TELKOMNIKA, Vol.12, No.4, December 2014, pp. 1023~1030
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v12i4.535

Complex Optimization Problems Using Highly Efficient Particle Swarm Optimizer

Kaiyou Lei[1], Changjiu Pu*[2]
[1] Intelligent Software and Software Engineering Laboratory, Southwest University, Chongqing 400715, Chongqing, China
[2] Network Center, Chongqing University of Education, Chongqing 400067, Chongqing, China
*Corresponding author, e-mail: pcj8880289@sina.com

Abstract

Many engineering problems are the complex optimization problems with the large numbers of global andlocal optima. Due to its complexity, general particle swarm optimization method inclines towards stagnation phenomena in the later stage of evolution, which leads to premature convergence. Therefore, a highly efficient particle swarm optimizer is proposed in this paper, which employ the dynamic transitionstrategy ofinertia factor, search space boundary andsearchvelocitythresholdbased on individual cognitionin each cycle to plan large-scale space global search and refined local search as a whole according to the fitness change of swarm in optimization process of the engineering problems, and to improve convergence precision, avoid premature problem, economize computational expenses, and obtain global optimum. Several complex benchmark functions are used to testify the new algorithm and the results showed clearly the revised algorithm can rapidly converge at high quality solutions.

Keywords: *particle swarm optimizer, complex optimization problem, premature convergence*

1. Introduction

As a newly developed population-based computational intelligence algorithm, Particle Swarm Optimization (PSO) was originated as a simulation of simplified social model of birds in a flock [1]-[4]. The PSO algorithm has less parameters, easy implementation, fast convergence speed and other characteristics, is widely used in many fields,such as solving combinatorial optimization, fuzzy control, neural network training, etc. But, the PSO algorithm with other algorithms is also easy to fall into local optimumin fast convergence process, affecting the convergence precision, so how to overcome premature convergence, and improve the accuracy of convergence is always a hot and difficult problem in the research field [5]-[11].

To avoidthe premature problem and speed up the convergence process, thereare many approaches suggested by researchers.According to the research results published in recent years, the improvement of PSO algorithm mainly includes adjusting algorithm parameters, the improvement of topological structure, and mixed with other algorithm, etc [6]-[12].The purpose of improvement strategiesis to balance the global search ability and local search ability of particles, so as to improve the performance of the algorithm.

In this paper, we modified the traditional PSO (TPSO) algorithm with the dynamic transition strategy ofinertia factor, search space boundary andsearchvelocitythresholdbased on individual cognitionin each cycle,whichcan balance the global search ability and local search ability of particles, and has an excellent search performance to lead the search direction in early convergence stage of search process. Experimental results on several complexbenchmark functions demonstrate that this is a verypromisingway to improve the solution quality and rate of success significantly in optimizing complex engineering problems.

Section 2 gives some background knowledge of the PSO algorithm. In section 3, the proposed method and the experimental design are described in detail, and correlative results are given in section 4. Finally, the discussions are drawn in section 5.

2. Back Ground

In 1995, the particle swarm optimizer (PSO) is a populationbasedalgorithm that wasinvented by James Kennedy and Russell Eberhart,which was inspired by the social behaviorof animals such as fish schooling and bird flocking.Similar to other population-based

Received August 28, 2014; Revised October 30, 2014; Accepted November 16, 2014

algorithms, suchas evolutionary algorithms, PSO can solve a variety ofdifficult optimization problems but has shown a fasterconvergence rate than other evolutionary algorithms onsome problems. Anotheradvantage of PSO is that it has very few parameters toadjust, which makes it particularly easy to implement [1]. In PSO, each potential solution is a “bird” in the search space, which is called “particle”. Each particle has a fitness value evaluated by the objective function, and flies over the solution space with a velocity by following the current global best particle and its individual best position. With the directions of best particles, all particles of the swarm can eventually land on the best solution.

