Improved particle swarm optimization algorithm with unidimensional search

Pu Han, Li Meng*, Biao Wang, Dongfeng Wang

School of Control and Computer Engineering, North China Electric Power University, Baoding 071003, Hebei, China

Received 1 March 2014, www.cmnt.lv

Abstract

In this paper, a strategy of unidimensional search is introduced to particle swarm optimization (PSO). The global exploration capability of PSO is used to identify a promising region in search space. With the region as the starting point, a unidimensional local search is applied to search a more accuracy solution. The local search does not rely on the population information, which makes it can jump out of a local optimum when the population stagnates. With combination of global exploration and local exploitation, the algorithm can discover more favourable search area effectively and obtain a better solution. The improved PSO method is tested on eight benchmark functions. Experimental results show that the method can not only improve the accuracy of solution, but also reduce the influence of initial population distribution upon the algorithm performance. Finally, the influence of parameter variation on algorithm is analysed.

Keywords: particle swarm optimization, unidimensional local search, population initialization, population distribution

1 Introduction

Particle swarm optimization (PSO), which was proposed by Kennedy and Eberhart [1] in 1995, is one of the most effective intelligent optimization algorithms developed recent years. Although has some common with evolutionary algorithm, PSO does not use evolution operators such as crossover and mutation. It emulates the flocking behaviour of birds and fish. Each particle adjusts its position according to the success of itself and its neighbourhood. As PSO is simple and easy to implement, it has already been applied in many real-world problems.

However, PSO suffers from trapping in local minima point easily and slow convergence speed. Many researchers have worked on solving these problems in various ways. Shi and Eberhart [2] propose a linearly decreasing inertia weight in evolution course. In [3], a new variant of PSO with nonlinear inertia weight which relative to generation number is proposed. Parameters are adjusted automatically according to the population distribution information in [4]. In [5] orthogonal experiment design method is used to construct a better guidance exemplar by using information lies in a particle's best historical position and its neighbourhood's best position. In [6, 7] particles have several updating strategies. A choice mechanism related to the population information is used to steer different strategies in an adaptive and parallel way. To remain the population diversity, [8] proposes comprehensive learning particle swarm optimizer, which enables each dimension of a particle, has the opportunity to learn from a different exemplar. In [9] information drawn from the distances between the global best particle and every other particle is applied to updating particle velocity. In [10] neighbourhood structure is changed dynamically during evolution. The topology structure in [11] is tree-like in which particles with better evaluations are placed in the upper nodes of the tree. Particles' positions in the tree are adjusted at each iteration. [12] uses global exploration capability of PSO algorithm to locate a good local minimum approximately, then quasi Newton-Raphson is done with the best solution as its starting point for accurate local exploration. [13] proposes a hybrid swarm optimizer which combines PSO with LM (Levenberg-Marquardt) algorithm. In [14] six discrete crossover operators are incorporated respectively into a global best particle swarm optimizer.

Although the existing variants of PSO have some exciting results, there still some effort to do. When the search space is high-dimensional, there exists such a phenomenon that some dimensions develop toward global optima, while some deviate from it. If the search can detail to each dimension, there will be a considerable improvement on algorithm performance. A new learning strategy is introduced in [15]. The whole position vector is divided into several subvectors. The subvectors are updated iteratively in the updating process of a position vector. In this paper this strategy is integrated into PSO for a more accurate single dimensional search based on the potential solution located by PSO. The search based on a dimension by dimension way can protect the dimensions which are in good positions and mitigate against premature convergence in a single dimension. Then the more favourable result obtained by local search is fed back to population evolution as a guide.

* ***Corresponding author*** e-mail: mengli2014@163.com

2 Standard PSO (SPSO)

In SPSO each particle represents a potential solution in D dimensional search space. $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$ is the position of particle i, $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$ is the velocity. $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$ is the best previous position of particle i, $p_g = (p_{g1}, p_{g2}, ..., p_{gD})$ is the best solution found by the whole population so far. The dth dimension of position and velocity of particle i are updated as Equations (1) and (2):

$$v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1 (p_{id} - x_{id}^{k}) + c_2 r_2 (p_{gd} - x_{id}^{k}), \quad (1)$$

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

where k is current iteration number, r_1 and r_2 are random number between $[0,1]$, keeping the diversity of population; c_1 and c_2 are acceleration constants; w is inertia weight which used to balance between the local and global search abilities.

