MATEC Web of Conferences **40**, 09009 (2016)
DOI: 10.1051/matecconf/20164009009

A Cuckoo Search Algorithm with Complex Local Search Method for Solving Engineering Structural Optimization Problem

Chiwen Qu ,Wei He

School of Information Engineering ,Baise University,Baise,China

Abstract. The standard cuckoo search algorithm is of low accuracy and easy to fall into local optimal value in the later evolution. In this paper, an improved cuckoo algorithm is proposed. Dynamic change of parameter of probability is introduced to improve the convergence speed. Complex method is quoted to improve the capabilities of local search algorithm. A non-fixed multi-segment mapping penalty function is adopted to realize constraint processing algorithms. The results of the optimization problem constrained by standard test functions and two engineering design show that this algorithm is effective for solving constrained optimization problems and suitable for engineering design and other constrained optimization problems.

1 Introduction

In the field of engineering applications, solving constrained optimization problems is a class of mathematical programming problems frequently encountered. Due to limited resources, engineering design optimization problems are often bound by many non-linear constraints. Therefore, it is of great importance to research constrained engineering optimal questions. Generally, a nonlinear constrained optimization problem can be described as:

$$\begin{aligned} &\min\ f(x) \\ \text{s.t.}\ &g_j(x) \le 0, j = 1,2,...,p \\ &h_j(x) = 0, j = p+1, p+2,...,m \\ &l_i \le x_i \le u_i, i = 1,2,....,n \end{aligned} \tag{1}$$

where $f(x)$ is the objective function, x is n-dimensioned vector which represents the design variables, $x = (x_1, x_2, ..., x_n)$, $g_j(x) \le 0$ and $h_j(x) = 0$ are respectively the problem of the j-th inequality constraints and the j-th equality constraints, l_i and u_i are the corresponding minimum and maximum values, respectively. Because the constraints of the problem (1) are usually nonlinear, it is much more complex to solve the problem than unconstrained optimization problems. The traditional way to solve the problem usually is based on deterministic gradient search method, such as the penalty function method, gradient projection method and so on. However, it requires the conditions of the problem and objective function are continuously differentiable, and the gradient search method can easily fall into local optima. Therefore, when the problem domain constraints and objective function are non-differentiable, this method is powerless. Compared with the constrained optimization problems in traditional way, intelligent swarm algorithms combined with appropriate constraint-handling technique do not need gradient information. Greater probability can converge to the global optimum value. Therefore, using intelligent swarm algorithms to solve constrained optimization problems are more extensive and applicable, and it is becoming one of the hotspots of evolution in computing research.

As a heuristic intelligent swarm algorithm, cuckoo search algorithm is presented by the British scholar Yang and Deb, which based on the spawning habits of the cuckoo in the nest and search process in 2009 [1]. The algorithm is simple and few parameters need to be set. Furthermore, optimization accuracy and the rate of convergence are better than PSO and Genetic algorithm. It has been widely used in solving constrained optimization problems [2]. Yang and Deb et al. proposed a multi-objective optimization cuckoo search algorithm for solving engineering design of multi-objective optimization problem [3]. Gandomt proposed a prime cuckoo search algorithm for solving optimization problems [4]. The two algorithms are merely using the basic cuckoo search algorithm for solving constrained optimization problems. However, in the presence of the low search accuracy and slow convergence in later evolutionary stages, the cuckoo search algorithm and other search algorithms are the same as intelligent swarm algorithm. Some researchers use the modified cuckoo optimization algorithm to solve constrained optimization problems. Wen Long and Ximing Liang proposed a mixing cuckoo algorithm which combined Solis & Wets local search with augmented Lagrangian method for solving constrained optimization problems [5]. Radovan R.Bulatovic et al. proposed an improved cuckoo algorithm for solving constrained optimization problems

[6], which dynamically changes the walk step of the basic cuckoo search algorithm and detection probability of the nests, and achieves better results. But these modified cuckoo algorithms do not consider the selection of punishment factor and uneven distribution of the initial population in the solution space, and affects the convergence and stability of the algorithm.

In this paper, an improved cuckoo search (ICS) algorithm is proposed. The simulation results show, compared with the basic cuckoo algorithm and some other intelligent swarm algorithms, the ICS optimization algorithm of this article has higher accuracy and stability.

