Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2012, Article ID 207318, 11 pages
doi:10.1155/2012/207318

Research Article

Adaptive Parameters for a Modified Comprehensive Learning Particle Swarm Optimizer

Yu-Jun Zheng,[1] Hai-Feng Ling,[2] and Qiu Guan[1]

[1] *College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, Zhejiang 310023, China*
[2] *Engineering Institute of Engineering Corps, PLA University of Science and Technology, Nanjing, Jiangsu 210007, China*

Correspondence should be addressed to Yu-Jun Zheng, yujun.zheng@computer.org

Received 5 October 2012; Accepted 25 November 2012

Academic Editor: Sheng-yong Chen

Particle swarm optimization (PSO) is a stochastic optimization method sensitive to parameter settings. The paper presents a modification on the comprehensive learning particle swarm optimizer (CLPSO), which is one of the best performing PSO algorithms. The proposed method introduces a self-adaptive mechanism that dynamically changes the values of key parameters including inertia weight and acceleration coefficient based on evolutionary information of individual particles and the swarm during the search. Numerical experiments demonstrate that our approach with adaptive parameters can provide comparable improvement in performance of solving global optimization problems.

1. Introduction

The complexity of many real-world problems has made exact solution methods impractical to solve them within a reasonable amount of time and thus gives rise to various types of nonexact metaheuristic approaches [1–3]. In particular, swarm intelligence methods, which simulate a population of simple individuals evolving their solutions by interacting with one another and with the environment, have shown promising performance on many difficult problems and have become a very active research area in recent years [4–11]. Among these methods, particle swarm optimization (PSO), initially proposed by Kennedy and Eberhart [4], is a population-based global optimization technique that involves algorithmic mechanisms similar to social behavior of bird flocking. The method enables a number of individual solutions, called particles, to move through the solution space and towards the most promising area for optimal solution(s) by stochastic search. Consider a D-dimensional optimization problem as follows:

$$\begin{aligned} \min \quad & f(\mathbf{x}), \quad \mathbf{x} = [x_1, x_2, \ldots x_D]^{\mathrm{T}} \\ \text{s.t.} \quad & x_{iL} \le x_i \le x_{iU}, \quad 0 < i \le D. \end{aligned} \tag{1.1}$$

In the D-dimensional search space, each particle i of the swarm is associated with a position vector $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ and a velocity vector $\mathbf{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, which are iteratively adjusted by learning from a local best pbest_i found by the particle itself and a current global best gbest found by the whole swarm:

$$\mathbf{v}_i^{(t+1)} = \mathbf{v}_i^{(t)} + c_1 r_1 \left(\text{pbest}_i^{(t)} - \mathbf{x}_i^{(t)}\right) + c_2 r_2 \left(\text{gbest}^{(t)} - \mathbf{x}_i^{(t)}\right), \tag{1.2}$$

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \mathbf{v}_i^{(t+1)}, \tag{1.3}$$

where c_1 and c_2 are two acceleration constants reflecting the weighting of "cognitive" and "social" learning, respectively, and r_1 and r_2 are two distinct random numbers in $[0, 1]$. It is recommended that $c_1 = c_2 = 2$ since it on average makes the weights for cognitive and social parts both to be 1.

To achieve a better balance between the exploration (global search) and exploitation (local search), Shi and Eberhart [12] introduce a parameter named inertia weight w to control velocity, which is currently the most widely used form of velocity update equation in PSO algorithms:

$$\mathbf{v}_i^{(t+1)} = w\mathbf{v}_i^{(t)} + c_1 r_1 \left(\text{pbest}_i^{(t)} - \mathbf{x}_i^{(t)}\right) + c_2 r_2 \left(\text{gbest}^{(t)} - \mathbf{x}_i^{(t)}\right). \tag{1.4}$$

Empirical studies have shown that a large inertia weight facilitates exploration and a small inertia weight one facilitates exploitation and a linear decreasing inertia weight can be effective in improving the algorithm performance:

$$w^{(t)} = w_{\min} + \frac{t_{\max} - t}{t_{\max}} (w_{\max} - w_{\min}), \tag{1.5}$$

where t is the current iteration number, $t_{\max}$ is the maximum number of allowable iterations, and $w_{\max}$ and $w_{\min}$ are the initial value and the final value of inertia weight, respectively. It is suggested that $w_{\max}$ can be set to around 1.2 and $w_{\min}$ around 0.9, which can result in a good algorithm performance and remove the need for velocity limiting.

