest constant [7].

$$sn_i = sm * \frac{y_{\max} - f(x_i) + \delta}{\sum_{i=1}^{n}(y_{\max} - f(x_i)) + \delta} \tag{2}$$

In order to control the quality of fireworks, Eq.3 is subject to the quantity of sparks. a and b are immutable parameters.

$$\hat{sn}_i = \begin{cases} round(a*sm) & \text{if } sn_i < a*sm \\ round(b*sm) & \text{if } sn_i > b*sm, a<b<1 \\ round(sn_i) & otherwise \end{cases} \tag{3}$$

2.1.2 Explosion Amplitude

Superiors firework generates more sparks on a smaller region and the amplitude of which is smaller than inferior fireworks. Eq.5 defines the explosion amplitude of each firework [1]. AC is a parameter, which is used to adjust the explosion amplitude. y_{min} is minimum fitness(the best firework) in the current fireworks population.

$$AE_i = AC * \frac{f(x_i) - y_{\min} + \delta}{\sum_{i=1}^{n}(f(x_i) - y_{\min}) + \delta} \tag{4}$$

2.1.3 Generating Explosion Sparks

In the FWA, sparks may be affected by the random z dimensions in the explosion, and we get the quantity of affected directions randomly, as shown in Eq.5 [1]. Rand (0, 1) is the normal distribution at the location x_i, and dm is the dimension.

$$z = round(dm * rand(0,1)) \tag{5}$$

Simulate the explosion process of the fireworks, initialize the location of the spark: $\hat{x}_j = x_i$. Randomly select z dimensions of $\hat{x}_j$, in the set of pre-selected, if the obtained location is found to fall out of the potential space, Eq.6 is used to calculate the potential space of each dimension $\hat{x}_k^j$. $\hat{x}_k^j$ is mapped to the potential space with Eq.7.

$$\hat{x}_k^j = \hat{x}_k^j + AE_i * rand(-1,1) \tag{6}$$

$$\hat{x}_k^j = x_k^{\min} + \left|\hat{x}_k^j\right| \% \left(x_k^{\max} - x_k^{\min}\right), \hat{x}_k^j < x_k^{\min} \text{ or } \hat{x}_k^j > x_k^{\max} \tag{7}$$

2.1.4 Generating Gaussian Sparks

To keep the diversity of fireworks, Gaussian explosion is introduced [7]. The location of spark $\hat{x}_j$ is first generated: $\hat{x}_j = x_i$. Then Eq.5 selects z dimensions of $\hat{x}_j$ randomly. Eq.8 is used to calculate each dimension $\hat{x}_k^j$. If the location is found to fall out of the potential space, $\hat{x}_k^j$ is mapped to the potential space by using Eq.7. Gaussian (-1,1) is a Gaussian distribution with mean 1 and standard deviations 1.

$$\hat{x}_k^j = \hat{x}_k^j + AE_i * Gaussian(-1,1) \tag{8}$$

2.1.5 Selection Strategy

In the selection strategy, the best location x^* corresponding to the current optimal function $f(x^*)$, which is always selected for the next-generation. To keep the diversity of sparks, the next (n − 1) individuals are selected based on their distance to other sparks locations. Eq.9 defines the general distance between an individual x_i and other individuals [10].

$$R(x_i) = \sum_{j \in k} d(x_i, x_j) = \sum_{j \in k} \left\| x_i - x_j \right\| \tag{9}$$

k denotes the set, which contains both sparks and fireworks in current locations. Eq.10 defines the selection probability of a location x_i. Manhattan distance [17] is utilized to measure distance in the FWA. $|f(x_i) - f(x_j)|$ defines $d(x_i, x_j)$, the selection probability is high in low crowed regions, which is same as the definition of the immune density-based probability [12].

$$p(x_i) = \frac{R(x_i)}{\sum_{j \in k} R(x_j)} \tag{10}$$

2.2 Framework of FWA

Algorithm 1 summarizes the framework of FWA. In the process of each explosion, two types of sparks are produced: explosion sparks and Gaussian sparks. For the first type, the quantity of sparks and the amplitude of the explosion depend upon the quality of the relevant fireworks $f(x_i)$.

By comparison, the second type is generated by the Gaussian explosion strategy, which carries out searching in the local Gaussian distribution around the firework. After the FWA gets access to the locations of these two types of sparks and selects n positions for the next generation in the explosion.

