Research on Fuzzy Adaptive Optimization Strategy of Particle Swarm Algorithm

Qi Kang[1], Lei Wang[1], Qi-di Wu[1]

[1] Control Department of Tongji University, No.1239, Siping Road, Shanghai, China 200092,
E-mail: kangqi_kz@hotmail.com

Abstract

This paper introduces a novel fuzzy adaptive optimization strategy (FAOPSO) for the particle swarm algorithm. Initially, to avoid falling into local optimums, the information of multi-optimum distribution state is introduced into the particle swarm movement programming. However, in this kind of multi-optimum static programming mode (MSPPSO), the programming proportion factor of multi-optimum cannot be dynamically adjusted in the optimization process. On the basis of MSPPSO, a kind of fuzzy adaptive programming strategy based on double-variable and single-dimensional fuzzy control structure is proposed and the performance of FAOPSO is also observed. The proposed approach is validated by function optimization problem from the standard literature. Simulation results indicated that the approach is highly competitive for its better general convergence performance.

Keywords. Particle swarm algorithm, multi-optimum, fuzzy programming

1 Introduction

Particle swarm optimization (PSO) algorithm is a population-based heuristic global optimization technology first introduced by Kennedy and Eberhart[1] in 1995. Its basic idea was based on the simulation of simplified animal social behaviors such as fish schooling, bird flocking, etc. In PSO algorithm, the individual is called particle which has not mass and volume, and the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience and the flying experience of the other particles in the search space. The PSO algorithm is becoming very popular due to its simplicity of implementation and ability to quickly converge to a reasonably good solution[2~3] . Now, PSO algorithm is effectively applied in power system optimization, traffic planning, engineering design and optimization, and computer system etc[4~7].

Since the introduction of the PSO algorithm in 1995, there has been a considerable amount of work done in developing the original version of PSO[8~14], through empirical simulations, in the integration of its self-adaptation, parameter selecting, swarm topology and integrating with other intelligent optimizing methods. In this paper, we present a novel fuzzy adaptive optimization strategy based on double-variable and single-dimensional fuzzy control structure for particle swarm optimization algorithm, called FAOPSO, in which the proportion factor of multi-optimum programming can be dynamically adjusted in the optimization process. Our current proposal is an improved version of the static programming PSO (MSPPSO) reported in paper [13], in which the knowledge of multi-modal distribution state is introduced into the particle swarm movement programming that avoid falling into local optimums and improves the exploratory capabilities of the standard algorithm.

FAOPSO is validated using function optimization problem from the standard literature and compared against three other algorithm modes: standard PSO (SPSO), the fuzzy PSO (FPSO), and the MSPPSO. In section 2, we will introduce them briefly.

2 Some Previous Work on PSO

In standard version of PSO (SPSO) [11], each particle represents a possible solution to the optimization task at hand. Assume swarm size is N, the position vector and the velocity vector of in *D*-dimension space coordinate of every particle can be indicated as $\mathbf{X}_i = (x_{i1}, \ldots, x_{id}, \ldots x_{iD})$ and $\mathbf{V}_i = (\mathrm{v}_{i1}, \ldots, \mathrm{v}_{id}, \ldots \mathrm{v}_{iD})$ respectively.

Then, the flight velocity and the new position of particle i $(i = 1 \sim N)$ for the next fitness evaluation in dth $(d = 1 \sim D)$ dimensional subspace are calculated using the following two equations:

$$v_{id} = \omega v_{id} + c_1 rand_1()(p_{id} - x_{id}) + c_2 rand_2()(p_{gd} - x_{id}) \quad (1)$$

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

where p_{id} is the record of the former best position of current particle and similarly, the difference between p_{id} and current particle is also applied to set the directional randomly movement of current particle. p_{gd} is the record of the former best position of the whole swarm, the difference between p_{gd} and the current position of particle i is applied to alter the incremental component of the movement toward swarm optimum. Acceleration coefficients c_1 and c_2 determine the relative influence of the social and cognition components, and are often both set to the same value to give each component (the cognition and social learning rates) equal weight. The variable ω is called the *inertia weight*, which is changed linearly with the running time:

$$\omega = K_1 + (K_2 - K_1)t / T \quad (3)$$

where T is the total cycle index, t is the cycle index of current computation , and K_1 , K_2 are constants which indicate the border of changing ω .

