TELKOMNIKA, Vol.15, No.1, March 2017, pp. 93~100
ISSN: 1693-6930, accredited **A** by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v15i1.3862

Research on Chaotic Firefly Algorithm and the Application in Optimal Reactive Power Dispatch

Genfu Xiao[1], Huan Liu*[2], Yanhui Zhou[3], Yuming Guo[4]
[1,3,4]Institute of Mechanical and Electrical Technology, Jinggangshan University, Ji'an, China
[1,2]Laboratory of watershed ecology and geographical environment monitoring, NASG, Ji'an, China
[2]Institute of Electronic and Information Engineering, Jinggangshan University, Ji'an, China
*Corresponding author, e-mail: liuhuan816618@163.com

Abstract

Firefly algorithm (FA) is a newly proposed swarm intelligence optimization algorithm. Like many other general swarm intelligence optimization algorithms, the original version of FA is easy to trap into local optima. In order to overcome this drawback, the chaotic firefly algorithm (CFA) is proposed. The methods of chaos initialization, chaos population regeneration and linear decreasing inertia weight have been introduced into the original version of FA so as to increase its global search mobility for robust global optimization. The CFA is calculated in Matlab and is examined on six benchmark functions. In order to evaluate the engineering application of the algorithm, the reactive power optimization problem in IEEE 30 bus system is solved by CFA. The outcomes show that the CFA has better performance compared to the original version of FA and PSO.

Keywords: *chaos, FA, optimization, reactive power*

1. Introduction

Metaheuristic techniques are well-known global optimization methods. These techniques attempt to mimic some characteristics of natural phenomena or social behavior [1, 2]. Metaheuristic algorithms are simple and easy to be implemented. These algorithms can carry out the parallel process, and have strong robustness [3, 4].

A lot of new metaheuristic algorithms are proposed for global search. The novel algorithms can solve larger scale problems and increase the computational efficiency [5]. Firefly algorithm (FA), one of new metaheuristic algorithms, was developed for multimodal optimization by Yang in 2008 [6]. The FA is based on the idealized behavior of the flashing characteristics of fireflies. FA has many similarities with particle swarm optimization (PSO) algorithm, and it is also based on swarm behavior. Compared with other swarm intelligence algorithms, FA has good efficiency and robustness in complex optimization problems. Yang has solved reactive power optimization problem with original version of FA and achieved good results [7]. Rajan has proposed a novel hybrid algorithm combining firefly algorithm and Nelder-Mead simplex method for solving power system optimal reactive power dispatch problems [8].

However, metaheuristic algorithms are usually easy to fall into local optimum. To overcome this shortcoming, chaotic functions are added to the metaheuristic algorithms, such as particle swarm optimization [9, 10], genetic algorithm [11], bee colony optimization [12], harmony search [13], imperialist competitive algorithm[14], differential evolution algorithm [15], simulated annealing [16], and so on.

In this paper, the chaotic firefly algorithm (CFA) is proposed to avoid falling into local optimum. In CFA, the strategies of chaos initialization, chaos population regeneration and linear decreasing inertia weight are introduced. In order to evaluate CFA, six benchmark functions are utilized. In the meantime, CFA is applied to solve highly nonlinear non convex optimal reactive power dispatch problem of power systems.

The rest of this paper is organized as follows. Section 2 discusses the descriptions of CFA. Section 3 describes the benchmark function calculation with CFA. Section 4 analyses reactive power optimization based on CFA, and Section 5 is the conclusions of this paper and the outlines for the further research.

Received October 11, 2016; Revised December 23, 2016; Accepted January 11, 2017

2. Chaotic Firefly Algorithm
2.1. Original Version of FA
The firefly algorithm is a metaheuristic algorithm inspired by flashing behavior of fireflies. A firefly will move toward a brighter firefly by reducing the distance between itself and the brighter one.

The intensity *I* of the firefly is proportional to their brightness. The firefly with less brightness will be attracted by the brighter one. The intensity *I* can be expressed as follow:

$$I = I_0 e^{-\gamma r_{ij}} \tag{1}$$

Where γ represents light absorption coefficient, I_0 is the original light intensity, r_{ij} represents the distance between two fireflies.

$$r_{ij} = \|x_i - x_j\|_2 = \sqrt{\sum_{k=1}^{\dim}(x_{ik} - x_{jk})^2} \tag{2}$$

Where *dim* is the dimension of x_i, x_{ik} is the *k*th component of x_i.