PSO is conceptually simple and easy to implement, and has been proven to be effective in a wide range of optimization problems [13–20]. Furthermore, It can be easily parallelized by concurrently processing multiple particles while sharing the social information [21, 22]. Kwok et al. [23] present an empirical study on the effect of randomness on the control coefficients of PSO, and the results show that the selective and uniformly distributed random coefficients perform better on complicate functions.

In recent years, PSO has attracted a high level of interest, and a number of variant PSO methods (e.g., [24–32]) have been proposed to accelerate convergence speed and avoid local optima. In particular, Liang et al. develop a comprehensive learning particle swarm optimizer (CLPSO) [26], which uses all other particles' historical best information (instead of pbest and gbest) to update a particle's velocity:

$$v_{i,d}^{(t)} = w^{(t)} v_{i,d}^{(t)} + c r_{i,d} \left(\text{pbest}_{fi(d),d}^{(t)} - x_{i,d}^{(t)} \right), \tag{1.6}$$

where $\text{pbest}_{fi(d),d}$ can be the dth dimension of any particle's pbest (including pbest_i itself), and particle $fi(d)$ is selected based on a learning probability P_{C_i}. The authors suggest a tournament selection procedure that randomly chooses two particles and then select one with the best fitness as the exemplar to learn from for that dimension. Note that CLPSO has only one acceleration coefficient c which is normally set to 1.494, and it limits the inertia weight value in the range of [0.4, 0.9].

According to empirical studies [29, 30, 33], CLPSO has been shown to be one of the best performing PSO algorithms, especially for complex multimodal function optimization. In [34] a self-adaptation technique is introduced to adaptively adjust the learning probability, and the historical information is used in the velocity update equation, which effectively improve the performance of CLPSO on single modal problems.

Wu et al. [35] adapt the CLPSO algorithm by improving the search behavior to optimize the continuous externality for antenna design. In [36] Li and Tan present a hybrid strategy to combine CLPSO with Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, which defines a local diversity index to indicate whether the swarm enters into an optimality region in a high probability. They apply the method to identify multiple local optima of the generalization bounds of support vector machine parameters and obtain satisfying results. However, to the best of our knowledge, modifications of CLPSO based on adaptive inertia weight and acceleration coefficient have not been reported.

In this paper we propose an improved CLPSO algorithm, named CLPSO-AP, by introducing a new adaptive parameter strategy. The algorithm evaluates the evolutionary states of individual particles and the whole swarm, based on which values of inertia weight and acceleration coefficient are dynamically adjusted to search more effectively. Numerical experiments on test functions show that our algorithm can significantly improve the performance of CLPSO.

In the rest of the paper, Section 2 presents our PSO method with adaptive parameters, Section 3 presents the computational experiments, and Section 4 concludes with discussion.

2. The CLPSO-AP Algorithm

2.1. Adaptive Inertia Weight and Acceleration Coefficient

To provide an adaptive parameter strategy, we first need to determine the situation of each particle at each iteration. In this paper, two concepts are used for this purpose. The first one considers whether or not a particle i improves its personal best solution at the tth iteration (in the paper we assume the problem is to minimize the objective function without loss of generality):

$$s_i^{(t)} = \begin{cases} 1 & \text{if } f\left(\text{pbest}_i^{(t)}\right) < f\left(\text{pbest}_i^{(t-1)}\right) \\ 0 & \text{else.} \end{cases} \tag{2.1}$$

And the second considers the particle's "rate of growth" from the $(t-1)$th iteration to the tth iteration:

$$\gamma_i^{(t)} = \frac{f\left(\mathbf{x}_i^{(t)}\right) - f\left(\mathbf{x}_i^{(t-1)}\right)}{\left|\mathbf{x}_i^{(t)} - \mathbf{x}_i^{(t-1)}\right|}, \tag{2.2}$$

where $|\mathbf{x}_i^{(t)} - \mathbf{x}_i^{(t-1)}|$ denotes the Euclidean distance between $\mathbf{x}_i^{(t)}$ and $\mathbf{x}_i^{(t-1)}$:

$$\left|\mathbf{x}_i^{(t)} - \mathbf{x}_i^{(t-1)}\right| = \sqrt{\sum_{k=1}^{D}\left(x_{ik}^{(t)} - x_{ik}^{(t-1)}\right)^2}. \tag{2.3}$$