Initially, a population of particles is generated with random positions, and then random velocities are assigned to each particle. The fitness of each particle is then evaluated according to an objective function. At each generation, the velocity of each particle is calculated according to equation (1) and the position for the next function fitness evaluation is updated according to equation (2). Each time if a particle finds a better position than the previously found best position, its location is stored in memory. Generally, each particle accelerates in the direction of its own personal best solution found so far, as well as in the direction of the global best position discovered so far by any of the particles in the swarm. This means that if a particle discovers a promising new solution, all the other particles will move closer to it, exploring the region more thoroughly in the process.

On the basis of the above standard mode (SPSO), Shi Y H and Eberhart R C[12] presented a fuzzy system successfully implemented to dynamically adapt the inertia weight of the particle swarm optimization algorithm, called FPSO. Here, we do not present it in detail.

Multi-optimum static programming mode (MSPPSO) [13]is proposed by the authors. From simulation results, we found that the initial population distribution affects the performance largely local optimal value in PSO. To solve this problem, the distributional characteristics of multi-optimum value of the whole swarm should be paid more attention. Therefore, besides its former optimum position, the M optimum values of the swarm were introduced into multi-variant programming of the movement of current particle and the velocity of the particle can be determined by following formula:

$$v_{id} = \omega v_{id} + \sum_{k=1}^{M} c_k rand_k()(p_{k,id} - x_{id}) \tag{4}$$

where $p_{k,id}$ is the kth optimum ranked in the whole swarm, c_k denotes the instruction factor (or called "programming coefficient") of M optimum values of the swarm, and $rand_k()$ indicate their matching random quantity respectively. Although the comparison of multi-variant optimum value is added in velocity computation to some extent in such movement pattern, the ability to avoid falling into local optima is strengthened with such expense.

3 Fuzzy Adaptive Optimization Strategy of PSO

The searching process of particle swarm optimization is a nonlinear and dynamic process. Therefore, when the environment itself is dynamically changed over the time, the algorithm should be able to adapt dynamically to the changing environment. The static programming strategy obtains the better programming coefficient c_k through plenty of experiments. Although this method improves the general convergence performance compared with the standard algorithm, the programming coefficient c_k cannot be adjusted dynamically to the current optimization ability; therefore, its adaptive ability is poor and its general convergence performance is limited to some extent.

In this paper, we try to carry out the intelligent multi-optimum programming of the PSO. During the implement of intelligent programming, fuzzy theory was introduced into PSO again, and a fuzzy adaptive system implemented to multi-optimum dynamic programming in PSO algorithm. The foundation of the fuzzy system is that after fuzzy evaluating current optimization performance, the current experience about the optimization performance influenced by each optimum is used to adjust the programming coefficient of multi-optimum adaptively in the optimization process, which contributes to realize the adaptive programming instruction on the particle swarm.

In this system, such a kind of double-variable and single-dimensional fuzzy control structure as **Fig.1** is adopted.

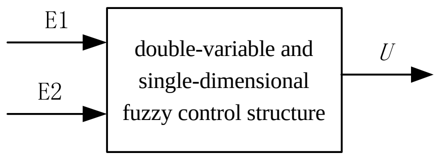

E1
E2
double-variable and single-dimensional fuzzy control structure
U

Fig.1 Double-variable and single-dimensional fuzzy control structure

In the algorithm defined by the authors, the influence of particle swarm movement caused by each optimum depends on the fuzzy distance between the current position of each particle and its position according to each optimum information value. Therefore, we can determine the instruction output of each optimum. The above distance information can be inputs of the fuzzy instruction controller, after reasoning of some intelligent fuzzy rules, output the fuzzy instruction variable, and the dynamic instruction ratio relationship among these multi-optimum optimization can be obtained through defuzzification.