3 Improved particle swarm optimization algorithm (IPSO)

In SPSO a particle updates each dimension of its position vector at one time, there is no guarantee that all dimensions fly to global optima. In our method, the search strategy borrowed from literature [15] is applied to single-dimensional local search around the promising region found by PSO. Then the more precise solution is employed to guiding population evolution.

3.1 THE IMPROVED ALGORITHM

Here, a new variable named p_g' is introduced. Meanwhile p_g which stands for the best solution found by population (as defined in section 2) is reserved. To distinguish, we call p_g' is the best solution found by the new algorithm, not by population. The initial value of p_g' is set as equal as p_g. When the population evolves a certain (M) iterations, a single dimension local search for p_g' is proceeded, updating each dimension of p_g' circularly. If a better solution is yielded, we update p_g' as the better solution. In the next M iterations, when p_g is superior to p_g', p_g' renewal as p_g. Otherwise p_g' remains unchanged until next local search. In our algorithm, global search and local search proceed repeatedly. In order to avoid converging to p_g' too quickly, p_g' and p_g are all acting on the updating of particle velocity, see Equation (3):

$$v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1 (p_{id} - x_{id}^{k}) + c_2 r_2 (r_3 (p_{gd} - x_{id}^{k}) + (1 - r_3)(p'_{gd} - x_{id}^{k})), \quad (3)$$

x_i, v_i, p_i, p_g, w, c_1, c_2, r_1 and r_2 in Equation (3) are the same as in SPSO. r_3 is a random number with uniformly distributed between $[0,1]$. The update of position vector is as Equation (2). Figure 1 is a flow chart of the IPSO, where minimum optimization problem is taken as a example. The process of local search can be seen from the chart. Also the distinction between p_g and p_g' is more easy to understand.

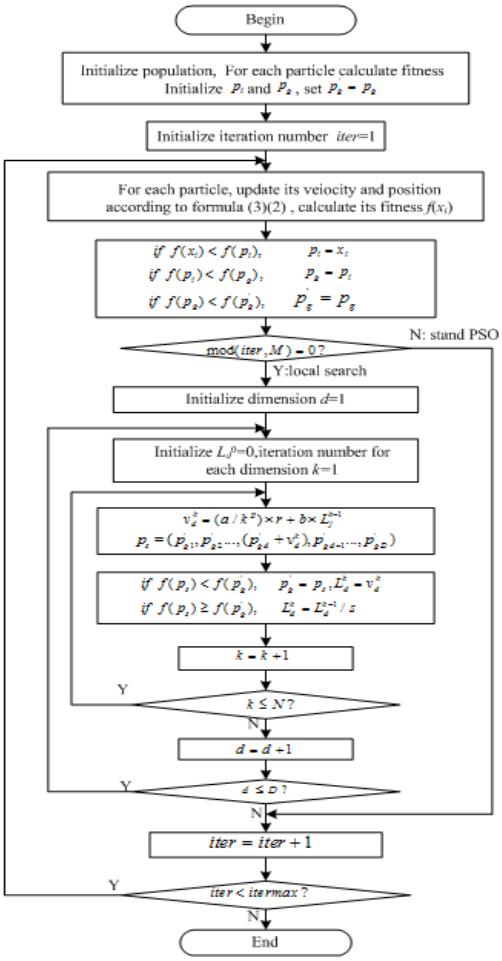

FIGURE 1 Flowchart of IPSO

In Figure 1, $mod()$ is remainder function. $itermax$ is maximum iterations of population. N is maximum iterations for single dimensional search. a, p, b, s and r are parameters for single dimensional search. Velocity vector consists of two parts [15]: diversity section $(a / k^p) \times r$ and the learning section $b \times L$. a is diversity factor, p is decreased factor, a and p are all positive constants. r is a random number with uniformly distributed between $[-0.5, 0.5]$. b and s are all constants greater than 1.

3.2 COMPARED WITH SPSO

In our work eight benchmark functions are used to test the algorithm. The name, search space and search target accuracy of every function are shown in Table 1.

Generally it is difficult to determine the appropriate range of search space for real-life optimization problems. If the performance of the optimum algorithm has well relationship to the distribution of initial population, the results may not be satisfactory. Given this situation, the experiment is designed as follows: Assuming the search space range is $[-Space_{Max}, Space_{Max}]$, there are two methods to initialize the population: $P1$: the particles are uniformly distributed in $[0, Space_{Max}]$; $P2$: the particles are uniformly distributed in $[-Space_{Max}, Space_{Max}]$.

