Computing, Performance and Communication Systems (2020) Vol. 4: 1-8
Clausius Scientific Press, Canada
DOI: 10.23977/cpcs.2020.41001
ISSN 2371-8889

A hybrid algorithm based on cuckoo search and differential evolution for numerical optimization

Jun Xi[a], Liming Zheng[b,*]

School of Information Science and Technology, Jinan University, Guangzhou, 510632, China
a. xij0533@163.com, b. zlmqit@126.com
**corresponding author*

Keywords: Cuckoo search; Differential evolution; Numerical optimization

Abstract: Cuckoo search (CS) is a new intelligent bionic algorithm, but it is easy to fall into the local optimum. In this paper, a hybrid algorithm based on cuckoo search and differential evolution (CSDE) is proposed. In the global optimization, a control factor is introduced to judge the optimization state of CS, further determine whether it is necessary to combine the differential strategy to improve the population diversity. Besides, different evolutionary strategies are used to enhance the local search. Numerical experiments, carried on 28 benchmark functions from CEC 2013, demonstrate that CSDE successfully achieves better optimization performance than other competitive optimization algorithms.

1. Introduction

Meta-heuristic algorithm is an effective optimization method inspired by the operating mechanism in nature, and it's committed to finding the best solution in numerical optimization [1-3]. Differential evolution (DE) imitates the biology evolutionary rule that the survival of the fittest to solve the optimization problem efficiently, which has received extensive attention from scholars [4-6]. Particle swarm optimization is a population-based algorithm derived from the study of the birds' predation behavior [7, 8]. Artificial bee colony (ABC) [9] and firefly algorithm (FA) [10] are enlightened by the survival behavior of bees and fireflies respectively.

CS is a new nature-inspired search technique produced by Yang and Deb in 2009 [11]. It has been widely used to solve many real-word optimization problems, such as image processing [12] and data clustering [13]. Lévy flights random walk (LFRW) and biased/selective random walk (BSRW) are devoted to searching for new solutions, and the combination makes CS superior to others [14]. However, the original algorithm can suffer from premature convergence with the number of iterations increases, because the single mutation strategy and the boundary value constraints result in the population diversity decrease too fast. Therefore, many improved CS variants have been proposed.

Nail and Panda proposed a novel adaptive CS (ACS), which controlled the step sizes by fitness rather than Lévy flights [15]. The collaboration between subpopulations was used in CS to improve the convergence precision [16]. The hybridization of meta-heuristic algorithms provides more powerful disturbance behavior, so that the modified algorithm can better solve the complex optimization problems [17, 18]. Compared with HA and PSO, the combination of HA and PSO (HFPSO) made a better balance between exploration and exploitation [19]. A new hybrid algorithm used the operations of DE and the operations of HS together to ameliorate the global convergence in solving optimal power flow (OPF) problem [20]. Many researchers have also used this hybrid technology in CS. The updating actions of krill herd (KH) were added to CS to explore better search space in an improved algorithm based on CS and KH (CSKH) [21]. Chi et al adopted the good global search mechanism of PSO to enhance the population diversity in CS (CSPSO) [22]. However, it's noteworthy that many improvements mentioned above are devoted to improve the performance of LFRW, and the development of BSRW is still not sufficient. Although a hybrid CS (HCS) based on DE was put forward to obtain better convergence speed in BSRW, it's easy to fall into the local optimum because the current best individual was always used to control the direction of evolution [23].

Therefore, the paper proposes a new hybrid algorithm based on CS and DE (CSDE). In LFRW, a control factor S is introduced, which ensure the differential mutation strategy can be adopted appropriately when the optimal value is not updated multiple times. In BSRW, new solutions are produced by using the crossover and selection strategy rather than the simple random rule, and the aim is to increase the diversity of solutions and achieve a better balance between the exploration and exploitation. The modified algorithm is evaluated on 28 benchmark functions derived from CEC 2013 [24]. The experimental results demonstrate that CSDE shows better performance than other comparative algorithms.