The foundation of PSO is based on the hypothesisthat social sharing of information among conspecificsoffers an evolutionary advantage.In the original PSO formula, particle i is denoted as $X_i=(x_{i1},x_{i2},...,x_{iD})$, which represents a potential solution to a problem in D-dimensional space. Each particle maintains a memory of its previous best position *Pbest,* and a velocity along each dimension, represented as $V_i=(v_{i1},v_{i2},...,v_{iD})$. At each iteration, the position of the particle with the best fitness in the search space, designated as g, and the P vector of the current particle are combined to adjust the velocity along each dimension, and that velocity is then used to compute a new position for the particle.

In TPSO, the velocity and position of particle *i* at (*t*+1)th iteration are updated as follows:

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * \left(p_{id}^{t} - x_{id}^{t}\right) + c_2 * r_2 * \left(p_{gd}^{t} - x_{id}^{t}\right) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (2)$$

Constants c1 and c2 determine the relative influence of the social and cognition components (learning rates), which often both are set to the same value to give each component equal weight; r_1 and r_2 are random numbers uniformly distributed in the interval [0,1].A constant, v_{max}, was used to limit the velocities of the particles. The parameter *w*, which was introduced as an inertia factor, can dynamically adjust the velocity over time, gradually focusing the PSO into a local search [5].

To speed up the convergence process and avoid the premature problem, Shi proposed the PSO with linearly decrease factor method (LDWPSO) [4],[5]. Suppose $w_{max}$is the maximum of inertia factor, w_{min} is the minimum of inertia factor, *run* is the current iterations, run_{max} is the total iterations. The inertia factor is formulated as:

$$w = w_{max} - \left(w_{max} - w_{min}\right) * \frac{run}{run_{max}} \quad (3)$$

3. A Highly Efficient Particle Swarm Optimizer (HEPSO)

Due to thecomplexity of a great deal global and local optima,TPSO isrevised as HEPSO by fourdynamic strategies to adapt complex optimizationproblems.

3.1. Dynamic Harmonization Inertia Factor *w*

First of all, the larger *w* can enhance global search abilities of PSO, so to explore large-scale search space and rapidly locate the approximate position of global optimum, the smaller *w* can enhance local search abilities of PSO, particles slow down, deploy refined local search, and obtain global optimum. Secondly, the more difficult the optimization problems are, the more fortified the global search abilities need, once located the approximate position of global optimum, the refined local search will further be strengthen to get global optimum [7]-[12]. Therefore,the *w*can harmonize global search and local searchautomatically, avoid premature convergenceand to rapidly gain global optimum.

According to the conclusions above, a new inertia factor decline curve (4)for PSO is constructed, demonstrated in Figure 1:

$$w = w_{max} * \exp\left(-n * \left(\frac{run}{run_{max}}\right) \wedge n\right) \quad (4)$$

Where n is a constant larger than 1, taken 50 as initial value in the paper.Evidently, the inertia factor decline curve of figure 1can forcefully search large-scale global space, and dynamically transform to refined local search, namely global search abilities and local search abilities are harmonized based on the strategy to adapt demand of complex optimizationproblems.

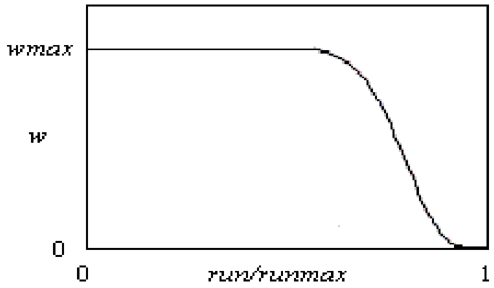

Figure1. Dynamic harmonization w curve

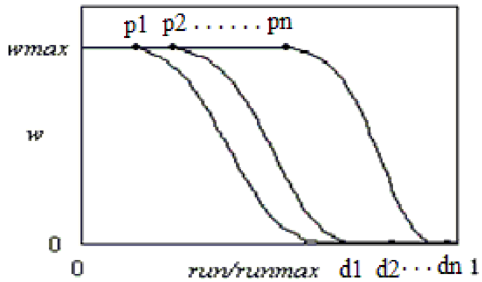

Figure 2. Dynamic transformation w curves

3.2.Dynamic Transformation Inertia Factor *w* Strategy

Global search and local search are two key aspects of PSO based on w. In a given time of search process, it is usually hard to determine, when to end the large-scale global search, and start local search,and gain quick convergence [8]-[10].