The attractiveness of a firefly is proportional to the light intensity. The attractiveness β of a firefly is defined as follow:

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

Where β_0 is the attractiveness in r=0.

The new position of a firefly is calculated by:

$$x_i^{t+1} = x_i^t + \beta(x_j^t - x_i^t) + \alpha^t \varepsilon_i^t \tag{4}$$

Where x_i^{t+1} is the position of the *i*th firefly at time t+1, α^t is the step length, ε_i^t represents a random number.

2.2. The Chaos Improvement of FA
Chaos is defined as an irregular motion, seemingly unpredictable random behavior exhibited by a deterministic nonlinear system under deterministic conditions. A chaos map is a map that exhibits some sort of chaotic behavior. In order to improve the global optimization performance of FA, chaos map has been increased into FA.

2.2.1. Chaos Initialization
The position of initial population has a great influence on the convergence and search efficiency of FA. The original version of FA initializes the position of the fireflies in random way, which may lead to uneven distribution of the fireflies.

In order to enhance the global searching ability of the algorithm, the position of fireflies is initialized by chaos map. By the way of chaos initializing the fireflies' position, the randomness of FA hasn't be changed, but both the diversity of the swarm and the ergodicity of search are improved.

The logistic map is utilized to generate chaotic sets, which is represented by the following equation.

$$v_{k+1} = \mu \quad v_k(1 - v_k), \quad 0 \leqslant v_1 \leqslant 1 \tag{5}$$

In this equation, v_k is the *k*th chaotic variable, with *k* denoting the iteration number. μ =4 is used for the experiments. It is needed to explain that the chaotic variables must be mapped to the search space of the solution.

2.2.2. Chaos Population Regeneration

In order to keep the diversity of the population and to enhance the dispersion of the search, the new algorithm adopts chaos to produce new fireflies to replace the low brightness fireflies in the original population. The threshold number of iteration G_t is set and the global search is started. Local search is performed around the current optimal solution, after the iteration of G_t times. N_f new fireflies is created through the chaotic model shown in formula (7), and they will replace the equivalent low brightness fireflies.

$$\begin{cases} p_1 = p_{gbest} + c \cdot rand(\dim) \\ p_{j+1} = \mu \cdot p_j (1 - p_j) \end{cases} \qquad 1 \leqslant j \leqslant N_j \mathbf{-1} \tag{6}$$

Where, p_j is the position of the *j*th new fireflies in the chaos space, p_{gbest} is the position of the global optimal firefly in the chaos space, c is a constant, N_f is the number of new fireflies, *dim* is the dimension of fireflies.

As section 2.2.1, the variables in the chaos space must be mapped to the solution space.

New fireflies scattered around the global best position, which can balance the global search and the local search. It greatly improves the convergence speed of the algorithm and the ability for finding more accurate solution.

2.2.3. Linear Decreasing Inertia Weight

The linear decreasing weight is added into the formula (4) as follow:

$$\begin{cases} x_i^{t+1} = \omega^t x_i^t + \beta(x_j^t - x_i^t) + \alpha^t \varepsilon_i^t \\ \omega^t = \omega_{\max} - (\omega_{\max} - \omega_{\min}) \cdot t / G_{Max} \end{cases} \tag{7}$$

Where $\omega_{\max}$ is the maximal weight and $\omega_{\min}$ is the minimal weight, G_{Max} is maximum number of iterations.

The inertia weight ω^t reflects the influence of the previous position to the current position, and the weight affects the balance of the global and local search ability of the optimization algorithm. Larger weight can enhance the global search capability. Smaller weight can enhance the local search ability. By using the linear decreasing inertia weight, in the early stage the firefly algorithm has stronger global searching ability and the convergence speed is improved, in the late stage the firefly algorithm has stronger local searching ability and local oscillation is avoided.

2.3. Algorithm Description

The new algorithm is proposed which is called as chaotic firefly algorithm (CFA). The procedure of the new algorithm can be described as follows:

Step 1, the attractiveness β_0, light absorption coefficient γ, the step length α and maximum number of iterations G_{Max} are initialized.

Step 2, the position of fireflies is initialized by chaos mapping.