Algorithm 1. Framework of FWA

```
Initialize the basic parameters of the algorithm
Randomly select n fireworks in the potential space
Evalute their fitness
  Repeat
    Calculate the number of sparks that each firework generates
    Calculate the exploding scope of sparks that each firework generates
    Generate the sparks, based on the number of sparks and scope of explosion
    Generate Gaussian sparks
    Combine fireworks, Explosion sparks, and Gaussian sparks to form a population
    Evaluate the fitness of each individual
    Select the next generation fireworks n
  Until termination criteria is met
Return the best individual and its fitness
```

3. The Proposed VACUFWA

3.1 Uniform Local Search

Peng [13] proposes the ULS operator based on uniform design, and use it to enhance the local search capability of the DE algorithm. Experimental results prove the excellent local search performance of the ULS. The uniform design experiment is arranged by a uniform design table. The uniform design table is a table with n rows and k columns, and the times of experiments are equal to the number of levels, namely $n=q.U_n(_q^k)$ is used to represent the uniform design table. Where U denotes the uniform design, n denotes the number of experiments, q denotes the number of levels of each factor, and k denotes the maximum number of independent factors [14].

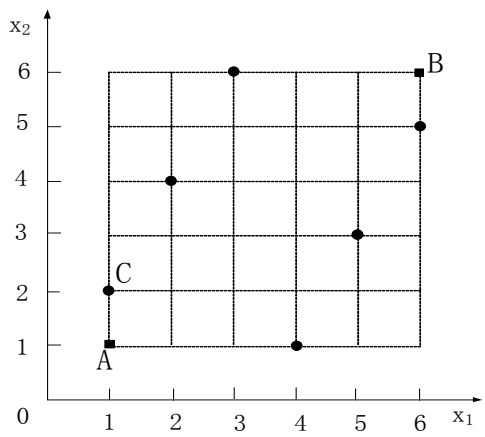
In the literature [18], the $U_7(_7^6)$ is used, as shown in Table 1, the process of forming a uniform design table can reference to the literature [15]. The uniform design allows each factor to have the largest number of levels. Therefore, the number of levels is equal to the number of experimental studies [13]. Considering the following two aspects of the $U_7(_7^6)$. Firstly, all the number of the last line in Table 1 are 7, so this represents that the last line is redundant information of individuals. Secondly, after several verification tests, $U_6(_6^6)$ is the most appropriate. Therefore, the last row of the $U_7(_7^6)$ is removed to form the uniform design table. Each factor of the $U_6(_6^6)$ has 6 levels that can obtain effective information from the experiment [18].

In two-dimensional space as shown in Figure 1, the ULS is used to select two individuals A and B randomly in population P and perform a set of experiments, Its search space is a two- dimensional space composed of A and B. Each dimension of the search space is decomposed evenly into six parts representing six levels. The six trial individuals are generated in the search space which is formed by A and B. Assuming that C is optimal in the six experimental individuals, then the first dimension of C is level 1, and the second dimension is level 2.

As a general local search framework, the ULS can be applied to other evolutionary algorithms to improve search capability. The ULS operator is shown in Algorithm 2.

(Table 1) Uniform design table $U_7(_7^6)$

Experiment number	Factors					
	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1
7	7	7	7	7	7	7



(Figure 1) Illustration of the uniform local search with two randomly chosen individuals

Algorithm 2. ULS operator

1: Input: population P, fitness function evaluation times *evaTime*
2: Two individuals, $x_{i,G}$ and $x_{j,G}$, are randomly selected from population P
3: Based on $U_6(6^6)$, six experimental individuals(y_1 ...y_6) are constructed between individual $x_{i,G}$ and $x_{j,G}$
4: Evaluate the target function value $f(y_1)...f(y_6)$
5: Choose the optimal individual x* from $y_1...y_6$
6: If ($f(x_{i,G})>f(x^*)$)
7: $x_{i,G}=x^*$
8: End if
9: *evaTime* = *evaTime* +6
10: Update population *P*, return *evaTime*
13: Return the best individual and its fitness

3.1 The uniform local search operator and variable amplitude coefficiency are fused

The ULS operator conducts a local search in the selected next-generation fireworks. Although the ULS is an effective method to improve the exploitation capability, it is a relatively greedy mechanism. The more times it is executed, the higher probability of falling into local extreme value. Therefore, the convergence rate and population diversity must be balanced, and ULS is executed once for each generation of the population.