In figure2,p1, p2,…, pn are transformation points, d1, d2, …, dn are global convergence points, the algorithm select a transformation point from them, and switch to refined local search to global convergence point. The selection of transformation pointis usually hard. Toconfirm the transformation point,the algorithm is designed to combine iteration times of current global optimum of functions.If the current global optimum is not improved after the search of an interval of definite iterations, the algorithm switch to refined local search with the smaller n, or continue current global search with the current n. Thecomputed equation is defined as:

$$IF\ \ p_{gd}^{i+k} \ge p_{gd}^{i} \quad n = n \quad esle \quad n = r_1 * r_2 * n \tag{5}$$

Where p_{gd}^{i+k} , p_{gd}^{i} are the $(i+k)$th, ith ,taken values of p_{gd}^{t} respectively, k is an interval of definite iterations.

3.3.Dynamic Transformation Search Space Boundary Strategy

In search process, all particles gather gradually to the current best region, the algorithmis propitious to quicken convergencebecause of the reducedsearch space, but, the global optima may be lost [7]-[10]. In most cases, the global optima may be hidden somewhere in the gathering area nearby, and the effective search areafound is not easy.To solve theproblem, the improved algorithm not only reduces the search space to quicken convergence, but also avoids the premature problem, especially in complex optimizationproblems.Thus, a dynamic transformation search spaceboundary strategy is designed based on individual cognition. Assume that a particle flight in the current boundary $[b_{max}(i),b_{min}(i)]$, the algorithmreduce the search boundary if the current optimum is better, otherwise, expand search boundary in next iteration, and in the same breath, randomly initialize the speed and position of each particle after the k iterations. Theb_{max} and b_{min} are the boundary of the swarm in the k iteration. The computed equation is defined as:

$$
\begin{aligned}
&IF\ gbest_{i+1} > gbest_i\\
&b_{max}(i+1) = b_{max}(i) + r_1 * r_2 * (|b_{max} - b_{min}|)\\
&b_{min}(i+1) = b_{min}(i) - r_1 * r_2 * (|b_{max} - b_{min}|)\\
&else\quad b_{max}(i+1) = b_{max}(i) - r_1 * r_2 * (|b_{max} - b_{min}|)\\
&b_{min}(i+1) = b_{min}(i) + r_1 * r_2 * (|b_{max} - b_{min}|)
\end{aligned} \tag{6}
$$

3.4. Dynamic Transformation Search Velocity ThresholdStrategy

Many published works based on parameters selection principles pointed out, velocity threshold [$v_{max}(i)$,$v_{min}(i)$] of a particleaffects the convergence precision and speed of algorithm strongly [9]-[11]. Large$v_{max}(i)$increases the search region, enhancing global search capability, as well as small $v_{max}(i)$ decreases the search region, adjusting search direction of each particle frequency. Thus, adynamic transformation search velocitythresholdstrategy is designed based on individual cognition. Thev_{max} and v_{min} are the threshold of the swarm in the k iterations, the computed equation is defined as:

$$
\begin{aligned}
&IF\ gbest_{i+1} > gbest_i\\
&v_{max}(i+1) = v_{max}(i) + r_1 * r_2 * (|v_{max} - v_{min}|)\\
&v_{min}(i+1) = v_{min}(i) - r_1 * r_2 * (|v_{max} - v_{min}|)\\
&else\quad v_{max}(i+1) = v_{max}(i) - r_1 * r_2 * (|v_{max} - v_{min}|)\\
&v_{min}(i+1) = v_{min}(i) + r_1 * r_2 * (|v_{max} - v_{min}|)
\end{aligned} \tag{7}
$$