TABLE 1 Test function used in this paper

Fun No.	Fun Name	Expession	Search Space	Optimal Solution	Target Precision
f_1	sphere	$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	$f(0,0\ldots0)=0$	10^{-20}
f_2	rosenbrock	$f_2(x)=\sum_{i=1}^{D-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$	$[-30,30]^D$	$f(1,1\ldots1)=0$	1
f_3	rastrigin	$f_3(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	$[-5.12,5.12]^D$	$f(0,0\ldots0)=0$	20
f_4	quadric	$f_4(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$	$[-100,100]^D$	$f(0,0\ldots0)=0$	1
f_5	schwefel's P 2.22	$f_5(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{d}\lvert x_i\rvert$	$[-10,10]^D$	$f(0,0\ldots0)=0$	10^{-2}
f_6	schwefel's P 2.21	$f_6(x)=\max_i\{\lvert x_i\rvert, 1\le i\le D\}$	$[-100,100]^D$	$f(0,0\ldots0)=0$	10^{-2}
f_7	penalized	$f_7(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2\cdot[1+\sin^2(3\pi x_{i+1})]+(x_D-1)^2\cdot[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D}U(x_i,5,100,4)$ $U(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<-a\end{cases}$	$[-50,50]^D$	$f(-1,-1\ldots-1)=0$	10^{-5}
f_8	ackley	$f_8(x)=-20e^{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}}-e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)}+20+e$	$[-32,32]^D$	$f(0,0\ldots0)=0$	10^{-2}

All functions are set as 30 dimensions. Each algorithm runs 50 times independently. The algorithm parameters are set as: SPSO: population size is 20, the maximum iterations is 5000, $c_1=c_2=1.49618$, $w=0.72984$; IPSO: parameters for population are the same as SPSO, parameters for local search: diversity factor $a=3$, decreased factor $p=6$, $b=2$, $s=4$. For each algorithm, record mean value (Mean) and standard deviation (SD) of the solutions obtained in the 50 independent runs. Meanwhile record the success ratio (R) for target accuracy. The results are given in Table 2. IPSO achieves better results than SPSO on all test functions. IPSO shows a success rate of 100% except f_2 and f_6. SPSO only has a so perfect work on f_1 ($P2$), but a rather unsatisfactory work on f_3, f_4 ($P1$), f_5 (P1) and f_8. For the two initialization modes, IPSO has a fairly stable performance on all functions. While SPSO performs very differently on f_1, f_2, f_4 and f_7. On the whole, SPSO gives a poor performance when the initial population distributes as $P1$. The optimal solution of test functions used here are mostly distributed in the middle of the space search, particles are far from the optimal point when they initialized as $P1$. This may increase the difficulty to local the optimal region.

Figure 2 gives a more intuitive comparison between the two algorithms. As can be seen from Figure 2, IPSO converges faster than SPSO no matter how the distribution of the initial population. When initial population obtained by means of $P1$, SPSO falls into local optimum and stagnates too early on most of the tested functions except f_6. Figure 3 shows the results obtained in 50 runs, illustrating the solution distribution.

TABLE 2 Test functions and results

		sphere		**rosenbrock**		**rastrigin**		**quadric**	
		P1	*P2*	*P1*	*P2*	*P1*	*P2*	*P1*	*P2*
SPSO	**Mean**	2.72e+004	3.85e-043	8.85e+007	7.52e+003	2.93e+002	1.01e+002	1.24e+005	4.7e+003
	SD	1.23e+004	2.63e-042	6.68e+007	2.40e+004	5.41e+001	2.53e+001	9.81e+004	6.2e+003
	R(%)	4.0	100.0	2.0	18.0	0.0	0.0	0.0	44.0
IPSO	**Mean**	2.36e-063	1.43e-066	2.53e+000	2.19e+000	0.00e+000	0.00e+000	9.51e-013	3.66e-013
	SD	1.08e-062	5.80e-066	2.87e+000	3.05e+000	0.00e+000	0.00e+000	1.24e-012	4.41e-013
	R(%)	100.0	100.0	50.0	60.0	100.0	100.0	100.0	100.0
		schwefel's P 2.22		**schwefel's P 2.21**		**penalized**		**ackley**	
		P1	*P2*	*P1*	*P2*	*P1*	*P2*	*P1*	*P2*
SPSO	**Mean**	5.62e+001	4.17e+000	6.54e-002	2.66e-001	3.20e+008	2.26e-001	1.14e+001	4.16e+000
	SD	1.64e+001	7.56e+000	1.17e-001	4.95e-002	3.45e+008	5.72e-001	5.04e-002	2.88e+000
	R(%)	0.0	70.0	22.0	52.0	34.0	68.0	0.0	0.0
IPSO	**Mean**	1.08e-013	1.92e-013	2.43e-003	5.05e-004	2.07e-004	2.03e-004	3.57e-013	4.43e-013
	SD	5.22e-013	9.75e-013	2.91e-003	1.63e-003	1.95e-019	1.86e-019	1.51e-012	1.72e-012
	R(%)	100.0	100.0	98.0	98.0	100.0	100.0	100.0	100.0