2. Cuckoo search algorithm

The cuckoo search is a metaheuristic algorithm, which imitates the cuckoos' manner of looking for suitable nests for egg hatching. The basic principle is as follows. (1) The parasitic nests of cuckoo parasitic eggs correspond to a solution in the search space. (2) Each parasitic nest location corresponds to a fitness value of the algorithm. (3) The walk process of cuckoos' manner of looking for parasitic nests map to the intelligent optimization process of the algorithm.

The new suitable parasitic nest is generated according to the following law

$$Nest_i^{t+1} = Nest_i^t + \alpha \oplus levy(\lambda) \tag{2}$$

where $Nest_i^t$ is the location of the t generation of the i-th parasitic nest, and α is the step size value depends on the optimization problem. In most cases, α can be set to the value of 1.The product $\oplus$ means entry-wise multiplication. The random step size is multiplied by the random numbers with Lévy distribution, which according to the following probability distribution

$$Levy \sim u = t^{-\lambda} \tag{3}$$

where t is step size drawn from a Levy distribution. After the location update, the egg laid by a cuckoo can be spotted by the host of the parasitic nest with a probability $p_a = [0,1]$. For such incidents, the host of the parasitic nest abandon the nest and seek for a new site to rebuild the nest.

Based on these rules which described above, the steps involved in the computation of the standard cuckoo search algorithm are presented in Algorithm 1.

Algorithm 1: The standard cuckoo search algorithm

1.Objective function
2. Generate initial population of n host of the parasitic nests
3.Generation iter=1,define probability , set walk step length
4. Evaluate the fitness function
5. while (iter < MaxGeneration) or (stop criterion)
6. Get a cuckoo egg from random host of the parasitic nest by Levy flight
7. Evaluate the fitness function
8. Choose a parasitic nest i among n host parasitic nests, and Evaluate the fitness function
9. If() then
10. Replace with
11. End if
12. A fraction () of the worse parasitic nests are evicted
13. Built new parasitic nests randomly to replace nests lost
14. Evaluate the new parasitic nests' fitness
15. Keep the best solutions
16. Rank the solutions
17. Update the generation number iter= iter+1
18.End while
19.Output the best solutions

3. The cuckoo search algorithm with complex local search method

The basic cuckoo algorithm uses Levy flight to update the location of parasitic nests. The update mode of Levy flight is essentially based on Markov chain methods. The destination of parasitic nest location update is determined by the current parasitic nest location and transition probabilities. Therefore, the way to update the position is blindness. At the same time, the probability p_a of parasitic nest is determined, leading to a slow convergence speed and an insufficient local search in the late evolution.

3.1 Dynamic change of parameter of probability

Although cuckoo algorithms have strong global search capability, the probability p_a of parasitic nest is determined, and the parasitic nests will be replaced with the same probability whether they are in poor positions or in the optimum positions. If the value p_a is too small, the solutions converge slowly. If the value is too big, the solutions are difficult to converge to the optimal ones. To improve the convergence speed and solution precision, we use a dynamic probability discovery mechanism shown in Eq. (4). At the beginning of the algorithm, a larger position change rate is need due to the large distance between the individuals and the optimal values. In the late evolution, a smaller position change rate is need owing to the fact that most individuals gather around the optimal position.

$$p_a(t) = (p_{a_{\max}} - p_{a_{\min}}) \cdot \frac{t_{\max} - t}{t_{\max}} + p_{a_{\min}} \tag{4}$$

where $p_a(t)$ is the finding probability of the t-th iteration, $p_{a_{\max}}$ and $p_{a_{\min}}$ are the minimum and maximum probability, $t_{\max}$ is the maximum number of iterations, t is the current number of iteration.

3.2 Complex local search algorithm

Complex method has been widely used in constrained nonlinear optimization problems due to the small calculated amount and strong local search capability. It achieves movement and shrinkage from eliminated individuals to optimal ones by the operations of elimination, reflection, compression and expansion. The steps of the method are as follows.

Step 1: Initialize compound form. N vertices are randomly generated within the scope of the solution space, $x_1, x_2, ..., x_n$.

Step 2: Select the worst vertexes. Calculate corresponding fitness values for each vertexes of the initial population, and select the worst vertex (x^H) and the best vertex (x^L).