According to the above methods, TPSO is modified as HEPSO, whichhas the excellent search performance to optimize complex problems. The flow of the HEPSOalgorithm is as follows:

Step1. Set algorithm parameters;
Step2. Randomly initialize the speed and position of each particle;
Step3. Evaluate the fitness of each particle and determine the initial values of the individual and global best positions: p_{id}^t and p_{gd}^t;
Step4. Update velocity and position using (1), (2) and (4);
Step5. Evaluate the fitness and determine the current values of the individual and global best positions: p_{id}^t and p_{gd}^t;
Step6. Detect the $gbest_i$, $gbest_{i+1}$ and $gbest_{i+k}$, to dynamically transform$w$,search space boundary and velocity threshold using (5), (6)and (7);
Step7. Randomly initialize the speed and position after the k iterations;
Step8. Loop to Step 4 and repeat until a given maximum iteration number is attained or the convergence criterion is satisfied.

4. Computational Experiments

4.1.Testing Functions

To test the HEPSO and compare it with other techniques in the literature, we adopt large variety of benchmark functions [8]-[16], among which most functions are multimodal, abnormal or computational time consuming, and can hardly get favorable results by current optimization algorithm. Due to limited space, we only select four representative functions optimization results to list in the paper.

$$f_1(x) = \sum_{i=1}^{n} \left(-x_i \sin\sqrt{|x_i|}\right) \quad -500 \le x_i \le +500 \tag{8}$$

$$f_2(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10] \qquad -5.12\le x_i\le +5.12 \tag{9}$$

$$f_3(x)=-20\exp(-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i 2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))+20+e \qquad -32\le x_i\le +32 \tag{10}$$

$$f_4(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1 \qquad -600\le x_i\le +600 \tag{11}$$

4.2. Algorithm Parameter Setting

Parameters used in our algorithm are set to be: learning rate c_1=c_2=2; w=0.7, w_{max} =1, w_{min}=0.1; maximumiterations run_{max}=30000; iterations k =1000(300 for $f_1(x)$); population size is 100; speed and position of particles are limited in definition area of functions; take function $f_1(x)$, $f_2(x)$, $f_3(x)$, $f_4(x)$ as fitness value. Stop rule is: |$f$best–$f$min|≤10^{-4} ($f$best and $f$min are the global optimum and the current getting optimum). The running environment is: MATLAB7.0, Pentium IV 2GHz CPU, 256M RAM, Win XP OS.

4.3.Experimental Results

The testing functions is run50 times based onTPSO, LDWPSO and HEPSO, the comparison of statistical results of 20-1000 dimensions functions are shown in table 1-2, respectively. In addition, the datum of literature [12] (MAGA) is likewise listed in table 1. 1000-10000 dimensions functions are test with MAGA,TPSO, LDWPSO and HEPSO based on the sampling interval 500, each testing function runs 20 times yet, the statistical results are shown in table 1-2 and figure 3-6, respectively.