Step 3, the attractiveness of fireflies is calculated by formula (3).

Step 4, the position of fireflies is updated by formula (7).

Step 5, if the number of iterations is equal to integer multiple of G_t, new fireflies are produced by formula (6) to replace the low brightness fireflies, and the brightness of new fireflies is calculated.

Step 6, if the algorithm satisfies the stopping criterion, go to step 7; otherwise, go to step 3.

Step 7, x_{gbest} is output and the end of algorithm is reached.

3. Implementation

Six benchmark functions shown in Table 1 are selected to evaluate the CFA. The parameters n=40, α=0.5, γ=1 and G_{Max}=300 are adopted in FA, where n is the fireflies population size. CFA and FA use the same parameters. The cost functions of firefly algorithm are taken to the six benchmark functions. In the case of CFA, the threshold number of iteration G_t=10, the number of new fireflies N_f=4. In addition, the standard version of PSO algorithm is used.

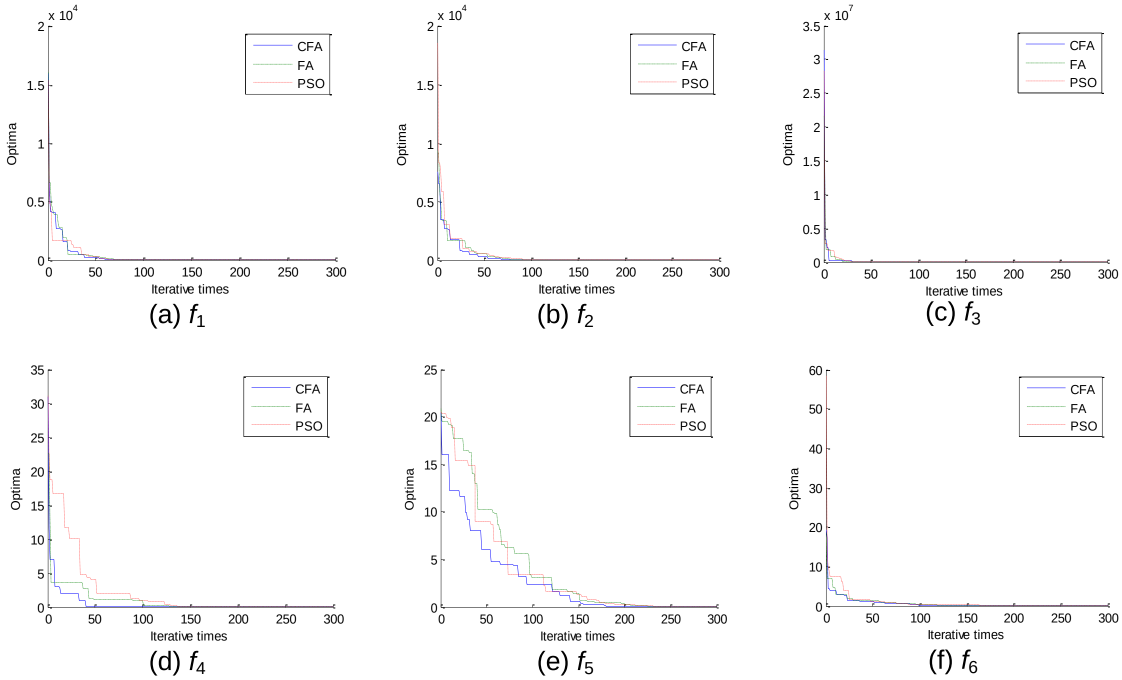
Table 1. Six Benchmark Functions

Function	Name	Dimensions	Range	Minimum
$f_1 = \sum_{i=1}^{n} x_i^2$	Sphere	10	[-100,100]	0
$f_2 = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	Quadric	10	[-100,100]	0
$f_3 = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	Rosenbrok	10	[-30,30]	0
$f_4 = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10$	Ranstrigin	10	[-5.12,5.12]	0
$f_5 = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	Ackley	10	[-100,100]	0
$f_6 = \sum_{i=1}^{n} x_i^2 \big/ 4000 - \prod_{i=1}^{n} \cos(x_i / \sqrt{i}) + 1$	Griewank	10	[-600,600]	0