The rest of the paper is organized as follows. Section 2 reviews the basic framework of the standard CS algorithm, and DE algorithm in brief. In Sect. 3, the improved CSDE algorithm is described and analyzed in sufficient details. Subsequently, the effectiveness and robustness of the approach is confirmed by numerical simulations in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2. The original CS and original DE

2.1 The original CS

CS is proposed by imitating the life and reproduction features of cuckoos in combination with the Lévy flight behaviors of some birds and fruit flies [11]. During the process of evolution, new individuals are generated in two essentially components, exploitation by LFRW, and exploration by BSRW. There are N nests $\{X_{i,G_0},\ i = 1, \ldots \ldots, N\}$ randomly generated from the search space [-100,100] at first. When the conditions of stopping the iteration are not satisfied, the population will continue to be updated with the following behaviors.

(1) Lévy flights random walk

Lévy flights can be described as a moving entity that occasionally takes unusually large steps to change its behavior. In LFRW, the process of CS generating new solutions is expressed as follows.

$$X_{i,G+1} = X_{i,G} + \alpha_0 \cdot \left(X_{i,G} - X_{best,G}\right) \oplus \text{Lévy}(\beta) \tag{1}$$

where the scaling factor α_0 usually set to 0.01, $X_{best,G}$ is the best solution obtained in the current population, Lévy(β) is subject to Lévy distribution. Besides, the symbol ⊕ represents entry-wise multiplication.

(2) Biased/selective random walk

In BSRW, if the cuckoo egg is found by the host bird, it will be replaced by a new egg. Excessive experiments have proved that it's sufficient for most numerical optimization problems when the probability of a cuckoo egg being discovered Pa is 0.25 [31]. The progress of BSRW can be calculated by Eq.(2).

$$X_{i,G+1} = X_{i,G} + r \cdot \left(X_{r1,G} - X_{r2,G}\right),\ if(rand[0,1] > Pa) \tag{2}$$

where r is a random number on the range [0, 1], $X_{r1,G}$ and $X_{r2,G}$ are two randomly solutions selected from the population.

2.2 The original DE

DE is an effective swarm intelligent algorithm. During the search, new offspring are mainly generated by three operators, namely mutation, crossover, and selection [4]. In DE, the initiation of the population is the same as in CS, and the implementation of others steps are described below.

(1) Mutation

The mutation process searches for new solutions based on individual differences in the population. It's shown via the following equation.

$$V_{i,G} = X_{r1,G} + F \cdot (X_{r2,G} - X_{r3,G}) \tag{3}$$

where the weighted factor F is set to 0.6, r1, r2 and r3 are random numbers taken from [0, N], and their values are not equal to each other and not equal to i.

(2) Crossover

The crossover technique is introduce to perturb the mutation vector, and it can be carried out as in Eq.(4), where CR is set to 0.1.

$$U_{i,j,G} = \begin{cases} V_{i,j,G} & if(rand[0,1] \le CR)\ or\ (j = j_{rand}) \\ X_{i,j,G} & otherwise \end{cases} \tag{4}$$

(3) Selection

Subsequently, the elite learning strategy is used. The current solution should be update when the candidate solution obtains better fitness, and the progress can be expressed in Eq.(5), where f (·) is the solution fitness.

$$X_{i,G+1} = \begin{cases} U_{i,G} & if\left(f(U_{i,G}) \le f(X_{i,G})\right) \\ X_{i,G} & otherwise \end{cases} \tag{5}$$

3 The hybrid CSDE algorithm

In CS, the individual diversity decreases rapidly with the iteration goes on due to the lack of mutation mechanism, which makes the algorithm is difficult to jump out of the local extremum, and that is the premature phenomenon. Therefore, it's beneficial for CS to enhance the search ability by introducing the excellent mutation strategy of DE. In the paper, CSDE mainly improves the exploration and exploitation ability from two aspects.

In LFRW, a control factor S is introduced to determine whether the algorithm is trapped in a local optimum. The initial value of S is 0, and the threshold value C is 7 in the paper. During the iterative process, if the optimal individual is not updated, $S = S + 1$, when $S > C$ means that the population diversity is too small to produce a better new solution, so the differential strategy can be used to increase the robustness of population, and then S reset to 0.