Here, defining the distance between current position and each optima position on each dimension for each particle as: $E_{k,id} = p_{k,id} - x_{id}$. Then, formula (4) can be described as formula (5):

$$v_{id} = wv_{id} + \sum_{k=1}^{M} rand_k()c_k E_{k,id} \tag{5}$$

In this paper, the binary optima fuzzy programming based dynamic step increment computation pattern is adopted in simulation, $M = 2$ in the equation（5）, $p_{1,id}$ and $p_{2,id}$ are optimum and sub-optimum of the whole swarm respectively. Through consider the historical experience about the algorithm performance influenced by $E_{1,id}$ and $E_{1,id}$, establish the fuzzy rules to implement the adaptive dynamic programming of the binary optima information instruction mode in particle swarm.

In the fuzzy instruction mode of binary optima information, the input variables are $E_{1,id}$ and $E_{1,id}$, and the output variable is the fuzzy change value of the ratio between the programming coefficients of binary optima information: $\Delta c_1 / c_2$.

In the process of the fuzzification of input and output variables, we selected the same fuzzy sets: (NB，NM，NS，Z，PS，PM，PB), in which, NB denotes “Negative Big”, NM is “Negative Medium”, NS is “Negative Small”, Z is ”Zero”, PS is “Positive Small”, PM is ”Positive Medium” and PB is “Positive Big”. The sketch map of the membership functions is Figure 2.

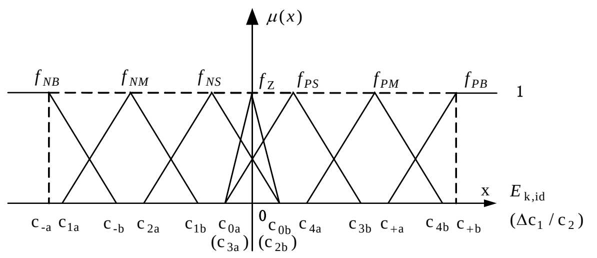

$\mu(x)$
f_{NB}
f_{NM}
f_{NS}
f_Z
f_{PS}
f_{PM}
f_{PB}
1
x
$E_{k,id}$
$(\Delta c_1 / c_2)$
0
c_{-a} c_{1a} c_{-b} c_{2a} c_{1b} c_{0a} (c_{3a}) c_{0b} (c_{2b}) c_{4a} c_{3b} c_{+a} c_{4b} c_{+b}

Fig.2 Sketch map of membership function for fuzzy variables

The authors established 49 dynamic fuzzy adjustment rules of multi-optimum information based on experience (see **Tab.1**).

Tab.1 Fuzzy Rules of the Fuzzy System (Output is $\Delta c_1 / c_2$)

E_1 / E_2	*NB*	*NM*	*NS*	*Z*	*PS*	*PM*	*PB*
NB	*PS*	*PS*	*NM*	*NS*	*NS*	*Z*	*PS*
NM	*PM*	*PS*	*NS*	*NS*	*NS*	*Z*	*PS*
NS	*PB*	*PM*	*Z*	*NM*	*Z*	*PS*	*PM*
Z	*PB*	*PB*	*PM*	*Z*	*PM*	*PB*	*PB*
PS	*PB*	*PB*	*Z*	*NM*	*PS*	*PS*	*PB*
PM	*PM*	*PS*	*Z*	*NS*	*Z*	*PM*	*PB*
PB	*PS*	*Z*	*NS*	*NS*	*NS*	*PB*	*PB*

The dynamic adjustment value of the ratio between the programming coefficients of binary optima information is obtained by the defuzzification of the fuzzy output.

We can describe the fuzzy adaptive optimization strategy of PSO (FAOPSO) as follows (**Fig.3** represents the flow char of FAOPSO):

Step.1 Initialize particle swarm in the way adopted in MSPPSO[13];

Step.2 Evaluate the adaptive value of each particle by computing the objective function;

Step.3 Compute he best position (p_{best}) for each particle, comparing and sorting all the current best positions, and finding the M optimums;

Step.4 Detect the distance information: E_1(the distance between current position and the optimum position) and E_2 (the distance between current position and the sub-optimum position) and inputting the fuzzy programming system;

Step.5 Output the change value of the ratio between the programming coefficients of binary optima information: $\Delta c_1 / c_2$ after fuzzy computation and defuzzification, and update the current c_1 、 c_2 ;

Step.6 Detect if particle enters the optimal neighbor area or not? If it is true, then update the velocity and position of particle according to multi-optimum programming rules and update multi-optimum; else update the velocity and position according to standard PSO algorithm

Step.7 Detect the terminate conditions (reaching the maximal generation or finding the idea optimum). If the terminate conditions was met, end the algorithm, or continue the computation.