Based on (2.1), we can calculate the percentage of particles that successfully improve their personal better solutions:

$$\rho^{(t)} = \frac{\sum_{i=1}^{p} s_i^{(t)}}{p}, \tag{2.4}$$

where p is the number of particles in the swarm. This measure has been utilized in [33] and some other evolutionary algorithms such as [37]. Generally, in PSO a high ρ indicates a high probability that the particles have converged to a nonoptimum point or slowly moving toward the optimum, while a low ρ indicates that the particles are oscillating around the optimum without much improvement. Considering role of inertia weight in the convergence behavior of PSO, in the former case the swarm should have a large inertial weight and in the latter case the swarm should have a small inertial weight. Here we use the following nonlinear function to map the values of ρ to w:

$$w^{(t)} = e^{\rho^{(t)}-1}. \tag{2.5}$$

It is easy to derive that w ranges from about 0.36 to 1. The nonlinear and non-monotonous change of inertial weight can improve the adaptivity and diversity of the swarm, because the search process of PSO is highly complicated on most problems.

Most PSO algorithms use constant acceleration coefficients in (1.4). But it deserves to note that Ratnaweera et al. [38] introduce a time-varying acceleration coefficient strategy where the cognitive coefficient is linearly decreased and the social coefficient is linearly increased. The basic CLPSO also uses a constant acceleration coefficient in (1.6), where c reflects the weighting of stochastic acceleration term that pulls each particle i towards the personal best position of particle $fi(d)$ at each dimension d. Considering the measure defined in (2.2), a large value of $\gamma_i^{(t)}$ indicates that at t time, particle i falls rapidly in the search space and gets a considerable improvement on the fitness function; thus it is reasonable to anticipate that the particle may also gain much improvement at least at the next iteration. On the contrary, a small value of $\gamma_i^{(t)}$ indicates that particle i progresses slowly and thus needs a large acceleration towards the exemplar.

From the previous analysis, we suggest that the acceleration coefficient should be an increasing function of $\gamma_i^{(t)}/\Gamma^{(t)}$, where $\Gamma^{(t)}$ is the SRSS of the rates of growth of all the particles in the swarm:

$$\Gamma^{(t)} = \sqrt{\sum_{j=1}^{p}\left(\gamma_j^{(t)}\right)^2}. \tag{2.6}$$

Based on our empirical tests, we use the following function to map the values of γ_i to c_i:

$$c_i^{(t)} = 1.8\left(1 - \frac{\gamma_i^{(t)}}{\Gamma^{(t)}}\right). \tag{2.7}$$

2.2. *The Proposed Algorithm*

Using the adaptive parameter strategy described in the previous section, the equation for velocity update for the CLPSO-AP algorithm is

$$v_{i,d}^{(t)} = w^{(t)} v_{i,d}^{(t)} + c_i^{(t)} r_{i,d}\left(\text{pbest}_{fi(d),d}^{(t)} - x_{i,d}^{(t)}\right), \tag{2.8}$$

where $w^{(t)}$ and $c_i^{(t)}$ are calculated based on (2.5) and (2.7), respectively.

Now we present the flow of CLPSO-AP algorithm as follows.

(1) Generate a swarm of p particles with random positions and velocities in the range. Let $t = 0$ and initialize $w^{(t)} = 1$ and each $c_i^{(t)} = 1$.

(2) Generate a learning probability P_{C_i} for each particle based on the following equation suggested in [10]:

$$P_{C_i} = 0.05 + 0.45\frac{\exp(10(i-1)/(p-1)) - 1}{\exp(10) - 1}. \tag{2.9}$$

(3) Evaluate the fitness of each particle and update its pbest and then select a particle with the best fitness value as gbest.

(4) For each particle i in the swarm do the following.

(4.1) For $d = 1$ to D do the next.