Step 3: Compute the center position ($\bar{x}$) of the vertexes except for the worst one (x^H), which is given by

$$\bar{x} = \frac{1}{n-1}\sum_{j=1}^{n} x^{j}, (j \neq H) \tag{5}$$

Step 4: Search the reflex vertex (x^{r}) of x^{H} according to Eq.(6)

$$x^{r} = \bar{x} + \alpha \cdot (\bar{x} - x^{H}) \tag{6}$$

where α is the reflection coefficient. If the reflex vertex is within the scope of the solution space, the step 5 works. Otherwise, the reflection coefficient α is halved. The algorithm searches for the reflected vertex according to Eq. (6) until x^{r} is in the solution space range. Then the algorithm enters the step 5.

Step 5: Compare the fitness values of x^{r} and x^{H} . Set the fitness values of x^{r} and x^{H} to be $f(x^{r})$ and $f(x^{H})$, respectively. if $f(x^{r}) < f(x^{H})$, the algorithm returns to Step 2 after replacing x^{H} with x^{r} . Otherwise, the reflection coefficient α is halved, and the algorithm returns to Step 4 until $f(x^{r}) < f(x^{H})$.

Step 6: Check the terminating condition. If it is false, the algorithm repeats the above Step 2 to Step 5.

3.3 Constraint handling mechanism

The penalty function method is the most commonly used technology for solving constrained optimization problems. A new objective function can be formed by adding a penalty term which can reflect constraints to the original objective function value. In general, the new form of the structure of the objective function is shown in Eq. (7).

$$f'(x) = f(x) + \eta(t) \cdot H(x) \tag{7}$$

where $f'(x)$ is the construction of a new objective function, $f(x)$ is the original objective function, $\eta(t)$ is the punishment coefficient of penalties, and $H(x)$ is the penalty factor. If $\eta(t)$ is too small, the solutions do not meet the constraint conditions, making the algorithm convergence speed slow. If $\eta(t)$ is too large, the objective function is poor in the boundary of the feasible region. Therefore, this paper handles the constraints by adopting a non-fixed multi-segment mapping penalty function method [7]. The approach is as follows.

$$H(x) = \sum_{s=1}^{p} \theta(q_s(x)) \cdot q_s(x)^{r(q_s(x))}$$

$$\eta(t) = t \cdot \sqrt{t} \tag{8}$$

where t is the number of iterations, $q_s(x)$ is the solution of constraint violation calculated by Eq. (9), $\gamma(q_s(x))$ is the punishment strength using Eq. (10), and $\theta(q_s(x))$ is a multi-segment mapping function.

$$q_s(x) = \begin{cases} \max\{0, g_s(x)\}, (s = 1,2,\ldots,p) \\ \max\{0, |h_j(x)|\}, (s = p+1, p+2,\ldots,m) \end{cases} \tag{9}$$

$$\gamma(q_s(x)) = \begin{cases} 1, q_s(x) < 1 \\ 2, q_s(x) \geq 1 \end{cases} \tag{10}$$

$$\theta(q_s(x)) = \begin{cases} 10, q_s(x) < 0.001 \\ 100, q_s(x) < 0.1 \\ 300, q_s(x) < 1 \\ 1000, other \end{cases} \tag{11}$$

3.4 Algorithm steps

The whole procedure of ICS is described as follows:

Step 1: Initialize the population, and produce n initial parasitic nests $x_i, (i = 1,2,\ldots,n)$ position.

Step 2: Calculate the fitness value of each parasitic nest position using Eqs. (7) - (11) , and acquire the current optimal parasitic nest location and its fitness value.

Step 3: Update the position of the parasitic nest using Eq.(2), and maintain the high fitness value location parasitic nest.

Step 4: Adjust the found probability p_a of the parasitic nest dynamically, and generate a random number $r \in (0,1)$. If $r > p_a$, the location of parasitic nest randomly is changed.

Step 5: Adopted complex method to improve the local search. Using the Eq. (5) to calculate the center of each parasitic nest $\bar{x}_i, (i = 1,2,\ldots,n)$. Using the Eq. (6) to calculate the reflection point of each parasitic nest and its fitness value. If $f(x_i') < f(x_i)$, x_i' is replaced according to certain probability.

Step 6: Maintain the optimal position and the fitness, if the conditions is false, the algorithm goes to Step 7. Otherwise, it goes to Step 3.

Step 7: Output the optimal parasitic nest position and the corresponding fitness values.

