Compound Particle Swarm Optimization in Dynamic Environments

Lili Liu[1], Dingwei Wang[1], and Shengxiang Yang[2]

[1] School of Information Science and Engineering, Northeastern University
Shenyang 110004, P. R. China
liulili1202@gmail.com, dwwang@mail.neu.edu.cn
[2] Department of Computer Science, University of Leicester
University Road, Leicester LE1 7RH, United Kingdom
s.yang@mcs.le.ac.uk

Abstract. Adaptation to dynamic optimization problems is currently receiving a growing interest as one of the most important applications of evolutionary algorithms. In this paper, a compound particle swarm optimization (CPSO) is proposed as a new variant of particle swarm optimization to enhance its performance in dynamic environments. Within CPSO, compound particles are constructed as a novel type of particles in the search space and their motions are integrated into the swarm. A special reflection scheme is introduced in order to explore the search space more comprehensively. Furthermore, some information preserving and anti-convergence strategies are also developed to improve the performance of CPSO in a new environment. An experimental study shows the efficiency of CPSO in dynamic environments.

1 Introduction

In recent years, there has been an increasing concern on investigating evolutionary algorithms (EAs) for dynamic optimization problems (DOPs) due to the relevance to real world applications, where many problems may involve stochastic changes over time. For DOPs, the goal of EAs is no longer to find a satisfactory solution, but to trace the moving optimum in the search space. This poses a great challenge to traditional EAs. To address this challenge, several approaches have been developed into EAs to improve their performance in dynamic environments, see [5, 12, 19, 20] for examples.

Recently, particle swarm optimization (PSO), as a class of EAs, has been applied to address DOPs with promising results [10, 15, 17]. In this paper, one behavior of particle swarms from the domain of physics is integrated into PSO and a compound particle swarm optimization (CPSO) is proposed to address dynamic environments. Within CPSO, a number of "compound particles" are constructed as a new type of particles that have a geometric structure similar to that described in [7]. But, instead of using geometric principles, a specialized reflection scheme is introduced in CPSO in order to explore the search space more comprehensively in a new environment. Furthermore, in order to improve the performance of CPSO in a new environment, some information preserving

and anti-convergence strategies are also developed to exploit various valuable information and avoid collision of particles respectively. An experimental study is carried out to validate the efficiency of CPSO in dynamic environments.

The rest of this paper is organized as follows. In the next section, we briefly review the usage of PSO for DOPs. Sec. 3 describes the CPSO proposed in this paper in details. The experimental results and analysis are given in Sec. 4. Finally, Sec. 5 concludes this paper with discussions on future work.

2 Particle Swarm Optimization in Dynamic Environments

Particle swarm optimization is a population based optimization technique with the inspiration from the social behavior of a swarm of birds (particles) that "fly" through a solution space [1, 13]. Each particle accomplishes its own updating based on its current velocity and position, the best position seen so far by itself and by the swarm. The behavior of a particle i, can be described as follows:

$$\boldsymbol{v_{ij}}(t) = \omega \boldsymbol{v_{ij}}(t-1) + c_1\xi(\boldsymbol{p_{ij}}(t) - \boldsymbol{x_{ij}}(t)) + c_2\eta(\boldsymbol{p_{gj}}(t) - \boldsymbol{x_{ij}}(t)) \quad (1)$$

$$\boldsymbol{x_{ij}}(t+1) = \boldsymbol{x_{ij}}(t) + \boldsymbol{v_{ij}}(t), \quad (2)$$

where $\boldsymbol{v_{ij}}(t)$ and $\boldsymbol{x_{ij}}(t)$ represent the current velocity and position of particle i in the j-th dimension at time t, and $\boldsymbol{p_{ij}}(t)$ and $\boldsymbol{p_{gj}}(t)$ represent the position of the best solution discovered so far by particle i and by all particles in the j-th dimension. The inertia weight ω controls the degree that a particle's previous velocity will be kept. Parameters c_1 and c_2 are individual and social learning factors, and ξ and η are random numbers in the range of $[0.0, 1.0]$.

PSO has been widely used for stationary problems [13, 14, 18]. In recent years, PSO has obtained an increasing concern to solve DOPs [3]. For DOPs, an efficient PSO must show continuous adaptation to track the variation of the optimal solution. For this aim, the basic PSO needs to be modified to improve the performance due to the following reasons. First, with the increasing of iterations, particles will congregate to a local or global optimum in the search space. When the optimum changes, the slackened velocities and convergent particles will result in a low exploration ability in the changing environment. Second, each particle takes into account the information from the best particles in the present swarm, while neglecting some valuable information contained in the inferior particles. This mono-directional mechanism restricts the ability of PSO to efficiently search for a new optimum.

Therefore, there are some key considerations to improve the adaptation of PSO in dynamic environments. They are shown as follows.

- Some weak particles should fly toward a better direction (i.e., the direction that intends to have increasing fitness) as quickly as possible, in order to adapt themselves to the changed environment and explore the search space comprehensively.