Table 1. Results of 20-10000 dimensions functions average convergence iterations

n	Function	Stopcriterion	Algorithm			
			MAGA	TPSO	LDWPSO	HEPSO
20	$f_1(x)$	10^{-4}	2483	2120	1872	802
	$f_2(x)$		4301	685	377	102
	$f_3(x)$		3583	731	608	77
	$f_4(x)$		2566	936	1873	1632
100	$f_1(x)$	10^{-4}	5443	3743	2659	867
	$f_2(x)$		10265	934	865	209
	$f_3(x)$		5410	872	864	151
	$f_4(x)$		4447	1169	948	2346
200	$f_1(x)$	10^{-4}	7284	6562	4458	887
	$f_2(x)$		14867	1183	962	1278
	$f_3(x)$		6061	1063	947	165
	$f_4(x)$		5483	1349	1135	2678
400	$f_1(x)$	10^{-4}	12368	10348	7659	1586
	$f_2(x)$		17939	1461	1123	1743
	$f_3(x)$		6615	1564	1062	245
	$f_4(x)$		6249	1723	1587	2356
103	$f_1(x)$	10^{-4}	22827	15617	13457	2654
	$f_2(x)$		20083	2834	1260	1668
	$f_3(x)$		7288	2034	1143	389
	$f_4(x)$		7358	2327	4562	2698
2×10^3	$f_1(x)$	10^{-4}	45621	27435	18652	10845
	$f_2(x)$		17521	6533	4534	3532
	$f_3(x)$		7156	2867	1145	452
	$f_4(x)$		14578	4021	2523	2034
4×10^3	$f_1(x)$	10^{-4}	90453	70123	40032	15034
	$f_2(x)$		13067	8545	4065	2056
	$f_3(x)$		7611	3823	1945	624
	$f_4(x)$		12034	6701	4967	2506
6×10^3	$f_1(x)$	10^{-4}	130067	85324	43136	25156
	$f_2(x)$		13166	9022	4517	2533
	$f_3(x)$		7607	4712	2055	1156
	$f_4(x)$		16223	8156	3578	3156

n	Function	Stopcriterion	Algorithm			
			MAGA	TPSO	LDWPSO	HEPSO
8×10^3	$f_1(x)$	10^{-4}	125245	81489	61378	18000
	$f_2(x)$		14523	85434	6556	3500
	$f_3(x)$		7921	4567	3467	568
	$f_4(x)$		18234	7067	4523	3512
10^4	$f_1(x)$	10^{-4}	198745	130679	85045	41289
	$f_2(x)$		19807	14332	8523	4668
	$f_3(x)$		7992	6621	4434	1745
	$f_4(x)$		27983	15357	13534	6123

Table 2.Comparison results of 20-10000 dimensions functions average convergence rate (%)

n	f_1 (x)			f_2 (x)		
	TPSO	LDWPSO	HEPSO	TPSO	LDWPSO	HEPSO
20	82.2	89.7	100	100	100	100
100	52.7	65.8	100	84.4	100	100
200	33.5	53.5	100	66.3	90.7	100
400	26.6	45.1	98.1	43.8	65.2	100
10^3	6.8	22.1	93.2	31.2	54.3	100
2×10^3	5.5	19.8	84.3	24.2	43.6	89.8
4×10^3	4.2	16.8	69.1	22.3	39.7	84.4
6×10^3	2.9	14.4	60.1	19.7	32.6	78.3
8×10^3	1.8	12.8	53.6	28.5	30.5	70.1
10^4	1.3	11.6	44.7	16.3	25.9	64.8
n	f_3 (x)			f_4 (x)		
	TPSO	LDWPSO	HEPSO	TPSO	LDWPSO	HEPSO
20	87.3	100	100	69.4	90.6	100
100	61.8	91.2	100	50.7	72.5	100
200	40.6	72.6	100	30.1	53.8	95.7
400	33.9	54.5	95.3	20.8	38.8	90.3
10^3	14.1	47.3	90.6	7.2	24.3	89.5
2×10^3	8.7	22.8	84.6	23.5	23.5	78.2
4×10^3	7.1	20.4	68.4	21.7	17.3	58.4
6×10^3	5.4	15.6	60.1	18.5	14.2	55.1
8×10^3	4.9	14.1	57.8	16.9	12.3	50.4
10^4	3.6	12.4	53.6	15.3	10.5	43.6