4 Experimental Settings and Simulation for Benchmark Testing

A. Benchmarks

In the simulation comparison work of this paper, two well-known Benchmarks: Rosenbrock function and Griewank function were used to evaluate the performance. The first function is simple unimodal function whereas the second function is multimodal function designed with a considerable amount of local minima. Simulations were carried out to find the global minimum of each function. Both benchmarks used are given in **Tab. 2**.

Tab. 2 Benchmarks for simulation

Name of function	Mathematical representation	Range of search
Rosenbrock function	$f_1(x)=\sum_{i=1}^{n}[100(x_{i-1}-x_i^2)^2+(x_i-1)^2]$	$(-100,100)^n$
Griewank function	$f_2(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}x_i)+1$	$(-600,600)^n$

B Population Initialization

During the early stages of the development of the PSO algorithm, symmetric initialization was widely used, where the initial population is uniformly distributed in the entire search space. Whereas, since the benchmarks used in this paper have the global minimum close to the origin of the search space, we use the asymmetric initialization method to observe the performance of the new development introduced in this paper. **Tab. 3** shows the range of population initialization and the maximum velocity with the limitation of $V_{\max}= X_{\max}$ for the benchmarks considered in this paper.

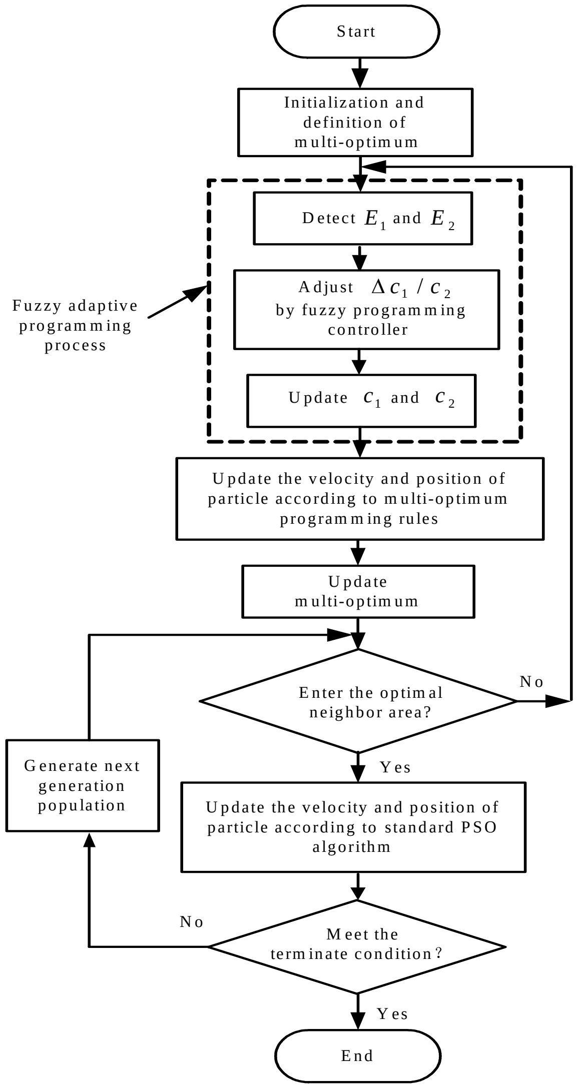

Start
Initialization and definition of multi-optimum
Detect E_1 and E_2
Adjust $\Delta c_1 / c_2$ by fuzzy programming controller
Update c_1 and c_2
Fuzzy adaptive programming process
Update the velocity and position of particle according to multi-optimum programming rules
Update multi-optimum
Enter the optimal neighbor area?
No
Yes
Generate next generation population
Update the velocity and position of particle according to standard PSO algorithm
No
Meet the terminate condition?
Yes
End