– Particles should also exploit the useful information from some other particles besides the best particle, in order to accelerate the optimization process in a new environment.

In recent years, several variations of the traditional PSO have been developed in the literature for promising performance in dynamic environments. Carlisle and Dozier [6] carried out a thorough investigation of PSOs on a large number of dynamic test problems and improved the performance of PSOs in both static and dynamic environments. An adaptive PSO that tracks various changes autonomously in a non-stationary environment was proposed in [8, 10]. Blackwell and Bentley [2] introduced a charged PSO for DOPs, which was then extended [4] by constructing interacting multi-swarms and using the charged sub-swarm to maintain population diversity for tracking multiple peaks in a dynamic environment. Parrott and Li [17] investigated a PSO model using a speciation scheme and employed this method to track multiple peaks simultaneously, the experiments manifested that the technique was able to track the changing trajectory.

3 Compound Particle Swarm Optimization

The concept of "compound particle" is derived from a branch of physics [9, 16]. It refers to a particular sort of particles that are composed by at least two particles through the chemical bond. Such particle possesses not only the qualities of each "member particle", but also some composite characters [16]. The characteristics of CPSO lie mainly in the following three aspects: 1) having the basic framework of canonical PSO; 2) incorporating the construction and update of compound particles; 3) employing a specialized reflection strategy and an integral-moving scheme for compound particles. In the following sections, the construction and operation for compound particles are described in details.

3.1 Initialization

Initially, a number of compound particles are created. Each compound particle is created as a simple geometrical structure that consists of three particles: one particle is selected from the initial swarm randomly and the other two are randomly generated to form a triangle with the length of the interconnecting edges being L. The three particles in a compound particle are denoted as "member particles". The particles in the swarm that do not belong to any compound particle are denoted as "independent particles".

3.2 Self-Adjustment

Each compound particle will adjust its internal structure in order to track the trace of the changing optimum. The essential steps involve constructing a new compound particle to explore good solutions in a new environment, and identifying a "representative particle" for each compound particle to participate in the canonical PSO. The procedures are exhibited as follows.

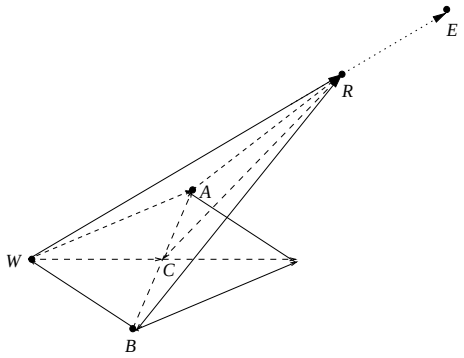

Fig. 1. Constructing a new compound particle.

Velocity-anisotropic reflection scheme: The construction of a new compound particle is illustrated in Fig. 1. The position of the worst particle in a compound particle is denoted as W and the position of the central point between the other two member particles is denoted as C. Then, the compound particle is reflected in accordance with the point W to a point R. The new compound particle consists of points A, R and B. If the solution at point R is better than that at point W, an expansion is further made in that direction to point E and the compound particle is updated to consist of points A, E, and B. The reflection point R and the extension point E are calculated as follows:

$$\boldsymbol{WR} = \boldsymbol{WC} + \boldsymbol{\gamma} \times \boldsymbol{WC} \tag{3}$$

$$\boldsymbol{WE} = \eta \times \boldsymbol{WR}, \quad \text{if } f(R) > f(W), \tag{4}$$

where $\boldsymbol{\gamma}$ is the inequality-velocity reflection vector and η is the extension factor.

Since the structure of a compound particle is a triangle in 2-dimensional space, in order to ensure that particles can explore the search space in N-dimension, a velocity-anisotropic reflection (abbreviated as VAR) scheme is introduced in the relevant vector.

Definition: An N-dimension vector $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_N)$ is a VAR vector, if it complies with:

$$0 < |\gamma_i - \gamma_j| \leq d, \quad i, j \in (1, 2, ..., N), \tag{5}$$

where d is the maximum difference between reflection velocities for each dimension, which determines the degree of departure from the initial direction $\boldsymbol{WC}$. The larger the value of d, the larger space of compound particles could explore.

It is clear that with the VAR vector shown in Eq. (5), $\boldsymbol{WR}$ can not be linearly represented by $\boldsymbol{WA}$ and $\boldsymbol{WB}$ in any case [11], that is, the VAR vector can drive compound particles to explore in the N-dimension search space. In this paper, each constituent in the VAR vector $\boldsymbol{\gamma}$ is generated as follows:

$$\gamma_{ij} = rand(0, e^{-|v_{ij}/v_{max}|}), \quad j \in (1, 2, \cdots, N), \tag{6}$$

where v_{ij} and γ_{ij} are the velocity and the reflection velocity of the i-th compound particle in the j-th dimension respectively.