In BSRW, the discovered solutions should be updated again. Frist, two candidate solutions are generated by the mutation operator and random walk respectively, with the probability Pa. The former is conducive to increasing the global search capability, while the latter is looking for the optimal value in a small range around the individual. The modification can be defined in following equations.

$$V1_{i,G} = X_{r1,G} + F(X_{r2,G} - X_{r3,G}) \ \ if(rand[0,1] > Pa) \tag{6}$$

$$V2_{i,G} = X_{i,G} + r\cdot\left(X_{r1,G} - X_{r2,G}\right) if(rand[0,1] > Pa) \tag{7}$$

Then, the cross-selection strategy is adopted to make the two mutation solutions compete with each other by using a self-adaptive crossover rate Cr. Cr is gradually reduced as the number of iterations increases, which makes the algorithm have a strong ability to climb the local peak, and it's shown in Eq.(8).

$$\mathrm{Cr} = \frac{\sqrt{1-\log(\frac{t}{T}+\lambda)}}{4} \tag{8}$$

where t is the current number of iterations, T is the maximum number of iterations, and λ is a smoothing factor. The update progress is accomplished by using Eq.(9).

$$U_{i,j,G+1} = \begin{cases} V1_{i,j,G} & if(rand[0,1] < Cr \ or \ j == rand[0,D]) \\ V2_{i,j,G} & otherwise \end{cases} \tag{9}$$

Finally, the better solution can be retained by comparing the fitness of the updated solution and the original solution with Eq.(5).

4 Experimental results

4.1 Experimental setup

In this section, 28 benchmark functions from CEC 2013 are chosen for testing the performance of

CSDE, including unimodal functions (F1-F5), multimodal functions (F6-F20) and composition functions (F21-F28). Meanwhile, the proposed CSDE is compares with CS [11], DE [4], PSO [7], HCS [23] and CSPSO [22]. For the sake of fairness, the values of the common parameters in all algorithms are set as follows: the population size NP is 50, the population dimension is D=30, and the number of maximum iterations is set to 10000*D. What's more, other personalized parameters of each algorithm are listed in Table 1.

Table 1: Personalized parameters setting of six algorithms.

Algorithm	Parameters and their values
CS	Pa=0.25, β=1.5, a_0=0.01;
DE	F=0.6, CR=0.1;
PSO	C1=C2=2.0, w_{max}=0.9, w_{max}=0.4
HCS	Pa=0.25, b=1.5, a_0=0.01,F_1=0.6, F_2=0.01, CR_1=0.1, CR_2=0.6;
CSPSO	Pa=0.25, α_{min}=0.01, α_{max}=0.5
CSDE	Pa=0.25, β=1.5, a_0=0.01; F=0.6

Table 2: Experimental results achieved by CS, DE, PSO, HCS, CSPSO and CSDE on 30-dimensional CEC 2013 benchmark functions.