4 Numerical experiments and analysis

In this paper , in order to evaluate the effectiveness of the proposed ICS algorithm, 5 constrained optimization problems (namely g01, g04, g05 g06, g07) are considered .The detailed formulation of these functions is shown in [13].To further verify the performances of ICS algorithm, comparisons are carried out with sex typical methods from the literatures, including the standard cuckoo search (CS) [8] algorithm , stochastic ranking(SR)[9] method, simple multimembered evolution strategy (SMES)[10], artificial bee colony(ABC)[11], effective differential evolution with level comparison (DELC)[12], differential evolution with dynamic stochastic selection(DEDS)[13] algorithm. The search results are from the six kinds of algorithm in the corresponding literatures. The following parameters are established experimentally of the ICS: the population size of cuckoo search was set to 100 parasitic nests,

$\lambda = 2$, $p_{a_{\max}=0.95}$, $p_{a_{\min}=0.05}$, $a = 0.4$ .the maximum iteration was 500. In complex from algorithm, the reflection coefficient α was set to 2. For each test function ,20 independent runs are performed in Matlab R2009b.

Table 1 reports the results of the test functions in terms of the mean, the worst, the best and the standard deviation of the objective values of the solutions obtained by ICS and other existing algorithms. As shown in table 1, all the function values reached the theoretical value or are very close to the theoretical values. Compared with other existing algorithms, the proposed algorithm is able to find the global minima in all cases with a very performance.

Table 1 Comparison of ICS with respect to CS,SR,SMES,ABC,DELC and DEDS on 9 benchmark functions

Test functions	Results	Algorithms						
		ICS	CS	SR	SMES	ABC	DELC	DEDS
g01	Best	-15.000	-15.0000	-15.000	-15.000	-15.000	-15.000	-15.000
	Mean	-15.000	-15.0000	-15.000	-15.000	-15.000	-15.000	-15.000
	Worst	-15.000	-14.999	-15.000	-15.000	-15.000	-15.000	-15.000
	S.D.	5.8E-06	3.01E-06	NA	0	0.00E+00	6.5E-15	1.3E-10
g04	Best	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	Mean	-30665.539	-30634.501	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	Worst	-30665.539	-30490.517	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
	S.D.	4.56E-13	5.41E+01	NA	0	2.22E-11	1.0E-11	2.7E-11
g05	Best	5126.4981	5126.4983	5126.497	5126.599	5126.736	5126.498	5126.497
	Mean	5129.8730	5186.1672	5128.881	5174.492	5178.139	5126.498	5126.497
	Worst	5131.4978	5311.5278	5142.472	5304.167	5317.196	5126. 500	5126.497
	S.D.	8.95E+00	6.21E+01	NA	5.0E + 01	5.60E+01	5.1E-04	0
g06	Best	-6961.8140	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814
	Mean	-6961.8140	-6961.814	-6875.940	-6961.814	-6961.814	-6961.814	-6961.814
	Worst	-6961.8140	-6961.814	-6350.262	-6952.482	-6961.814	-6961.814	-6961.814
	S.D.	1.55E-15	1.09E-04	NA	1.9E + 00	0	7.3E-10	0
g07	Best	24.3064	24.3062	24.307	24.327	24.315	24.306	24.306
	Mean	24.5092	24.7987	24.374	24.475	24.415	24.306	24.306
	Worst	24.9603	27.6021	24.642	24.843	24.854	24.306	24.306
	S.D.	1.57E-05	9.95E-01	NA	1.32E-01	1.24E-01	1.5E-06	7.5E-07

5.Application engineering structure design optimization

In this paper, in order to further evaluate the performance of solving the engineering structure design optimization, the proposed ICS algorithm is tested against the following two well-know benchmark design problems.

5.1.Tension-compression spring design

Tension-compression spring design problem is one of the most well-know design benchmark problems. The design optimization problem involves three continuous variables and four nonlinear inequality constraints. This problem has already been solved by many researchers, including He and Wang[14],who proposed hybrid particle swarm optimization (HPSO), Zhang et al.[13],who used differential evolution with dynamic stochastic selection algorithm (DEDS) , Wang and Li[12], who proposed differential evolution with level comparison algorithm(DELC), Sadollah et al. [15], who employed a mine blast algorithm(MBA), Wen et al.[5], who used effective hybrid cuckoo search(HCS - LSAL), and Yang [2], who applied a cuckoo search algorithm(CS).The best solutions obtained by the above mentioned approaches are shown in Table 2.Their statistical results are shown in table 3.