In the FWA, the value of the explosion amplitude coefficiency is fixed, which may not correspond to a real process of a fireworks explosion. Generally, a larger value of explosion amplitude is suitable for exploring new search space, while a smaller value of explosion amplitude is suitable for a local search. Therefore, the explosion amplitude has a significant influence on global exploration and convergence of the algorithm. The variable amplitude coefficiency (VAC) decreases the explosion amplitude as the number of evaluations increases. The formula of VAC in VACUFWA is as follows:

$$VAC(evaTime) = 40 / (1 + \exp(0.015 * (evaTime\text{-}maxEva) / 3)) \quad (11)$$

Where *evaTime* is the current number of iterations, and *maxEva* is the maximum number of iterations.

Using excellent local search performance of the ULS operators to improve the quality of the FWA, the VACUFWA enhances local search capability and achieves the balance of convergence speed and global exploration. The framework of the VACUFWA isas follows Algorithm 3.

Algorithm 3. Framework of VACUFWA

1: Initialize the basic parameters of the algorithm
2: Randomly select n fireworks in the potential space
3: Evalute their fitness
4: Repeat
5: Calculate the number of sparks that each firework generates
6: Calculate the exploding scope of sparks that each firework generates
7: Generate the sparks, based on the number of sparks and scope of explosion
8: Generate Gaussian sparks
9: Combine fireworks, Explosion sparks, and Gaussian sparks to form a population
10: Evaluate the fitness of each individual
11: Select the candidate fireworks *n*
12: ULS is used to search next-generation fireworks *n*
13: Until termination criteria is met
14: Return the best individual and its fitness

4. Experimental simulation and analysis

4.1 Experimental parameters settings and test function

The experimental hardware environment contains Intel Core i7-4770 CPU@3.40GHz processor, 8GB memory, and 64-bit operating system. The software environment contains the Windows7 operating system and the Matlab R2018a version.

The parameters of VACUFWA and the comparison algorithms are set to the same. In the experiment, the dimensions of all 13 test functions are set to 30, and each function is run 30 times.

13 well-known benchmark functions are adopted to analyze and verify the quality of the VACUFWA. Function f_1~f_7 is the unimodal function, and f_8~f_{13} is the multimodal function with multiple minimum values [15].

To evaluate the experimental objectivity, the Wilcoxon rank-sum test and Friedman test in statistics are used to analyze the experimental results. The Wilcoxon rank-sum test based on the rank-sum of samples, which works for estimating whether two samples come from the same population and for analyzing whether the experimental results of the comparison algorithm have significant differences [19]. Friedman test uses rank to analyze whether there are significant differences in the population distribution of multiple independent samples. By ranking the mean rank of each sample to reflect the performance of the algorithm, the smaller the mean rank, the better the performance of the algorithm [20].

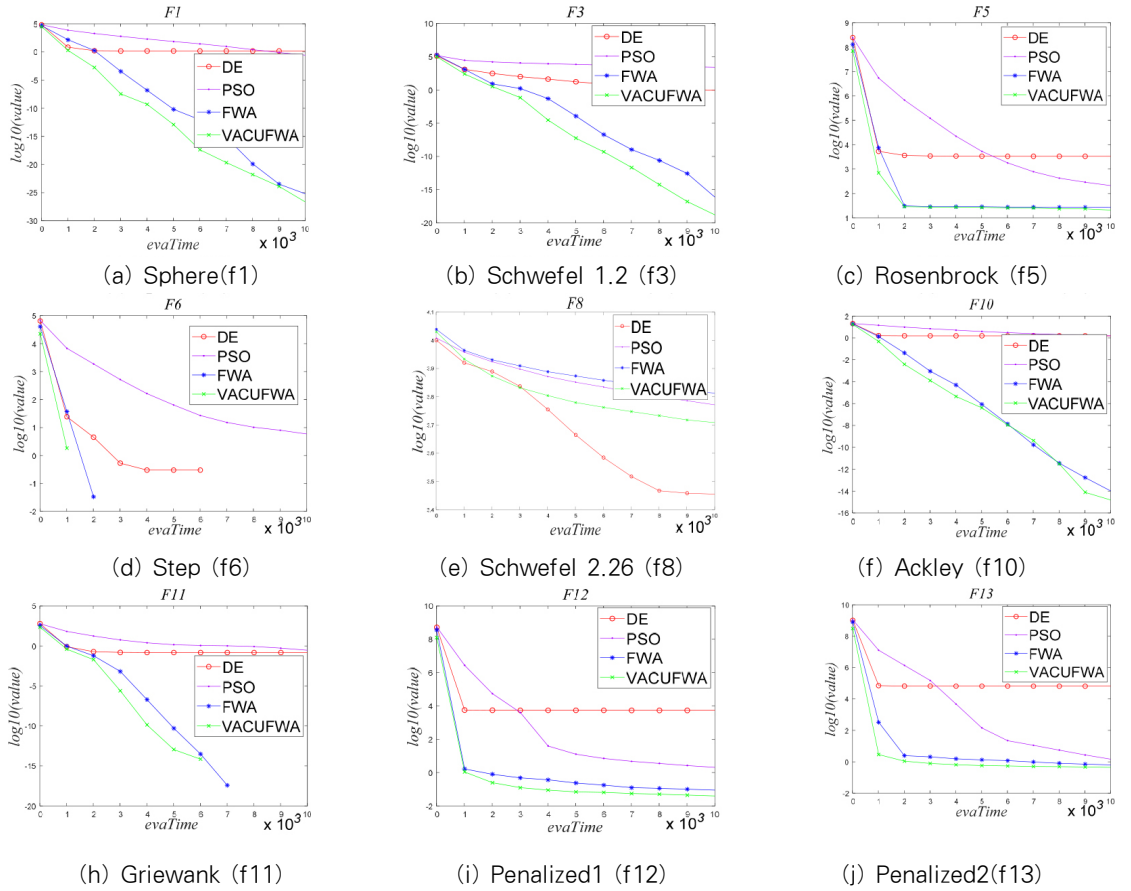
4.2 Experimental results and comparative analysis