Funcs	CS	DE	PSO	HCS	CSPSO	CSDE
F01	7.59E-08(+)	2.27E-13(+)	1.82E-04(+)	2.88E-13(+)	2.43E-13(+)	**0.00E+00**
F02	6.07E+06(–)	3.59E+07(+)	**2.08E+05**(–)	1.60E+07(+)	3.79E+06(–)	1.23E+07
F03	5.20E+08(+)	1.47E+09(+)	3.04E+08(+)	2.10E+09(+)	2.78E+07(+)	**1.58E+07**
F04	5.83E+04(+)	4.33E+04((+)	**2.28E+01**(–)	3.33E+04(–)	3.06E+04(–)	4.28E+04
F05	6.04E-05(+)	1.29E-13(+)	7.64E-03(+)	2.61E-13(+)	2.77E-13(+)	**1.14E-13**
F06	1.57E+01(–)	3.44E+01(+)	2.55E+01(+)	3.29E+01(+)	**1.48E+01**(–)	1.85E+01
F07	1.34E+02(+)	8.83E+01(+)	4.54E+05(+)	1.03E+02(+)	7.48E+01(+)	**7.19E+01**
F08	2.09E+01(=)	2.10E+01(=)	2.09E+01(=)	2.10E+01(=)	2.09E+01(=)	2.09E+01
F09	3.06E+01(–)	3.02E+01(–)	3.94E+01(+)	3.27E+01(+)	**2.89E+01**(–)	3.16E+01
F10	**2.02E-02**(–)	3.68E+01(+)	1.24E-01(–)	6.57E-01(–)	4.40E-02(–)	9.90E-01
F11	9.84E+01(+)	**5.49E-14**(–)	1.84E+03(+)	6.38E+01(+)	6.14E+01(+)	3.77E+01
F12	2.77E+02(+)	1.60E+02(+)	1.35E+03(+)	1.10E+02(+)	1.31E+02(+)	**8.26E+01**
F13	2.92E+02(+)	1.79E+02(+)	1.44E+03(+)	1.57E+02(+)	1.63E+02(+)	**1.34E+02**
F14	2.84E+03(+)	**1.97E+01**(–)	3.43E+03(+)	2.81E+03(+)	4.68E+03(+)	2.38E+03
F15	4.39E+03(+)	6.43E+03(+)	4.51E+03(+)	4.40E+03(+)	5.60E+03(+)	**4.33E+03**
F16	2.13E+00(+)	2.38E+00(+)	**8.63E-01**(–)	2.05E+00(–)	2.04E+00(–)	2.11E+00
F17	2.75E+02(+)	**3.05E+01**(–)	2.24E+03(+)	1.14E+02(+)	1.90E+02(+)	9.55E+01
F18	3.88E+02(+)	2.22E+02(+)	2.26E+03(+)	1.60E+02(+)	1.95E+02(+)	**1.46E+02**
F19	1.59E+01(+)	**3.11E+00**(–)	3.00E+01(+)	2.01E+01(+)	9.93E+00(+)	6.32E+00
F20	1.29E+01(+)	1.33E+01(+)	1.50E+01(+)	1.43E+01(+)	1.26E+01(+)	**1.22E+01**
F21	**1.73E+02**(–)	2.36E+02(–)	4.34E+02(+)	2.79E+02(=)	3.46E+02(+)	2.78E+02
F22	3.33E+03(+)	**4.33E+02**(–)	6.43E+03(+)	3.14E+03(+)	4.99E+03(+)	2.59E+03
F23	5.30E+03(+)	6.77E+03(+)	6.95E+03(+)	5.01E+03(+)	6.07E+03(+)	**4.90E+03**
F24	2.88E+02(+)	2.78E+02(–)	5.21E+02(+)	**2.76E+02(–)**	2.81E+02(=)	2.82E+02
F25	3.07E+02(+)	2.88E+02(=)	4.20E+02(+)	**2.83E+02**(–)	2.95E+02(+)	2.88E+02
F26	2.01E+02(–)	2.04E+02(–)	4.40E+02(+)	2.90E+02(+)	**2.00E+02**(–)	2.13E+02
F27	**7.99E+02**(–)	1.03E+03(–)	1.73E+03(+)	1.15E+03(+)	9.85E+02(–)	1.12E+03
F28	4.07E+02(+)	3.00E+02(=)	6.76E+03(+)	6.60E+02(+)	3.00E+02(=)	3.00E+02
+/=/-	20/1/7	15/3/10	23/1/4	21/2/5	17/3/8	

4.2 Simulation and comparison

To verify the effectiveness of CSDE, we compared it with other five algorithms, including some original algorithm (CS, DE, PSO), and their variants (HCS, CSPSO). For each algorithm, the simulation result is shown in Table 2, which is about the mean error between the actual optimal value and the theoretical optimal value obtained by each function.