Fig.3 The flow char of FAOPSO

Tab. 3 Initialization range and maximum velocity for benchmarks

Name of function	Range of initialization	$V_{\max}$
Rosenbrock function	$(15,30)^n$	100

Griewank function	$(300,600)^n$	600

C Experimental settings

Simulations were carried out to observe the quality of the optimum solution of the new algorithm introduced in this paper. Both benchmarks were tested with dimensions 10, 20, and 30. A different number of maximum generations (Gmax) is used according to the complexity of the problem under consideration. For each function, 100 trials were carried out and the average optimal values are presented. In addition, the basic parameters setting are: The size of the particle swarm is 80; $k_1 = 0.4$, $k_2 = 0.9$.

For both benchmarks, the boundary settings of fuzzy variables $E_{1,id}$, $E_{2,id}$, and $\Delta c_1 / c_2$ are presented in **Tab. 4**.

Tab. 4 The boundary settings of fuzzy variables $E_{1,id}$, $E_{2,id}$, and $\Delta c_1 / c_2$

Fun	Fuzzy Variables	NB	NM	NS	Z	PS	PM	PB
f_1	$E_{1,id}$	(-100, -50)	(-60, -20)	(-30, 5)	(-5, 5)	(-5, 30)	(20, 60)	(50, 100)
	$E_{2,id}$	(-100, -50)	(-60, -20)	(-30, 5)	(-5, 5)	(-5, 30)	(20, 60)	(50, 100)
	$\Delta c_1 / c_2$	(-0.1, -0.04)	(-0.045, -0.015)	(-0.025, 0.005)	(-0.005, 0.005)	(-0.005, 0.025)	(0.015, 0.045)	(0.04, 0.1)
f_2	$E_{1,id}$	(-600, -300)	(-400, -100)	(-125, 25)	(-25, 25)	(-25, 125)	(100, 400)	(300,600)
	$E_{2,id}$	(-600, -300)	(-400, -100)	(-125, 25)	(-25, 25)	(-25, 125)	(100, 400)	(300,600)
	$\Delta c_1 / c_2$	(-0.1, -0.04)	(-0.045, -0.015)	(-0.025, 0.005)	(-0.005, 0.005)	(-0.005, 0.025)	(0.015, 0.045)	(0.04, 0.1)

5 Results from Benchmarks Simulations

We observed the performance in terms of quality of the average optimum value for 100 trials, of the new algorithm developed in this paper under the same parameter setting except c_1, c_2. **Tab.5** and **Tab.6** show the simulation results and compares with SPSO, FPSO and MSPPSO for two benchmarks. In the table, figures in bold represent the comparatively best values.

Tab.5 Comparison of the average values for 100 trials

(Rosenbrock function)

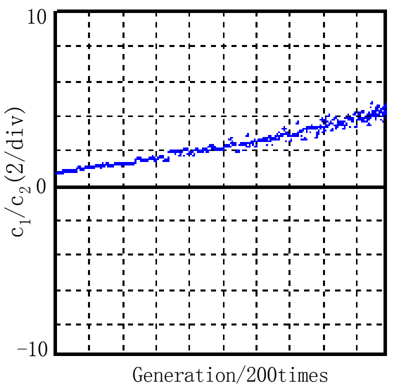
Dim	Gmax	SPSO[11]	FPSO[12]	MSPPSO[13]
10	1000	36.2945	15.8165	8.0234
20	1500	87.2802	45.9999	32.8245
30	2000	205.5590	124.418	53.8489
Dim	Gmax	FAOPSO $c_1/c_2(0)=0.2$	FAOPSO $c_1/c_2(0)=2.5$	FAOPSO $c_1/c_2(0)=1$
10	1000	6.0449	6.4582	**5.0035**
20	1500	14.5275	13.9667	**13.7362**
30	2000	**30.5783**	32.8956	30.7947