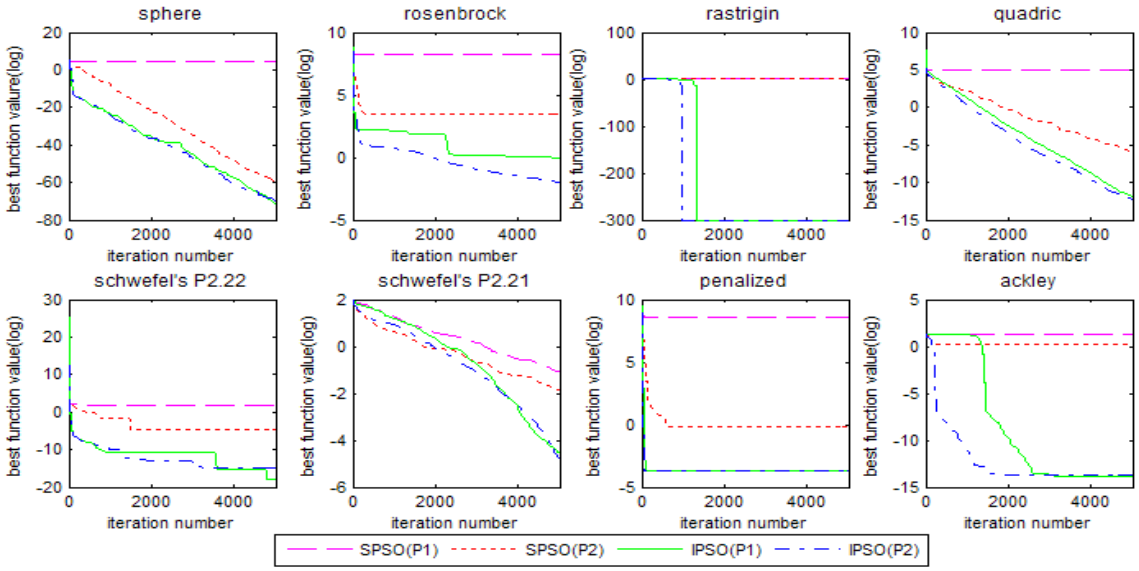

FIGURE 2 Best solution found by algorithm during evolution

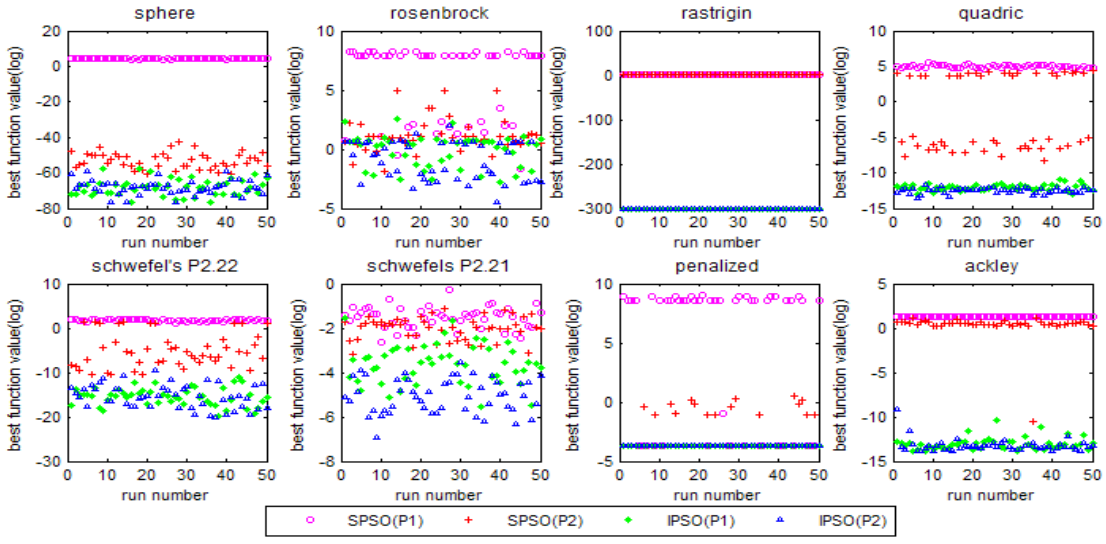

FIGURE 3 Solution distribution of 50 runs

3.3 THE INFLUENCE OF PARAMETERS

Form section 3.2, we can see that with same control parameters, IPSO have good performances than SPSO on all tested functions. In this section, we study the performances of IPSO when inertia weight w and acceleration constants c_1, c_2 changes. Here, w varies from 0.6 to 0.8, with the variation amplitude of 0.02. Set $c_1 = c_2 = c$, c varies from 1.0 to 2.0, with the variation amplitude of 0.1. The other parameters are set as the same as in section 3.2. The initial population distributes as $P2$, namely distributes uniformly within the whole search space. For each set of w and c, record the average value of results of 50 independent runs. Then a 3D graphics can be obtained, please see Figure 4.

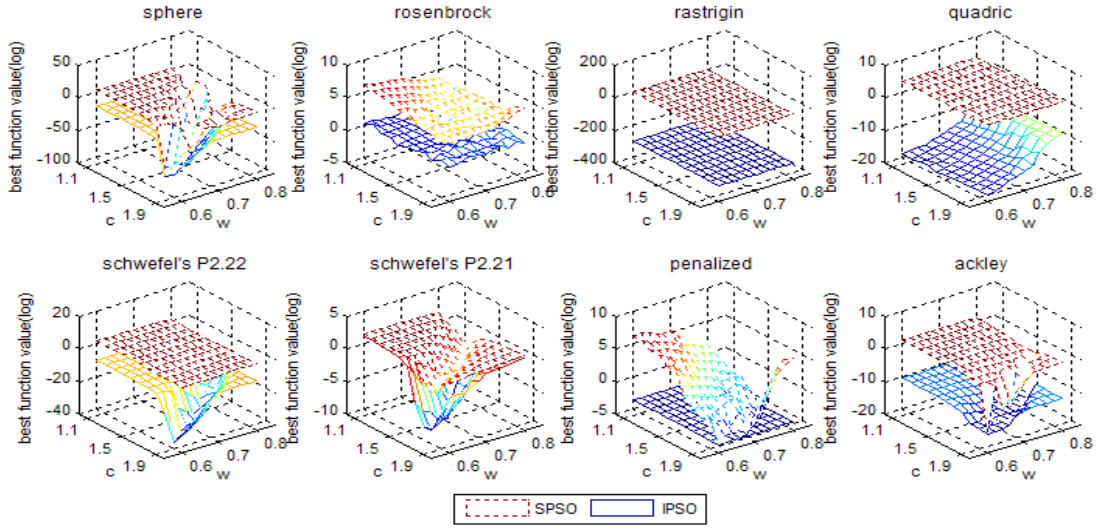

FIGURE 4 Influence of parameter changing

It can be seen from Figure 3, IPSO is always superior to SPSO when parameters vary. The affects of parameters variation on different functions are different. For f_1 and f_6, when c takes value of 1.0 to 1.2, the varying of w is basically no influences on the performances of the two algorithms; when c takes value of 1.3 to 2.0, the influences of w on the two algorithms are the same by the large: when c increases, w should decrease in order to get a better performance. For f_2 and f_8, the effect trends of parameters changing on the two algorithms are the same in the gross, except that SPSO has some mutation phenomenon, and IPSO is relatively stable. For f_7 SPSO is influenced by parameters greatly while IPSO is more stable. For f_5 SPSO is more stable when parameter varies, the change trend of performances of IPSO is the same as that for f_1. Moreover, when c is 1.8 and w is 0.6, IPSO achieve the best solutions on f_1 and f_5. For f_4, SPSO is stable; when c is fixed, the performance of IPSO steady decline with the increase of w, when w is fixed, the effect of c on the algorithm is relatively small. Generally, we can choose population parameters according to the principle which in SPSO.