According to the table 2, the optimal value of ICS algorithm is superior to that of other 6 methods. From Table 3, it can be seen that the results in terms of optimal value, average value and the worst by ICS are better than the results by the other 6 kinds of algorithm. The standard deviation of ICS is smaller nine orders of magnitude than those of HPSO, DEDS, MBA, and smaller seven orders of magnitude than DELC and HCS-LSAL.

Table 2 Comparison of the best solution for tension-compression spring design problem by different algorithms

	ICS	HPSO[18]	DEDS[17]	DELC[16]	MBA[19]	HCS-LSAL[6]	CS[2]
x_1	0.05000048	0.051706	0.051689	0.051689	0.051656	0.051689	0.051690
x_2	0.37444332	0.357126	0.356717	0.356717	0.355940	0.356718	0.356750
x_3	8.54625726	11.265083	11.288965	11.288965	11.344665	11.28896	11.287126
$g_1(x)$	-8.39074200E-6	-3.06E-06	1.45E-09	-3.40E-09	0	-6.41E-06	-3.5656491E-5
$g_2(x)$	-2.04040589E-6	1.39E-06	-1.19E-09	2.44E-09	0	-3.90E-06	-0.13424716
$g_3(x)$	-4.86067759E+0	-4.054583	-4.053785	-4.053785	-4.052248	-4.053775	-4.05378705
$g_4(x)$	-7.17037466E-1	-0.727445	-0.727728	-0.727728	-0.728268	-0.727729	-0.72770667
$f(x)$	0.00987263	0.0126652	0.012665	0.012665	0.012665	0.0126652	0.012665

Table 3 Statistical results of different approaches for tension-compression spring design problem

Algorithm	Best	Mean	Worst	Std
ICS	0.00987263	0.00987282	0.00987300	2.00289E-14
HPSO[18]	0.0126652	0.0127072	0.0127190	1.58E-05
DEDS[17]	0.012665233	0.012669366	0.012738262	1.3E-05
DELC[16]	0.012665233	0.012665267	0.012665575	1.3E-07
MBA[19]	0.012665	0.012713	0.012900	6.30E-05
HCS-LSAL[6]	0.0126652	0.0126683	0.0126764	5.37E-07
CS[2]	0.012665	NA	NA	NA

5.2.Pressure vessel design problem

Another well-know engineering optimization task is the design of the pressure vessel for a minimum cost, including the cost of the welding, material and forming. The approaches applied to this problem include six different numerical optimization techniques, a standard cuckoo search algorithm(CS)[20] ,a fish swarm optimization algorithm (FSO)[7], a co-evolutionary particle swarm optimization for constrained optimization tasks (CPSO)[21],a bat algorithm(BA)[22], an effective hybrid cuckoo search algorithm for constrained global optimization (HCS-LSAL) [6], and a hybrid nelder-mead simplex search and particle swarm optimization (NM-PSO)[23]. The results of the various methods for solving the optimal shown in table 4. table 5 shows the statistical results.

It is shown from Table 4 that, the optimal solution solved by ICS algorithm is 5896.504188, which is less than those of the standard CS, FSO, CPSO, BA, HCS-LSAL and NM-PSO algorithms by 163.2101468, 164.573612, 164.573512, 163.2101468, 163.210112 and 33.80951, respec-tively. As can be seen from the statistical results of Table 5, the opti-mum value, the average value and the worst value of the ICS algorithm are better than those of the other 6 algorithms.

Table 4 Comparison of the best solution for pressure vessel design problem by different algorithms

	ICS	CS[20]	FSO[7]	CPSO[21]	BA[22]	HCS-LSAL[6]	NM-PSO[23]
x_1	0.778748	0.8125	0.812500	0.8125	0.8125	0.8125	0.8036
x_2	0.386207	0.4375	0.437500	0.4375	0.4375	0.4375	0.3972
x_3	40.334626	42.0984456	42.09127	42.0913	42.0984456	42.09844	41.6392
x_4	200.000000	176.6365958	176.7466	176.7465	176.6365958	176.6366	182.4120
$g_1(x)$	-2.898467E-04	NA	-0.000139	-1.37E-06	NA	-2.01E-09	3.65E-05
$g_2(x)$	-1.414842E-03	NA	-0.035949	-3.59E-04	NA	-0.035880	3.79E-05
$g_3(x)$	-1.067180E+03	NA	-116.3827	-118.7687	NA	-0.002495	-1.5914
$g_4(x)$	-4.000000E+01	NA	-63.25350	-63.2535	NA	-63.36340	-57.5879
$f(x)$	5896.504188	6059.7143348	6061.0778	6061.0777	6059.7143348	6059.7143	5930.3137