(4.1.1) Generate a random number rand in the range $[0, 1]$.

(4.1.2) If rand $> P_{C_i}$, let $fi(d) = i$.

(4.1.3) Else, randomly choose two other distinct particles i_1 and i_2, and select the one with better fitness value as $fi(d)$.

(4.1.4) Update the dth dimension of the particle's velocity according to (2.8).

(4.2) Update the particle's position according to (1.3).

(4.3) Calculate $s_i^{(t)}$ and $r_i^{(t)}$ for the particle according to (2.1) and (2.2), respectively.

(5) Calculate $w^{(t)}$ of the swarm based on (2.4) and (2.5).

(6) Calculate $\Gamma^{(t)}$ based on (2.5), and then calculate $c_i^{(t)}$ for each particle i based on (2.7).

(7) Let $t = t + 1$. If $t = t_{\max}$ or any other termination condition is satisfied, the algorithm stops and returns gbest.

(8) Go to step 3.

In Step (7), other termination conditions can be that a required function value is obtained, or all the particles converge to a stable point.

3. Numerical Experiments

In order to evaluate the performance of the proposed algorithm, we choose a set of well-known test functions as benchmark problems, the definitions of which are listed in the Appendix section. The search ranges, optimal points and corresponding fitness values, and required accuracies are shown in Table 1.

We comparatively execute the basic PSO algorithm, CLPSO algorithm, and our CLPSO-AP algorithm on the test functions with 10 and 30 dimensions, where each experiment is run for 40 times. The parameter settings for the algorithms are given in Table 2.

We use the mean best fitness value and the success rate (with respect to the required accuracy shown in Table 1) as two criteria for measuring the performance of the algorithms. The experimental results (averaged over 40 runs) of which are presented in Tables 3, 4, 5, and 6 respectively.

As we can see from the experimental results, for all 10D and 30D dimensional problems, CLPSO-AP performs better than CLPSO in terms of both mean best values and success rates. Among the seven test functions, Ackley function is the only one on which CLPSO performs no better than the basic PSO However, CLPSO-AP performs better than basic PSO on both the 10D and 30D Ackley functions, and thus CLPSO-AP also outperforms basic PSO on all benchmark problems. It also deserves to note the 10D Rosenbrock function, for which CLPSO and most of the other PSO variants hardly succeed [26, 39, 40] while our CLPSO-AP algorithm gains a 10% success rate. Except for the 30D Rosenbrock function, CLPSO-AP successfully obtains the global optimum for all the other functions. In summary, our algorithm performs very well and overwhelms the other two algorithms on all of the test problems.

4. Conclusion

CLPSO has been shown to be one of the best performing PSO algorithms. The paper proposes a new improved CLPSO algorithm, named CLPSO-AP, which uses evolutionary information of individual particles to dynamically adapt the inertia weight and acceleration coefficient at each iteration. Experimental results on seven test functions show that our algorithm can significantly improve the performance of CLPSO. Ongoing work includes applying our algorithm to intelligent feature selection and lighting control in robotics [41–43] and extending the adaptive strategy to other PSO variants, including those for fuzzy and/or multiobjective problems [44, 45].

Table 1: Detailed information of the test functions used in the paper.

No.	Function	Search range	$\mathbf{x}^*$	$f(\mathbf{x}^*)$	Required accuracy
f_1	Sphere	$[-100, 100]^D$	$[0, 0, \ldots, 0]$	0	0.000001
f_2	Rosenbrock	$[-2.048, 2.048]^D$	$[1, 1, \ldots, 1]$	0	0.01
f_3	Schwefel	$[-500, 500]^D$	$[420.96, 420.96, \ldots, 420.96]$	0	0.01
f_4	Rastrigin	$[-5.12, 5.12]^D$	$[0, 0, \ldots, 0]$	0	0.01
f_5	Griewank	$[-600, 600]^D$	$[0, 0, \ldots, 0]$	0	0.000001
f_6	Ackley	$[-32.768, 32.768]^D$	$[0, 0, \ldots, 0]$	0	0.000001
f_7	Weierstrass	$[-0.5, 0.5]^D$	$[0, 0, \ldots, 0]$	0	0.000001

Table 2: Parameter settings of the algorithms.