Table 2. The results of different methods

Methods	f_1 Best	f_1 Worst	f_1 Average	f_1 Std.Dev	f_2 Best	f_2 Worst	f_2 Average	f_2 Std.Dev
PSO	1.73e-6	2.82e-4	6.5e-5	9.95e-5	6.63e-5	8.81e-4	2.41e-4	3.06e-4
FA	1.22e-6	2.31e-4	4.95e-5	8.34e-5	2.88e-4	9.7e-4	5.46e-4	2.52e-4
CFA	4.3e-7	1.65e-5	2.53e-6	5.45e-6	1.87e-5	1.68e-4	5.14e-5	5.66e-5
Methods	f_3 Best	f_3 Worst	f_3 Average	f_3 Std.Dev	f_4 Best	f_4 Worst	f_4 Average	f_4 Std.Dev
PSO	0.0098	0.0193	0.0148	0.0028	1.64e-5	8.64e-4	3.48e-4	2.61e-4
FA	0.0017	0.0099	0.0053	0.0031	1.89e-5	9.89e-4	2.75e-4	3.75e-4
CFA	4.54e-4	7.84e-4	5.58e-4	1.15e-4	5.80e-6	2.80e-4	8.16e-5	9.39e-5
Methods	f_5 Best	f_5 Worst	f_5 Average	f_5 Std.Dev	f_6 Best	f_6 Worst	f_6 Average	f_6 Std.Dev
PSO	0.0691	0.1775	0.1117	0.0329	0.0027	0.0106	0.0071	0.0028
FA	0.0764	0.1843	0.1251	0.0234	0.0038	0.0879	0.0175	0.0234
CFA	0.0075	0.0224	0.0102	0.0051	5.64e-5	0.0068	0.0033	0.0025

The calculation results of these functions are shown in Table 2. The convergence characteristics of f_1-f_6 using different algorithms are shown in Figure 1. Because the chaos initialization and chaos population regeneration increase the population diversity and enhance the global searching ability, and linear decreasing inertia weight helps to speed up global convergence speed, CFA shows good performance. As shown in the Table 2 and Figure 1, the optima value, average value and standard deviation found by CFA are significantly better than FA and PSO. CFA can find the optimal solution faster than FA and PSO.



(a) f_1 (b) f_2 (c) f_3

(d) f_4 (e) f_5 (f) f_6

Figure 1. The Convergence Curves of f_1~f_6

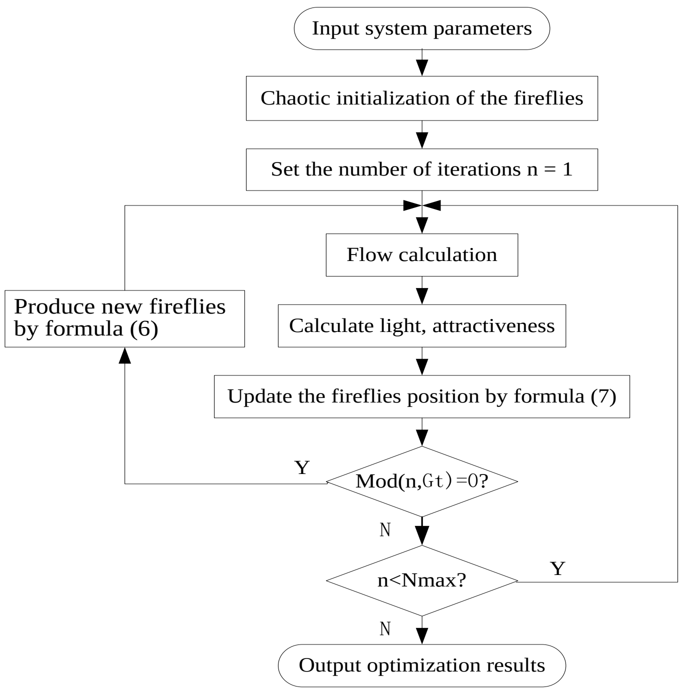


Figure 2. The Flow Chart of Reactive Power Optimization

4. Reactive Power Optimization Based on CFA

4.1. Math Models of Reactive Power Optimization

In the premise of safety, the ultimate goal of reactive power optimization of power system is to reduce the active power loss by adjusting the control variables, and the mathematical model includes the objective function, the power flow equation and variable constraint.