The experimental mean and standard deviation of the corresponding DE, PSO, FWA, and VACUFWA are compared. The Friedman test means rank of the experimental results of DE, PSO, FWA, and VACUFWA are 3.62, 3.15, 1.81, and 1.42 respectively, which indicates that the performance of VACUFWA is the best among the comparison algorithms.

The specific data is shown in Table 2; The former three algorithms are compared with the quality of the solution of VACUFWA. The three symbols of "-," "+," and "≈" represent the inferior, superior, and similarity of the solution of these algorithms respectively. The **bold** font indicates the optimal value of the mean in the comparison. Statistics of the Wilcoxon rank-sum test results of experimental data show that VACUFWA is superior to the other three algorithms in the 13 test functions. The mean of VACUFWA for the most functions is significantly improved than the FWA. It can be seen that the ULS operator and variable amplitude

(Table 2) Mean(standard deviation) comparison of DE, PSO, FWA, and VACUFWA(D=30, EavTimes=10000)

Fun	DE	PSO	FWA	VACUFWA
f1	8.47E+00(2.14E+01)-	1.80E-01(2.15E-01)-	2.99E-29(1.56E-28)-	**3.49E-31**(1.80E-30)
f2	1.48E-01(2.83E-01)-	1.92E-01(2.91E-01)-	**1.02E-19**(2.80E-19)+	2.69E-18(9.85E-18)
f3	1.02E+03(4.92E+02)-	2.34E+03(8.84E+02)-	3.68E-18(1.94E-17)-	**7.84E-21**(4.10E-20)
f4	2.46E+01(7.25E+00)-	1.29E+01(3.46E+00)-	**1.14E-13**(3.25E-13)+	2.62E-12(1.30E-11)
f5	4.44E+04(1.83E+05)-	1.98E+02(1.25E+02)-	2.69E+01(5.81E+00)-	**2.09E+01**(1.12E+01)
f6	2.65E+01(4.15E+01)-	5.60E+00(1.07E+01)-	**0.00E+00**(0.00E+00)≈	**0.00E+00**(0.00E+00)
f7	1.25E-01(6.46E-02)-	5.92E-02(2.08E-02)-	**3.54E-03**(4.36E-03)+	4.61E-03(3.65E-03)
f8	**-4.17E+03**(4.58E+02)+	5.85E+03(6.47E+02)-	6.37E+03(7.51E+02) -	4.99E+03(9.55E+02)
f9	1.88E+02(1.83E+01)-	5.05E+01(1.27E+01)-	**0.00E+00**(0.00E+00)≈	**0.00E+00**(0.00E+00)
f10	2.11E+00(1.27E+00)-	1.50E+00(8.88E-01)-	4.74E-16(1.21E-15)-	**2.37E-16**(8.86E-16)
f11	6.52E-01(4.43E-01)-	2.09E-01(1.60E-01)-	**0.00E+00**(0.00E+00)≈	**0.00E+00**(0.00E+00)
f12	9.88E+03(3.23E+04)-	1.87E+001.30E+00)-	8.03E-02(6.07E-02)-	**3.61E-02**(2.54E-02)
f13	1.26E+05(5.59E+05)-	1.19E+00(2.01E+00)-	5.97E-01(3.55E-01)-	**4.22E-01**(2.00E-01)
-/+/≈	12/1/0	13/0/0	7/3/3	