Algorithm	Inertial weight	Acceleration coefficient(s)	10D functions		30D functions	
			p	$t_{\max}$	p	$t_{\max}$
PSO	$[0.4, 1.2]$	$c_1 = c_2 = 1.49445$	10	5000	30	10000
CLPSO	$[0.4, 0.9]$	$c = 1.49445$	10	5000	30	10000
CLPSO-AP	N/A	N/A	10	5000	30	10000

Table 3: The mean best values obtained by the algorithms for 10D problems.

Function	PSO	CLPSO	CLPSO-AP
Sphere	$3.239E-27$	$8.138E-79$	$5.479E-96$
Rosenbrock	$2.172E+00$	$2.005E+00$	$1.106E+00$
Schwefel	$1.048E+03$	0	0
Rastrigin	$5.774E-01$	0	0
Griewank	$8.132E-02$	$1.313E-02$	$6.467E-03$
Ackley	$5.951E-15$	$2.181E-13$	$4.707E-15$
Weierstrass	0	0	0

Table 4: The success rates obtained by the algorithms for 10D problems.

Function	PSO	CLPSO	CLPSO-AP
Sphere	100%	100%	100%
Rosenbrock	0	0	10%
Schwefel	5%	100%	100%
Rastrigin	12.5%	100%	100%
Griewank	0	1.75%	75%
Ackley	100%	100%	100%
Weierstrass	100%	100%	100%

Table 5: The mean best values obtained by the algorithms for 30-D problems.

Function	PSO	CLPSO	CLPSO-AP
Sphere	$4.900E-14$	$1.077E-57$	$3.852E-75$
Rosenbrock	$5.164E+01$	$2.158E+01$	$1.954E+01$
Schwefel	$1.425E+02$	$7.913E+01$	$6.883E+01$
Rastrigin	$4.726E+00$	$4.974E-02$	0
Griewank	$2.745E-02$	$1.616E-10$	$1.167E-12$
Ackley	$1.106E-13$	$6.594E-12$	$3.242E-14$
Weierstrass	$1.155E-14$	$1.155E-14$	$1.155E-14$

Table 6: The success rates obtained by the algorithms for 30-D problems.

Function	PSO	CLPSO	CLPSO-AP
Sphere	100%	100%	100%
Rosenbrock	0	0	0
Schwefel	0	50%	55%
Rastrigin	0	95%	100%
Griewank	37.5%	100%	100%
Ackley	100%	100%	100%
Weierstrass	100%	100%	100%

Appendix

Definitions of the Test Functions

(1) Sphere function:

$$f_1(x) = \sum_{i=1}^{D} x_i^2. \quad \text{(A.1)}$$

(2) Rosenbrock's function:

$$f_2(x) = \sum_{i=2}^{D-1} \left(100\left(x_i^2 - x_{i-1}\right)^2 + (x_i - 1)^2 \right). \quad \text{(A.2)}$$

(3) Schwefel's function:

$$f_3(x) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin\left(|x_i|^{1/2}\right). \quad \text{(A.3)}$$

(4) Rastrigin's function:

$$f_4(x) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10 \right). \quad \text{(A.4)}$$

(5) Griewank's function:

$$f_5(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \quad \text{(A.5)}$$

(6) Ackley's function:

$$f_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i) \right) + 20 + e. \quad \text{(A.6)}$$

(7) Weierstrass function:

$$f_7(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k_{\max}}\left[a^k \cos\left(2\pi b^k (x_i + 0.5)\right)\right]\right) - D\sum_{k=0}^{k_{\max}}\left[a^k \cos\left(2\pi b^k \cdot 0.5\right)\right]. \tag{A.7}$$

Acknowledgment

This work was supported in part by grants from National Natural Science Foundation (Grant no. 61020106009, 61105073, 61103140, and 61173096) of China.

References

[1] R. Chiong and T. Weise, "Special issue on modern search heuristics and applications," *Evolutionary Intelligence*, vol. 4, no. 1, pp. 1–2, 2011.

[2] S. Chen, W. Huang, C. Cattani, and G. Altieri, "Traffic dynamics on complex networks: a survey," *Mathematical Problems in Engineering*, vol. 2012, Article ID 732698, 23 pages, 2012.