Table 5 Statistical results of different approaches for pressure vessel design problem

Algorithm	Best	Mean	Worst	Std
ICS	5896.504188	5902.026658	5910.686367	23.955
CS[20]	6059.714	6447.736	6447.736	502.693
FSO[7]	6061.0778	NA	NA	NA
CPSO[21]	6061.0777	6147.1332	6363.8041	86.45
BA[22]	6059.71	6179.13	6318.95	137.223
HCS-LSAL[6]	6059.7143	6087.3225	6137.4069	2.21E-02
NM-PSO[23]	5930.3137	5946.7901	5960.0557	9.161

6.Conclusion

This paper presents an improved method for optimizing the cuckoo algorithm to solve constrained optimization problems. A dynamic adaptive change probability of parasitic nest is adopted to increase the convergence of the algorithm. A complex method local search algorithm is used to improve the accuracy of the algorithm optimization. A non-fixed multi-segment mapping penalty function is adopted to process constraints handling mechanism, and the constrained optimization problem is transformed into a non-constrained problem. The experimental results obtained by the improved cuckoo search algorithm has shown to be very efficient for several benchmark test functions and engineering design optimization problems. The proposed ICS algorithm also privates better performance than CS algorithm and the other intelligent ones in the literature for solving the two engineering design optimization problems.

ACKNOWLEDGEMENTS

This work is financially supported by the Natural Science Foundation of Guangxi Province (Grant No. 2014GXNSFBA118283) and the Higher School Scientific Research Project of Guangxi Province (Grant No. 2013YB247).

References

1. Yang, Xin-She, and Suash Deb,Cuckoo search via Lévy flights,IEEE World Congress on Nature & Biologically Inspired Computing (NaBIC),pp:210-214(2009)
2. Yang, Xin-She and Suash Deb,Engineering optimisation by cuckoo search, International Journal of Mathematical Modelling and Numerical Optimisation 1(4):330-343(2010)
3. Yang X S and Deb S, Multiobjective cuckoo search for design optimization, Computers & Operations Research , 40(6):1616-1624(2013)
4. Gandomi A H, Yang X S and Alavi A H, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Engineering with Computers, 29(1):17-35(2013)
5. Wen Long,Ximing Liang, Yafei Huang and Yixiong Chen , An effective hybrid cuckoo search algorithm for constrained global optimization, Neural Comput & Applic 25(34):1577-1586(2014)
6. Radova R.Bulatovic, GoranBoskovic, Mile M. Savkovic and Milomir M. G, Improved Cuckoo Search(ICS) algorthm for constrained optimization problems, Latin American Journal of Solids and Structures 11(8):1349-1362(2014)
7. Parsopoulos K E and Vrahatis M N , Particle swarm optimization method for constrained optimization problems, Intelligent Technologies–Theory and Application: New Trends in Intelligent Technologies, 76: 214-220(2002)
8. Valian E, Tavakoli S, Mohanna S and A Haghi, Improved cuckoo search for reliability optimization problems, Computers & Industrial Engineering, 64(1):459-468 (2013)
9. Runarsson T P and Yao X , Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation, 4(3):284-294(2000)
10. Mezura Montes E and Coello Coello C A, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Transactions on Evolutionary Computation , 9(1):1-17 (2005)
11. Mezura-Montes E and Cetina-Domínguez O , Empirical analysis of a modified artificial bee colony for constrained numerical optimization, Applied Mathematics and Computation, 218(22):10943-10973(2012)
12. Wang L and Li L, An effective differential evolution with level comparison for constrained engineering design, Structural and Multidisciplinary Optimization 41(6):947-963(2010)
13. Zhang M, Luo W and Wang X ,Differential evolution with dynamic stochastic selection for constrained optimization, Information Sciences 178(15):3043-3074(2008)
14. He Q and Wang L , A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization, Applied Mathematics and Computation ,186(2):1407-1422(2007)
15. Sadollah A, Bahreininejad A, Eskandar H and Mohd Hamdi , Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Applied Soft Computing 13(5):2592-2612(2013)