(a) Sphere(f1) (b) Schwefel 1.2 (f3) (c) Rosenbrock (f5)

(d) Step (f6) (e) Schwefel 2.26 (f8) (f) Ackley (f10)

(h) Griewank (f11) (i) Penalized1 (f12) (j) Penalized2(f13)

(Figure 2) The convergence curves of the standard DE, PSO, FWA, and VACUFWA on 9 test functions

coefficiency contributes significantly to VACUFWA, which has a good effect on functions f1, f3, f5, f8, f10, f12, and f13. To reflect the convergence process of the improved algorithm in detail, Figure 2 shows the convergence curves of 9 representative functions.

The abscissa in the figure represents the evaluation times of the function, which upper bound is 10000. The ordinate represents the mean of the objective function of 30 experiments in the form of $log_{10}(f)$.

The following conclusions can be drawn from Figure 2. Firstly, the convergence of FWA is significantly better than DE, PSO, and the VACUFWA converges faster than FWA in the most test function. This phenomenon is caused by the strong local search capabilities of the ULS operator. Secondly, the VAC increases the rate of convergence of the test functions. In the early stage of evolution, the convergence speed of the FWA is similar to VACUFWA. In the middle and late stages of evolution, the VAC will decrease with the increase of evaluation times, which will cause the amplitude of explosion to decrease to generate more sparks in a small explosion range. Therefore, the convergence speed of VACUFWA is faster than the FWA in terms of functions f1, f3, f6, f8, f11, f12, and f13. Finally, function f6 and f11 quickly converge to the optimal value; the solution quality of VACUFWA is also better than the FWA.

5. Conclusion

In this paper, the ULS operator and VAC are used to VACUFWA for improving the search quality and

convergence speed of FWA respectively. VACUFWA balances further exploration and exploitation capability. According to the experimental results, the quality of VACUFWA is better than the standard DE and PSO. Meanwhile, VACUFWA also shows the superiority of the FWA, which proves that the ULS operator and VAC are effective. The result of experiments shows that VACUFWA is very competitive. We believe VACUFWA can bring significant advantages to practical optimization problems. Our ongoing work includes improving the current operators of VACUFWA to make the algorithm suitable for different functions and implementing the experiment of VACUFWA on the cloud computing platform. In the future, we will also seek the potential hybridization strategies of FWA.

References

[1] Y. Tan and Y. Zhu, “Fireworks Algorithm for Optimization, ” Advances in Swarm Intelligence, pp. 355–364, Springer, 2010.
https://doi.org/10.1007/978-3-642-13495-1_44

[2] Y. Tan and Z. Xiao, “Clonal Particle Swarm Optimization and Its Applications in Evolutionary Computation,” CEC 2007, pp. 2303 –2309, 2007.
https://doi.org/10.1109/CEC.2007.4424758

[3] R. Kstron and Pricek, “Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” Journal of Global Optimization, 11(4), pp. 341-359, 1997.
https://doi.org/10.1023/A:1008202821328

[4] S. Zhang and Y. Tan. “A Unified Distance Measure Scheme for Orientation Coding in Identification,” 2013 International Conference on Information Science and Technology, pp. 979-985, 2013.
https://doi.org/10.1109/ICIST.2013.6747701

[5] W. He and G. Mi, “Parameter Optimization of Local-concentration Model for Spam Detection by Using Fireworks Algorithm,” Advances in Swarm Intelligence, Springer, pp. 439-450, 2013.
https://doi.org/10.1007/978-3-642-38703-6_52

[6] Ia. Mohamed and m. Kowsalya, “A New Power System Reconfiguration Scheme for Power Loss Minimization and Voltage Profile Enhancement Using Fireworks Algorithm,” International Journal of Electrical Power and Energy Systems, vol. 62, pp. 312-322, 2014.
https://doi.org/10.1016/j.ijepes.2014.04.034

[7] S. Zheng, A. Janecek, and Y. Tan, “Enhanced Fireworks Algorithm,” 2013 IEEE Congress on, pp.2069-2077, 2013.
https://doi.org/10.1109/CEC.2013.6557813

[8] J. Liu, S. Zheng, and Y. Tan. “The Improvement on Controlling Exploration and Exploitation of Firework Algorithm,” Advances in Swarm Intelligence, Springer, 11-23, 2013.
https://doi.org/10.1007/978-3-642-38703-6_2