[3] M. Li, W. Zhao, and S. Chen, "mBm-based scalings of traffic propagated in internet," *Mathematical Problems in Engineering*, vol. 2011, Article ID 389803, 21 pages, 2011.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, 1995.

[5] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 26, no. 1, pp. 29–41, 1996.

[6] X. Li, *A new intelligent optimization artificial fish school algorithm [doctor thesis]*, Zhejiang University, Hangzhou, China, 2003.

[7] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[8] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Advances in Swarm Intelligence*, vol. 6145 of *Lecture Notes in Computer Science*, pp. 355–364, 2010.

[9] S. Chen, Y. Zheng, C. Cattani, and W. Wang, "Modeling of biological intelligence for SCM system optimization," *Computational and Mathematical Methods in Medicine*, vol. 2012, Article ID 769702, 10 pages, 2012.

[10] X. Wen, Y. Zhao, Y. Xu, and D. Sheng, "Quasiparticle swarm optimization for cross-section linear profile error evaluation of variation elliptical piston skirt," *Mathematical Problems in Engineering*, vol. 2012, Article ID 761978, 15 pages, 2012.

[11] Y. J. Zheng, X. L. Xu, H. F. Ling, and S. Y. Chen, "A hybrid fireworks optimization method with differential evolution operators," *Neurocomputing*. In press.

[12] Y. Shi and R. Eberhart, "Modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, Anchorage, Alaska, USA, May 1998.

[13] X. D. Li and A. P. Engelbrecht, "Particle swarm optimization: an introduction and its recent developments," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 3391–3414, London, UK, 2007.

[14] W. N. Chen, J. Zhang, H. S. H. Chung, W. L. Zhong, W. G. Wu, and Y. H. Shi, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 2, pp. 278–300, 2010.

[15] R. C. Eberhart and Y. H. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 94–97, Seoul, Korea, 2001.

[16] M. Thidaa, H. L. Eng, D. N. Monekosso, and P. Remagnino, "A particle swarm optimisation algorithm with interactive swarms for tracking multiple targets," *Applied Soft Computing*. In press.

[17] S. Y. Chen, H. Tong, Z. Wang, S. Liu, M. Li, and B. Zhang, "Improved generalized belief propagation for vision processing," *Mathematical Problems in Engineering*, vol. 2011, Article ID 416963, 12 pages, 2011.

[18] M. Li, S. C. Lim, and S. Chen, "Exact solution of impulse response to a class of fractional oscillators and its stability," *Mathematical Problems in Engineering*, vol. 2011, Article ID 657839, 9 pages, 2011.

[19] C. Cattani, “Fractals and hidden symmetries in DNA,” *Mathematical Problems in Engineering*, vol. 2012, Article ID 507056, 31 pages, 2010.

[20] Y. J. Zheng and S. Y. Chen, “Cooperative particle swarm optimization for multiobjective transportation planning,” *Applied Intelligence*. In press.

[21] B. I. Koh, A. D. George, R. T. Haftka, and B. J. Fregly, “Parallel asynchronous particle swarm optimization,” *International Journal for Numerical Methods in Engineering*, vol. 67, no. 4, pp. 578–595, 2006.

[22] K. E. Parsopoulos, “Parallel cooperative micro-particle swarm optimization: a master-slave model,” *Applied Soft Computing*, vol. 12, pp. 3552–3579, 2012.

[23] N. M. Kwok, D. K. Liu, K. C. Tan, and Q. P. Ha, “An empirical study on the settings of control coefficients in particle swarm optimization,” in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 3165–3172, Vancouver, Canada, 2006.

[24] R. Mendes, J. Kennedy, and J. Neves, “The fully informed particle swarm: simpler, maybe better,” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[25] K. E. Parsopoulos and M. N. Vrahatis, “UPSO: a unified particle swarm optimization scheme,” in *Proceedings of the International Conference of Computational Methods in Sciences and Engineering (ICCMSE ’04)*, vol. 1 of *Lecture Series on Computer and Computational Sciences*, pp. 868–873, VSP International Science Publishers, 2004.