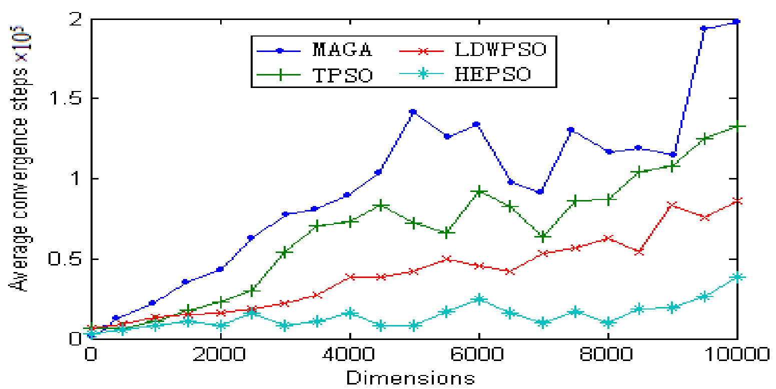

Figure 3. The convergence results of $f_1(x)$

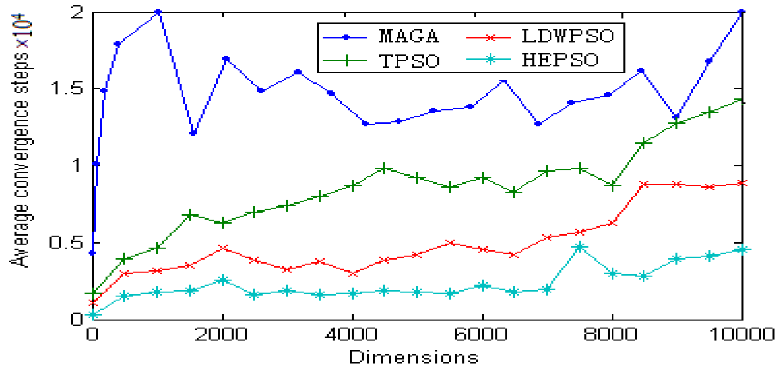

Figure 4. Theconvergence results of $f_2(x)$

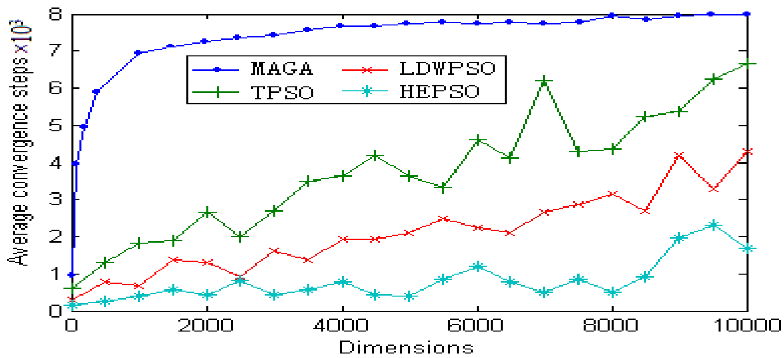

Figure 5.Theconvergence results of $f_3(x)$

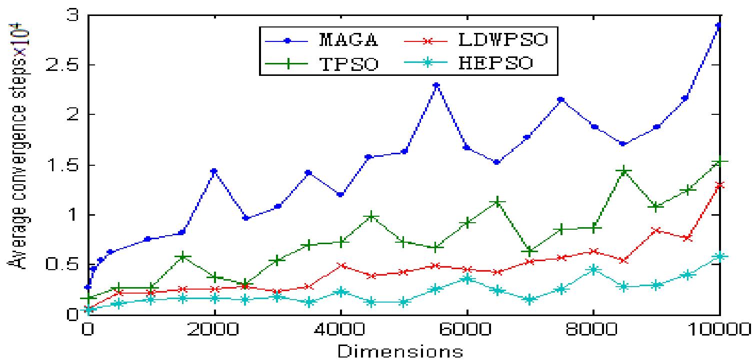

Figure 6. Theconvergence results of $f_4(x)$

5. Conclusion

The experimental results of Table 1-2 can deduce that the effectiveness of the HEPSOalgorithm based on individual cognitionis validated,which guide particles to search in the more effective areathrough dynamic adjustmentthe search space, provide stable convergence, resulting in higher success rate and accuracy of convergence. The algorithm runs classical PSO only, so to keeps its simple and easy characteristic.