[9] Y. J. Zheng, X. L. Xu, and H. F. Ling et al. “A Hybrid Fireworks Optimization Method with Differential Evolution Operators,” Neurocomputing, vol. 148, pp. 75-80, 2012.
https://doi.org/10.1016/j.neucom.2012.08.075

[10] Y.Tan and S. q. Zheng. “Research Progress of Fireworks Algorithm,” CAAI Transactions on Intelligent Systems, vol. 9, no. 5, pp. 515-528, 2014.
https://doi.org/10.3969/ j.issn.1673-4785.201409010

[11] Engelbrecht and Andries P. ″Fundamentals of Computational Swarm Intelligence,″ New York : Wiley, 93-129, 2005
https://www.researchgate.net/publication/220695804_Fundamentals_of_Computational_Swarm_Intelligence

[12] G. Lu, D. Tan and H. Zhao, “Improvement on Regulating Definition of Antibody Densityof Immune Algorithm,” Proceedings of the 9th International Conference on Neural Information Processing, vol. 5, pp. 2669–2672, 2002.
https://doi.org/10.1109/ICONIP.2002.1201980

[13] H.Peng, Z J.Wu and C S. Deng, “Enhancing Differential Evolution with Commensal Learning and Uniform local search,” Chinese Journal of Electronics, vol. 26, no. 4, pp. 725-733, 2017.
https://doi.org/10.1049/cje.2016.11.010

[14] X. J. Wang, H. Peng and C. S. Deng et al. “Firefly Algorithm Based on Uniform Local Search and Variable Step Size,” Journal of Computer Applications, vol. 38, no. 3, pp. 715-721, 2018.
http://en.cnki.com.cn/Article_en/CJFDTotal-JSJY201803

020.htm

[15] X.Yao, Y. Liu and G. Lin, “Evolutionary Programming Made Faster,” *IEEE* Transactions on Evolutionary Computation, vol. 3, no. 2, pp. 82-102, 1999. https://doi.org/10.1007/978-3-642-25566-3_28

[16] S. Bureerat, “Hybrid Population-Based Incremental Learning Using Real Codes,” Learning and Intelligent Optimization, pp. 379-391, 2011. https://doi.org/10.1007/978-3-642-25566-3_28

[17] V. Perlibakas, “Distance Measures for PCA-Based Face Recognition,” Pattern Recognition Letters, vol. 25, no. 6, pp.711-724, 2004. https://doi.org/10.1016/j.patrec.2004.01.011

[18] K. T. Fang, D. K. J. Lin and P.Winker et al, “Uniform Design: Theory and Application,” Technometrics, vol. 42, no. 3, pp. 237-248, 2000. https://doi.org/10.1080/00401706.2000.10486045

[19] B. Rosner, R. J. Glynn and M. L. Ting Lee. “Incorporation of Clustering Effects for The Wilcoxon Rank Sum Test: A Large Sample Approach,” Biometrics, vol. 59, no. 4, pp.1089-1098, 2003. https://doi.org/10.1111/j.0006-341X.2003.00125.x

[20] M. Friedman. “The Use of Ranks to Avoid The Assumption of Normality Implicit in The Analysis of Variance,” Journal of The American Statistical Association, vol. 32, no. 200, pp. 675-701, 1937. https://doi.org/10.1080/01621459.1937.10503522

◐ 저 자 소 개 ◑

Lixian Li

2005 B.S. in Computer Science and Technology, Zhengzhou University, Zhengzhou, P.R.China
2010 M.S. in Computer Technology, Huazhong University of Science and Technology, Wuhan, P.R.China
2005.07~2018.08 Lecturer at School of Information Science and Technology, Jiujiang University, Jiujiang, P.R.China
2018.09~Present pursuing Ph.D. in Kunsan National University, Kunsan, Korea
Research Interests: Intelligent Operation, Data Mining
E-mail: llxkansan@kunsan.ac.kr

Jaewan Lee

1984 B.S. in Computer Engineering, Chung-Ang University,Seoul, Korea
1987 M.S. in Computer Engineering, Chung-Ang University, Seoul, Korea
1992 Ph.D. in Computer Engineering, Chung-Ang University, Seoul, Korea
1995.03~present, Professor at the Department of Information and Communication Engineering, Kunsan National University
Research Interests: distributed systems, cloud computing, data mining, and computer networksE-mail: jwlee@kunsan.ac.kr