Tab.6 Comparison of the average values for 100 trials

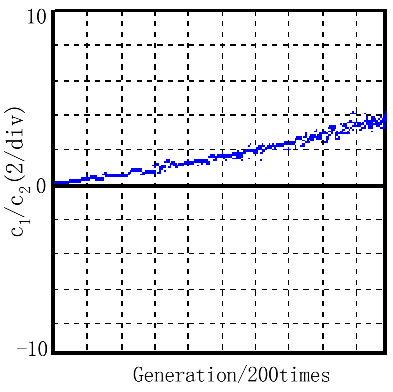
(Griewank function)

Dim	Gmax	SPSO[11]	FPSO[12]	MSPPSO[13]
10	1000	0.07600	0.06832	0.08759
20	1500	0.02880	0.02596	0.02410
30	2000	0.01280	0.01495	0.01037
Dim	Gmax	FAOPSO $c_1/c_2(0)=0.2$	FAOPSO $c_1/c_2(0)=2.5$	FAOPSO $c_1/c_2(0)=1$
10	1000	0.8460	0.8276	**0.8115**
20	1500	0.02456	**0.02390**	0.02478
30	2000	0.00983	0.01047	**0.00964**

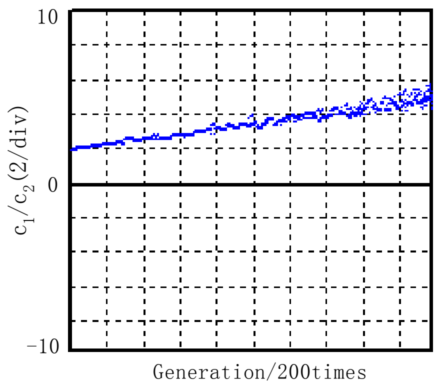
The typical dynamic curves of c_1/c_2 are shown in **Fig.4 (a~c)** (30 dimensional function optimization, and initialization value is separately $c_1/c_2(0)=1$ 、$c_1/c_2(0)=0.2$ and $c_1/c_2(0)=2$).



(a) $c_1/c_2(0)=1$



(b) $c_1/c_2(0)=0.2$



(c) $c_1/c_2(0)=2.5$

Fig.4 Typical dynamic curves of c_1/c_2 in FAOPSO

In **Tab.5** and **Tab.6**, we compare the performance of new algorithm in different initialization of c_1/c_2 with SPSO, FPSO and MSPPSO for Rosenbrock function and Griewank function. From the results, FAOPSO always has the best average optimum solution (figures in bold) in different simulations.

Fig.4 displays the typical dynamic variation of the ratio between the programming coefficients of binary optima information c_1/c_2 from the different initial setting values ($c_1/c_2(0)=1$、 $c_1/c_2(0)=0.2$ and $c_1/c_2(0)=2$) in the process of 30 dimensional function optimization.

Generally, the average optimum solutions and typical dynamic curves of c_1/c_2 above fully prove that the general optimization performance of the new algorithm is significantly good. From the average optimization results, we can also see that the performance of the algorithm was hardly not influenced by the initialization value of

c_1 and c_2 (or c_1/c_2), that is to say, even when the sub-optimum is setting to have more attraction to particle than the optimum at initialization ($c_1/c_2(0)<1$), FAOPSO can also find good optimization result. From Fig.4, we can find that c_1/c_2 reach a similar value ($c_1/c_2=5$) from different initialization values, which is consistent with that $c_1/c_2=5$ is a preferable setting in the static multi-optimum programming mode.

Although the new fuzzy programming strategy is applied to the dynamic programming of particle swarm optimization and the general optimization performance of the algorithm is improved, the fuzzy rules used in this paper are got from the specific problem, and can not apply them to other type of optimization problems directly, that is to say, when the algorithm is used for other kinds of problems, the fuzzy rules should be adjusted proper according to specific characteristics of the given function in order to achieve the best effects of speed and convergence.

6 Conclusions

We have described a novel fuzzy adaptive optimization strategy for particle swarm optimization algorithm aiming to dynamically adjust the proportion factor of multi-optimum programming so that it can improve the performance in terms of the optimal solution within a reasonable number of generations. Initially, we introduced fuzzy control method into PSO and presented a kind of fuzzy control strategy on the basis of multi-optimum static programming mode. Then we designed the fuzzy system with the double-variable and single-dimensional fuzzy control structure, and validated the performance using two benchmarks. The average results compared with other methods show that the new algorithm FAOPSO has significant performance of convergence. Further, in the view of the authors, the applications of fuzzy theory in particle swarm algorithm optimization with intelligent characteristics can be discussed further in future and the convergence pattern, dynamic and steady-state performances of the algorithm can be improved to specific complex optimization functions.