The minimum power loss is treated as the objective function, which is expressed as follow:

$$\min(P_{loss}) = \min \sum_{i,j \in N_L} G_{ij}[U_i^2 + U_j^2 - 2U_iU_j \cos\theta_{ij}] \quad (8)$$

Where, P_{loss} is the active power loss in the power system. U_i, U_j are the voltages of ith and jth buses respectively. θ_{ij} is the voltage angle difference between bus i and j. N_L is the number of transmission lines. G_{ij} is the conductance.

4.2. Calculation Flow

Based on the CFA described in section 2.3, the flow chart of reactive power optimization is shown in Figure 2.

The power system parameters and variables are described in detail as follows:

(1) The power system parameters read from database are shown as follows: generator parameters, load parameters, lines and transformers parameters, reactive power compensation parameters, the constraints of various control variables.

(2) The power system control variables include generator voltage V_{Gi}, amount of reactive power compensation Q_{ci}, transformers ratio K_{Ti}. The components of fireflies are made up of these control variables, and the dimension of fireflies is the total number of control variables, namely:

$$x_i = [V_{G1}, \cdots, V_{GN_G}, Q_{C1} \cdots Q_{CN_C}, K_{T1}, \cdots K_{TN_T}]^T$$

4.3. Results of Reactive Power Optimization

In order to verify the superiority of CFA in reactive power optimization, this paper uses CFA to optimize the IEEE 30 nodes system, and the results are compared with FA and PSO algorithm. The IEEE 30 nodes system is made up of 41 transmission lines, 6 generators, 2 shunt capacitors and 4 transformers. The power reference value is set to 100MW. The optimization program is completed in Matlab.

The results of reactive power optimization in the IEEE 30 nodes system are shown in Table 3. In virtue of optimization, the control variables are selected reasonably and the active power loss has reduced. The active power loss calculated by CFA is smaller than that calculated by FA and PSO. The optimal algorithm convergence characteristic is shown in figure 3. The figure shows that CFA isn't easy to be trapped in local optimum and the convergence speed of CFA is faster than that of FA and PSO.

Table 3. Optimization Results of the IEEE 30 Nodes System

Variables	Variable names	Original values	Algorithms		
			PSO	FA	CFA
Generator voltage	U1	1	1.0291	1.0213	1.05
	U2	1	1.0269	1.0197	1.049
	U22	1	1.0288	1.0140	1.02523
	U27	1	1.0493	1.032	1.0451
	U23	1	1.0386	1.0233	1.0349
	U13	1	1.05	1.0499	1.05
Compensation capacitors	Q10	0	0.1851	0.1341	0.3231
	Q24	0.04	0.1043	0.1134	0.3150
Transformer ratio	T6-9	0	0.9646	0.9735	0.9929
	T6-10	0	1.0282	0.9944	1.0326
	T4-12	0	0.9904	1.0312	1.0315
	T28-27	0	1.0049	1.0282	1.05
Power loss	Ploss	2.4438	2.1456	2.1368	2.0693

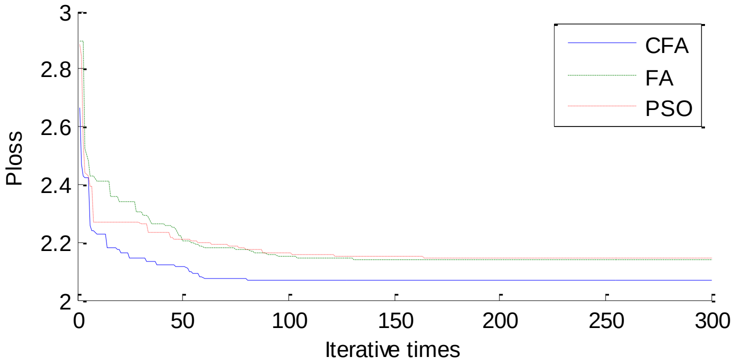


Figure 3. The Convergence Curves Of Reactive Power Optimization

5. Discussions and Conclusions

In the present work, a novel firefly algorithm adding chaos map is proposed. It is verified that by using the proposed approach, the performance of the original firefly algorithm can apparently be improved.