4 Conclusions

When solving high-dimensional complex functions, SPSO is easy to prematurely fall into local optimum and optimizing accuracy is not satisfactory. In addition to the loss of diversity, the updating mode of a particle is also responsible for the defects, in which all dimensions of the position are updating at one time. In this paper, on the basis of solution found by population global search, a single-dimension local search is applied for a more precise solution. Then this better solution is used to guide the population search. By iteratively repeated, the global and local search fully combined. Based on the experimental results, it can be concluded that IPSO is superior to the SPSO on convergence speed and solution precision on the chosen benchmark functions.

Another advantage of IPSO is that its performance is basically not affected by the mode of initial population distribution. This is splendid for practical problems. However, the performance of IPSO is influenced by the parameters, which is the same as SPSO. How to select appropriate parameters for different problems is also a problem worthy of studying.

Besides SPSO, the unidimensional search strategy can be combined with other PSO variants. Also the method for unidimensional updating can be replaced. And another more exciting algorithm may be obtained.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61203041).
The authors thank the editor and reviewers for their valuable comments to improve this paper.

References

[1] Kennedy J, Eberhart R C 1995 Particle swarm optimization *Proc Conf on Neural Networks Piscataway USA* 1942-8
[2] Shi Y H, Eberhart R C 1998 A modified particle swarm optimizer *Proc. Conf. on Evolutionary Computation Anchorage AK* 69-73
[3] ChaReoee A, Siarry P 2006 Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization *Computers and Operations Research* **33**(3) 859- 71
[4] Zhan Z H, Zhang J, Li Y, Chung H S-H 2009 *IEEE Transactions On Systems Man and Cybernetics-part B: Cybernetics* **39**(6) 1362-81
[5] Zhan Z H, Zhang J, LI Y, Shi Y H 2011 *IEEE transactions on Evolutionary Computation* **15**(6) 832-47
[6] Wang Y, Li B, Thomas W, Wang J Y, Yuan B, Tian Q J 2011 Self-adaptive learning based particle swarm optimization *Information Science* **181**(20) 4514-38
[7] Li C H, Yang S X, Neuyen T T 2012 *IEEE Transactions On Systems Man and Cybernetics-part B: Cybernetics* **42**(3) 627-46
[8] Liang J J, Qin A K, Suganthan P N, Baskar S 2006 *IEEE Transactions on Evolutionary Computation* **10**(3) 281-95
[9] Martins A A, Oluyinka A A 2013 An adaptive velocity particle swarm optimization for high-dimensional function optimization *Proc. Conf. on Evolutionary Computation Cancun Mexico* 2352-9
[10] Suganthan P N 1999 Particle swarm optimizer with neighborhood operator *Proc. Conf. on Evolutionary Computation Piscataway NJ* 1958-62
[11] Janson S, Middendorf M 2005 *IEEE Transactions on Systems Man and Cybernetics Part B: Cybernetics* **35**(6) 1272-82
[12] Noel M M 2012 A new gradient based particle swarm optimization algorithm for accurate computation of global minimum *Applied Soft Computing* **12**(1) 353-9
[13] Katare S, Kalos A, West D 2004 A hybrid swarm optimizer for efficient parameter estimation *Proc Conf on Evolutionary Computation Portland US* 309-15
[14] Engelbrecht A P 2013 Particle swarm optimization with discrete crossover *Proc Conf on Evolutionary Computation Cancun Mexico* 2457-64
[15] Ji Z, Liao H L, Wang Y W, Wu Q H 2007 A novel intelligent particle optimizer for global optimization of multimodal functions *Proc Conf on Evolutionary Computation Singapore* 3272-5

Authors

Pu Han, born in September, 1959, China

Current position, grades: a professor and the Ph.D. supervisor with the School of Control and Computer Engineering, China.
University studies: B.Eng. degree in thermal measurement and automation from North China Electric Power University in 1982.
Scientific interest: modern control theory and its application, modeling and control of electric power process,.
Publications: 12 books.

Li Meng, born in July, 1985, China

Current position, grades: Ph.D. student in North China Electric Power University, China.
University studies: B.Eng. degree in automation from North China Electric Power University in 2008.
Scientific interest: swarm intelligence-based optimization and its application.

Biao Wang, born in October, 1987, China

Current position, grades: a laboratory teacher in North China Electric Power University, China.
University studies: M.Eng. degree in control engineering from North China Electric Power University in 2013.
Scientific interest: modern control theory and its application.

Dongfeng Wang, born in January, 1971, China

Current position, grades: a professor with the School of Control and Computer Engineering, China.
University studies: Ph.D. degree in thermal engineering from North China Electric Power University in 2001.
Scientific interest: swarm intelligence- based optimization and intelligent control.