Acknowledgment

The work of this paper is supported by National Science Foundation of China (70271035, 60104004 , 60374005) , Shanghai Riding-star Foundation for Young Researchers (03QG14053) and sub-item of National 973 project (2002CB312202).

References

[1] Kennedy J, Eberhart R C. “Particle swarm optimization”, Proceedings of IEEE International Conference on Neural Networks, Perth, Australia: IEEE Piscataway, 1995, 1942-1948.
[2]Eberhart R C, Shi Y. “Particle swarm optimization: developments, applications and resources”, Proceedings of Congress on Evolutionary Computation, Soul, Seoul Korea: IEEE, 2001, 81-86.
[3] Eberhart R C, Shi Y. “Comparing inertia weights and constriction factors in particle swarm optimization”, Proceedings of Congress on Evolutionary Computation, California, USA: IEEE, 2000, 84-88.
[4]Esmin A, Lambert-Torres G, Zambroni de Souza A C. “A hybrid particle swarm optimization applied to loss power minimization”, IEEE Transactions on Power Systems, 2005, 20(2): 859-866.
[5]Sousa T, Silva A. “Particle swarm based data mining algorithms for classification tasks”, Parallel Computing, 2004, 30(5-6): 767-783.
[6] Xia W J, Wu Z M, Zhang W, et al. “Applying particle swarm optimization to job-shop scheduling problem”, Chinese Journal of Mechanical Engineering (English Edition), 2004, 17(3):437-441.
[7] Gaing Z L. “A particle swarm optimization approach for optimum design of PID controller in AVR system”, IEEE Transactions on Energy Conversion, 2004, 19(2): 384-391.
[8] Clerc M. “The swarm and the queen: towards a deterministic and adaptive particle swarm optimization”, Proceedings of Congress on Evolutionary Computation, Washington DC, USA: IEEE Piscataway, 1999, 1951-1957.
[9] Fan Huiyuan. “A modification to particle swarm optimization algorithm”, Engineering Computations, 2002, 19 (8): 970-989.
[10] Kennedy J. “Particle swarm: social adaptation of knowledge”, Proc. IEEE. Int. Conf. on Evolutionary Computation, Indianapolis: IEEE Piscataway, 1997, 303-308.
[11] Shi Y, Eberhart R C. “Empirical study of particle swarm optimization”, Proceedings of the Congress on Evolutionary Computation, Washington DC, USA: IEEE Piscataway, 1999, 1945-1950.
[12] Shi Y, Eberhart R. “Fuzzy adaptive particle swarm optimization”, Proceedings of Congress on Evolutionary Computation, Soul, Seoul Korea: IEEE, 2001, 101-106.
[13] Wang Lei, Kang Qi, Wu Qidi. “Multi-optimum programming based particle swarm optimization algorithm and its application in multi-dimensional & multi-modal function optimization”, IEEE Proceedings of Conference on Control Application, Taipei: IEEE, 2004, 149-152.
[14] Wang Lei, Kang Qi, Wu Qidi. “Fuzzy Logic based Multi-Optimum Programming in Particle Swarm Optimization”, IEEE International Conference on Networking, Sensing and Control, Tucson, Arizona, USA, 2005, 473-477.

Qi Kang received the B.S. and M.S. degrees from the Institute of Electronics and Information Engineering at Tongji University, Shanghai, China, in 2002 and 2005, respectively.

Currently, he is a Ph.D candidate in the Control Department at Tongji University, Shanghai, China. His research interests are swarm intelligence theory and evolutionary computation.

Lei Wang received the M.S. and Ph.D. degrees from the Institute of Electronics and Information Engineering at Tongji University, Shanghai, China, in 1995 and 1998, respectively.

Currently, he is an Associate Professor in the Control Department at Tongji University, Shanghai, China. His research interests are intelligent control theory and artificial intelligence.