There are three new improved methods in CFA. The method of chaos initialization may improve the uniformity of the position distribution of fireflies. The method of chaos population regeneration can maintain the diversity of the population and reduced the possibility of falling into local optimum. In addition, the method of linear decreasing inertia weight helps to speed up the global convergence speed, reduce the oscillation near the extreme point, and improve the search ability near the extreme point. The experimental results show that CFA is superior to the original version of FA and PSO in both speed and accuracy. The capability and robustness of CFA make it suitable to solve complex optimization problems like optimal reactive power dispatch.

Furthermore studies on convergence analysis can be fruitful. It can be expected that for different types of problem, different chaotic maps may have different convergence rates. In addition, further topic of studies can also focus on the extension of the CFA to solve discrete problems and mixed-type optimization problems.

Acknowledgements

This work was supported by the doctoral research project of Jinggangshan university (Nos. JZB15009, JZB15016), the science & technology project of Ji’an city (20151016), the project of humanities & social sciences in universities of Jiangxi province (YS1546), the natural science foundation of Jiangxi province (Nos. 20151BAB217012, 2016BAB202049, 20161BAB206135), the science & technology project of Jiangxi education department (GJJ150787), the bidding project of the key laboratory of watershed ecology and geographical environment monitoring, NASG (Nos. WE2016013, WE2016015) and the national natural science foundation of China (Nos.51465020, 61462046).

References

[1] XS Yang, S Deb, T Hanne, X He. Attraction and diffusion in nature-inspired optimization algorithms. *Neural comput & Applic*. 2015; (5): 1-8.
[2] Dinda Novitasari, Imam Cholissodin. Hybridizing PSO with SA for Optimizing SVR Applied to Software Effort Estimation. *TELKOMNIKA*. 2016; 14(1): 245-253.
[3] A Baykasoglu, FB Ozsoydan. Adaptive firefly algorithm with chaos for mechanical design optimization problems. *Applied Soft Computing*. 2015; 36: 152-164.
[4] AH Gandomi, XS Yang, S Talatahari. Firefly algorithm with chaos. *Communications in Nonlinear Science & Numerical Simulation*. 2013; 18(1): 89–98.
[5] A Kaveh, N Farhoudi. Dolphin monitoring for enhancing metaheuristic algorithms: Layout optimization of braced frames. *Computers & Structures*. 2016; 165: 1-9.
[6] Yang XS. Nature-inspired metaheuristic algorithms. Luniver Press. 2008.
[7] Yang Xin She, Sadat Hosseini SS, Hossein Gandomi Amir. Firefly algorithm for solving non-convex economic dispatch problem with valve point loading effect. *Appl Soft Comput*. 2012; 12(3): 1180-1186.

[8] Abhishek Rajan, T Malakar. Optimal reactive power dispatch using hybrid Nelder–Mead simplex based firefly algorithm. *Electrical power and energy systems*. 2015; 66(66): 9-24.
[9] Yalin Wu, Shuiping Zhang. A self-adaptive chaos particle swarm optimization algorithm. *TELKOMNIKA*. 2015; 13(1): 331-340.
[10] Y Wu, G Liu, X Guo, Y Shi, L Xie. A self-adaptive chaos and Kalman filter-based particle swarm optimization for economic dispatch problem. *Soft Computing*. 2016; (1): 1-13.
[11] Şengul Dogan. A new data hiding method based on chaos embedded genetic algorithm for color image. *Artificial Intelligence Review*. 2016; 46(1): 1-15.
[12] Alatas B. Active power loss minimization in electric power systems through chaotic artificial bee colony algorithm. *Tehnicki Vjesnik*. 2016; 23(2): 491-498.
[13] Xiaofang Yuan, Jingyi Zhao. Hybrid parallel chaos optimization algorithm with harmony search algorithm. *Applied Soft Computing*. 2014; 17(4): 12-22.
[14] Talatahari S, Farahmand Azar B, Sheikholeslami R, Gandomi AH. Imperialist competitive algorithm combined with chaos for global optimization. *Communications in Nonlinear Science & Numerical Simulation*. 2012; 17(3): 1312-1319.
[15] TN Malik, S Zafar, S Haroon. Short-term economic emission power scheduling of hydrothermal systems using improved chaotic hybrid differential evolution. *Turkish Journal of Electrical Engineering and Computer Sciences*. 2016; 24(4): 2654-2670.
[16] X Wang, C Liu, D Xu, C Liu. Image encryption scheme using chaos and simulated annealing algorithm. *Nonlinear Dynamics*. 2016. 84(3): 1417-1429.