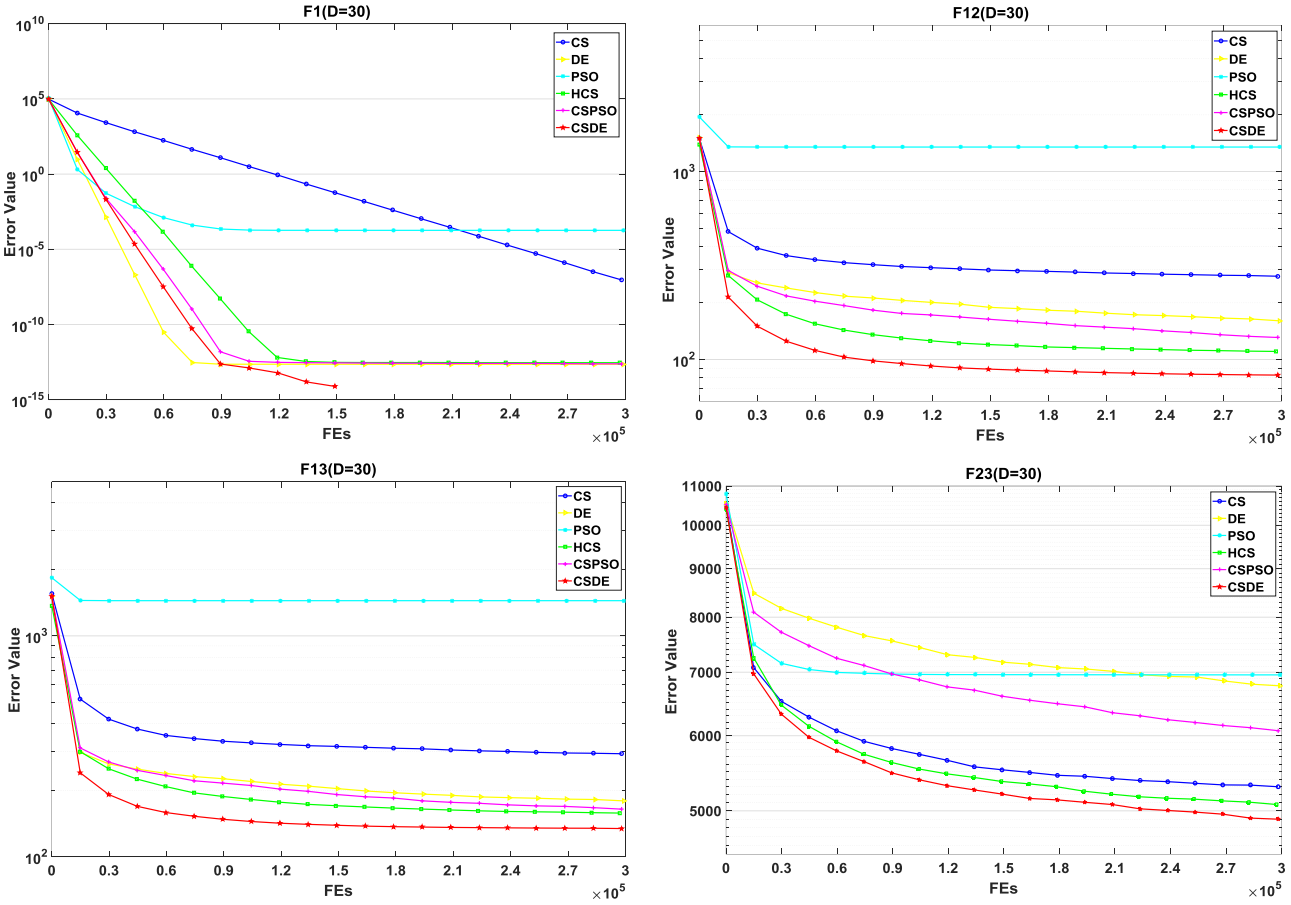

Figure 1: Convergence graph for all compared algorithms on 30-dimensional CEC 2013 benchmark functions F1, F12, F13 and F23.

For unimodal functions F1-F5, CSDE achieves better optimization results in three functions, especially on F1, the mean error is 0 means that the presented algorithm converges to the global optimum. For multimodal functions F6-F20, CSDE can search for better solutions compared to other competing algorithms on 6 out of 15 cases, although DE has a clear advantage on F11. For composition functions F21-F25, the proposed CSDE gets the lowest mean error value on F23, and exhibits similar effort on the remaining functions. Overall, CSDE performs well in unimodal and multimodal functions, but it's slightly insufficient in combination functions. In terms of the symbol "+/=/–", DSCS surpasses CS, DE, PSO, HCS and CSPSO in 20, 15, 23, 21 and 17 functions.

Besides, the convergence graphs of four 30-dimensional CEC 2013 benchmark functions, namely F1, F12, F13 and F23, are respectively plotted in Figure 1. CSDE can find the best solution on function F1. For F12 and F13, the improved variant achieves better search ability and faster convergence compared to other algorithms. The convergence speed of CSDE is also faster than others

on F23.

For statistical comparison, the Friedman test [25] is used to evaluate the performance differences between CSDE and other mentioned algorithms. The results are displayed in Table 3. We can see that CSDE algorithm outperforms its competitors with the smallest ranking value. Therefore, CSDE is competitive in dealing with numerical optimization problem.

Table 3: Experimental results achieved by Friedman test for six algorithms.

Algorithm	Ranking	Algorithm	Ranking
CS	3.73	HCS	3.63
DE	3.36	CSPSO	2.91
PSO	4.95	**CSDE**	**2.43**

5 Conclusions

Based on the differential evolution strategy, a new modified CS (CSDE) is proposed in this paper. CSDE improves performance mainly from two aspects. One is to introduce the mutation strategy indirectly by judging the optimization state in LFRW, and it's beneficial to enhance the global search. The other is that the self-adaptive Cr is adopted to control the cross-selection strategies in BSRW, which is good for getting out of a local extremum. Extensive experiments are conducted based on 28 benchmark functions from CEC 2013, and simulation results verify that CSDE is superior to other methods mentioned above for most cases. In the future, we consider introducing mutation strategy into the optimization of multi-objective problems.