The experimental results of Figure 3-6 show that the HEPSOalgorithm has excellent search performance, especially complex engineering problems. As the dimensions of the functions grow fleetly, the increase of the average convergence steps is slow, so the algorithm has rapid convergence speed and can avoid premature.In addition, it can easily be applied to large and more complex practical multi-objective optimization problems.

Acknowledgements

This work is supported by Key Project of Chinese Ministry of Education (104262).

References

[1] J. Kennedy, RC. Eberhart. *Particle swarm optimization*. Proceedings of IEEE International Conference on Neural Networks. Piscataway, USA. 1995: 1942-1948.
[2] X. Hu, RC. Eberhart, YH. Shi. *Engineering optimization with particle swarm.* Proceedings of the IEEE Swarm Intelligence Symposium. Indianapolis, Indiana. 2003: 53-57.
[3] Clerc, J. Kennedy. The particle swarm: explosion, stability, and convergence in a multi-dimensional complex space. *IEEE Transactions on Evolutionary Computation*. 2002; 6(1): 58-73.
[4] YH. Shi, RC. Eberhart. *Empirical study of particle swarm optimization*. Proceedings of the IEEE Congress on Evolutionary Computation. Piscataway, USA. 1999: 1945-1950.
[5] YH. Shi, RC. Eberhart. *A modified particle swarm optimizer*. Proceedings of the IEEE Congress on Evolutionary Computation. New York. 1998: 69-73.
[6] Chen D, Wang GF, Chen ZY. *The inertia weight self-adapting in PSO*. Proceedings of the Seventh world Congress on intelligent Control and automation. New York. 2008: 5313-5316.
[7] Angeline, P. *Using selection to improve particle swarm optimization*. Proceedings of IJCNN'99. Washington, USA. 1999: 84-89.
[8] Kaiyou Lei, Yuhui Qiu, Yi He. *A new adaptive well-chosen inertia weight strategy to automatically harmonize global and local search ability in particle swarm optimization*. Proceedings of 1st International Symposium on Systems and Control in Aerospace and Astronautics.HarbinChina. 2006: 342-346.
[9] Chen Bing-rui, Feng Xia-ting. Particle swarm optimization with contracted ranges of both search space and velocity. *Journal of Northeastern University (Natural Science)*. 2005; 5: 488-491.
[10] Y.Shi, RC. Eberhart. *Parameter selection in particle swarmoptimization*. Proceedings of theSeventh Annual Conference on Evolutionary Programming. New York. 1998; 1447: 591–600.
[11] Wang Jun-wei, Wang Ding-wei. Experiments and analysis on inertia weight in particle swarm optimization. *Journal of Systems Engineering*. 2005; 2: 184-198.
[12] Zhong Wei-cai, Xue Ming-zhi, et al. Multi-agent genetic algorithm for super-high dimension complex functions optimization. *Natural Science Progress*. 2003; 10: 1078-1083.
[13] Li Bing-yu, Xiao Yun-shi, Wang Lei. A hybrid particle swarm optimization algorithm for solving complex functions with high dimensions. *Information and Control*. 2004; 1: 27-30.
[14] Hui wang, Zhijian Wu, S Rahnmayan. Enhanced opposition based differential evolution for solving high-dimensional continuous optimization problems. *Soft Compting-A Fusion of Foundations, Methodologies and Applications*. 2011; 15(11): 2127-2140.
[15] Li Wen-feng, Liang Xiao-lei, Zhang Yu. Research on PSO with clusters and heterogeneity. *Acta Electronica Sinica*. 2012; 40(11): 2194-2199.
[16] Y Shi, L Gao, G Zhang. Cellular particles warm optimization. *Information Sciences*. 2011; 181(20): 4460-4493.