[26] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[27] N. M. Kwok, Q. P. Ha, D. K. Liu, G. Fang, and K. C. Tan, “Efficient particle swarm optimization: a termination condition based on the decision-making approach,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’07)*, pp. 3353–3360, September 2007.

[28] N. M. Kwok, G. Fang, Q. R. Ha, and D. K. Liu, “An enhanced particle swarm optimization algorithm for multi-modal functions,” in *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA ’07)*, pp. 457–462, Harbin, China, August 2007.

[29] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, “Adaptive particle swarm optimization,” *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 39, pp. 1362–1381, 2009.

[30] Y. Shi, H. Liu, L. Gao, and G. Zhang, “Cellular particle swarm optimization,” *Information Sciences*, vol. 181, no. 20, pp. 4460–4493, 2011.

[31] M. S. Leu and M. F. Yeh, “Grey particle swarm optimization,” *Applied Soft Computing*, vol. 12, no. 9, pp. 2985–2996, 2012.

[32] M. M. Noel, “A new gradient based particle swarm optimization algorithm for accurate computation of global minimum,” *Applied Soft Computing*, vol. 12, no. 1, pp. 353–359, 2012.

[33] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, “A novel particle swarm optimization algorithm with adaptive inertia weight,” *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3658–3670, 2011.

[34] J. J. Liang and P. N. Suganthan, “Adaptive comprehensive learning particle swarm optimizer with history learning,” in *Simulated Evolution and Learning*, vol. 4247 of *Lecture Notes in Computer Science*, pp. 213–220, 2006.

[35] H. Wu, J. Geng, R. Jin et al., “An improved comprehensive learning particle swarm optimization and its application to the semiautomatic design of antennas,” *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 10, pp. 3018–3028, 2009.

[36] S. Li and M. Tan, “Tuning SVM parameters by using a hybrid CLPSO-BFGS algorithm,” *Neurocomputing*, vol. 73, pp. 2089–2096, 2010.

[37] H.-G. Beyer and H.-P. Schwefel, “Evolution strategies—a comprehensive introduction,” *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.

[38] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients,” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[39] M. Pant, R. Thangaraj, and A. Abraham, “Particle swarm optimization using adaptive mutation,” in *Proceedings of the 19th International Conference on Database and Expert Systems Applications (DEXA ’08)*, pp. 519–523, Turin, Italy, September 2008.

[40] J. Vesterstrøm and R. Thomsen, “A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems,” in *Proceedings of the Congress on Evolutionary Computation (CEC ’04)*, pp. 1980–1987, June 2004.

[41] S. Y. Chen, Y. F. Li, and N. M. Kwok, “Active vision in robotic systems: a survey of recent developments,” *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1343–1377, 2011.

[42] S. Y. Chen, J. W. Zhang, H. X. Zhang, N. M. Kwok, and Y. F. Li, "Intelligent lighting control for vision-based robotic manipulation," *IEEE Transactions on Industrial Electronics*, vol. 59, pp. 3254–3263, 2012.

[43] S. Wen, W. Zheng, J. Zhu, X. Li, and S. Chen, "Elman fuzzy adaptive control for obstacle avoidance of mobile robots using hybrid force/position incorporation," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 42, no. 4, pp. 603–608, 2012.

[44] Y. J. Zheng, H. H. Shi, and S. Y. Chen, "Fuzzy combinatorial optimization with multiple ranking criteria: a staged tabu search framework," *Pacific Journal of Optimization*, vol. 8, pp. 457–472, 2012.

[45] K. Wang and Y. J. Zheng, "A new particle swarm optimization algorithm for fuzzy optimization of armored vehicle scheme design," *Applied Intelligence*, vol. 37, pp. 520–526, 2012.

Advances in Operations Research
Advances in Decision Sciences
Mathematical Problems in Engineering
Algebra
Journal of Probability and Statistics
The Scientific World Journal
International Journal of Differential Equations
International Journal of Combinatorics
Advances in Mathematical Physics
Journal of Complex Analysis
Journal of Mathematics
International Journal of Stochastic Analysis
Abstract and Applied Analysis
Discrete Dynamics in Nature and Society
International Journal of Mathematics and Mathematical Sciences
Journal of Discrete Mathematics
Journal of Function Spaces
Journal of Applied Mathematics
Journal of Optimization