Acknowledgements

This work is supported by the International Science and Technology Cooperation Program of China (No. 2015DFR11050), the Applied Science and Technology Research and Development Special Fund Project of Guangdong Province of China (No. 2016B010126004), and the External Cooperation Program of Guangdong Province of China (No. 2013B051000060) .

References

[1] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Engineering with computers, 29(1), 17-35.

[2] Hussain, K., Salleh, M. N. M., Cheng, S., & Shi, Y. (2019). Metaheuristic research: a comprehensive survey. Artificial Intelligence Review, 52(4), 2191-2233.

[3] Wang, D., Tan, D., & Liu, L. (2018). Particle swarm optimization algorithm: an overview. Soft Computing, 22(2), 387-408.

[4] Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), 341-359.

[5] Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution–an updated survey. Swarm and Evolutionary Computation, 27, 1-30.

[6] Wu, X., & Che, A. (2019). A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. Omega, 82, 155-165.

[7] Eberhart, R., & Kennedy, J. (1995, November). Particle swarm optimization. In Proceedings of the IEEE international conference on neural networks (Vol. 4, pp. 1942-1948).

[8] Kennedy, J. (2010). Particle swarm optimization. Encyclopedia of machine learning, 760-766.

[9] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39(3), 459-471.

[10] Yang, X. S. (2009, October). Firefly algorithms for multimodal optimization. In International symposium on stochastic algorithms (pp. 169-178). Springer, Berlin, Heidelberg.

[11] Yang, X. S., & Deb, S. (2009, December). Cuckoo search via Lévy flights. In 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC) (pp. 210-214). IEEE.

[12] Bhandari, A. K., & Maurya, S. (2019). Cuckoo search algorithm-based brightness preserving histogram scheme for low-contrast image enhancement. Soft Computing, 1-27.

[13] Boushaki, S. I., Kamel, N., & Bendjeghaba, O. (2018). A new quantum chaotic cuckoo search algorithm for data clustering. Expert Systems with Applications, 96, 358-372.

[14] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Engineering with computers, 29(1), 17-35.

[15] Naik, M. K., & Panda, R. (2016). A novel adaptive cuckoo search algorithm for intrinsic discriminant analysis based face recognition. Applied Soft Computing, 38, 661-675.

[16] Ma, H. S., Li, S. X., Li, S. F., Lv, Z. N., & Wang, J. S. (2019). An improved dynamic self-adaption cuckoo search algorithm based on collaboration between subpopulations. Neural Computing and Applications, 31(5), 1375-1389.

[17] Blum, C., Puchinger, J., Raidl, G. R., & Roli, A. (2011). Hybrid metaheuristics in combinatorial optimization: A survey. Applied Soft Computing, 11(6), 4135-4151.

[18] Hassan, A., & Pillay, N. (2019). Hybrid metaheuristics: An automated approach. Expert Systems with Applications, 130, 132-144.

[19] Aydilek, I. B. (2018). A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Applied Soft Computing, 66, 232-249.

[20] Reddy, S. S. (2019). Optimal power flow using hybrid differential evolution and harmony search algorithm. International Journal of Machine Learning and Cybernetics, 10(5), 1077-1091.

[21] Wang, G. G., Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2016). A new hybrid method based on krill herd and cuckoo search for global optimisation tasks. International Journal of Bio-Inspired Computation, 8(5), 286-299.

[22] Chi, R., Su, Y. X., Zhang, D. H., Chi, X. X., & Zhang, H. J. (2019). A hybridization of cuckoo search and particle swarm optimization for solving optimization problems. Neural Computing and Applications, 31(1), 653-670.

[23] Wei, J., & Yu, Y. (2017). An effective hybrid cuckoo search algorithm for unknown parameters and time delays estimation of chaotic systems. IEEE Access, 6, 6560-6571.

[24] Liang, J. J., Qu, B. Y., Suganthan, P. N., & Hernández-Díaz, A. G. (2013). Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 201212(34), 281-295.

[25] Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the american statistical association, 32(200), 675-701.