In Eq. (6), the reflection velocity is designed to be relevant to the velocity of the member particle. We adopt such a rule for two reasons. First, when the velocities have a tendency to shrink up to a small value, especially when the population becomes convergent, the numerical range of the reflection velocity tends to be larger. Hence, the exploration ability will be enhanced adaptively because the degree of departure from the original direction is enlarged. second, the difference of each dimensional reflection velocity d will be restricted to a moderate degree in case the reflection direction deviates from the "better" direction significantly.

Identifying the representative particle: In order to maintain diversity as well as guarantee the searching precision in compound particles, two factors are integrated in identifying the representative particle: one is the fitness and the other is the total distance from one member particle to the other two particles. For each compound particle, its representative particle is identified according to the following probability:

$$P_{ci} = (1 - \beta)P_{ci}^{f} + \beta P_{ci}^{d}, \tag{7}$$

where P_{ci}^{f} and P_{ci}^{d} is the proportion of the fitness and the proportion of the total distance of the i-th member particle in the c-th compound particle respectively, β is the identification factor, and P_{ci} is the probability that the i-th member particle of the c-th compound particle becomes the representative particle.

3.3 Integral Movement

After the representative particles have updated their positions, an information preserving scheme is employed. In Blackwell and Branke's model [4], the updating of member particles all rely on their sub-swarm attractors and particle attractors. In this work, the velocity of a representative particle is conveyed to the other two member particles in the compound particle. That is, we first calculate the distance that a representative particle has moved and then move the other two member particles in the corresponding compound particle by the same distance. The reason for introducing this scheme lies in that the tendency of all member particles moving towards the local optimum is reduced and hence the convergence of the population is avoided and that, in the meantime, valuable information is preserved for the next iteration.

Fig. 2 is the pseudo-code of the CPSO we proposed.

4 Experimental Study

To test the validity of the proposed algorithm, Branke's moving peaks function [5] was used as a benchmark dynamic problem. The fitness at a point of the

begin
  Parameterize
  $t := 0$
  Initialize population (swarm of particles) $P(0)$ randomly
  Initialize compound particles $C(0)$ based on the value of L
  repeat
    for each compound particle **do**
      Perform self-adjustment according to Eqs. (3) and (4)
      Identify the representative particle according to Eq. (7)
    end for
    Create population $P(t)$ containing independent and representative particles
    for each particle i in $P(t)$ **do**
      Update $\boldsymbol{v_i}$ and $\boldsymbol{x_i}$ by Eqs. (1) and (2)
      if $f(\boldsymbol{x_i}) < f(\boldsymbol{p_i})$ **then** $\boldsymbol{p_i} := \boldsymbol{x_i}$
    end for
    $\boldsymbol{p_g} := \arg\min_{\boldsymbol{p_i}}\{f(\boldsymbol{p_i})|i = 1, \cdots, swarm\_size\}$
    for each compound particle **do**
      Calculate the distance the representative particle has moved after update
      Move the other two member particles by the same distance
    end for
    $t := t + 1$
  until the termination condition is met
end

Fig. 2. Pseudo-code of the Compound Particle Swarm Optimization (CPSO)

fitness landscape is assigned the maximum height of all optima at that point as below.

$$F(\boldsymbol{x}, t) = \max_{i=1,\ldots,M} \frac{H_i(t)}{1 + W_i(t) \sum_{j=1}^{N} (x_j(t) - X_{ij}(t))^2}, \tag{8}$$

In the experiments, we set $N = 5$, $M = 10$, and $\boldsymbol{x} \in [X_{max}, X_{min}]^5 = [0, 100]^5$. The height and width of each peak were randomly generated with a uniform distribution in $[30, 70]$ and $[1, 12]$ respectively. The locations of peaks are changed as follows:

$$\boldsymbol{v}_i(t) = \frac{s}{|\boldsymbol{r} + \boldsymbol{v}_i(t-1)|}((1-\lambda)\boldsymbol{r} + \lambda \boldsymbol{v}_i(t-1)) \tag{9}$$

$$\boldsymbol{X_i}(t) = \boldsymbol{X_i}(t-1) + \boldsymbol{v}_i(t), \tag{10}$$

where the vector $\boldsymbol{v}_i(t)$ is a linear combination of a random vector $\boldsymbol{r} \in [0.0, 1.0]^N$ and the previous vector $\boldsymbol{v}_i(t-1)$ and is normalized by the length factor s (s controls the severity of changes), and λ is the correlation parameter. In our experiment, λ is set 0, which indicates that the movement of peaks is uncorrelated.

Table 1. Dynamic environments.

τ	Scenario		
10	1	2	3
50	4	5	6
100	7	8	9
$s \rightarrow$	0.05	0.5	1.0

Table 2. The t-test results of comparing algorithms on dynamic problems.

t-test results	Scenario								
	1	2	3	4	5	6	7	8	9
SPSO − PSO	+	+	∼	+	+	+	+	+	+
R-CPSO − PSO	+	+	∼	+	+	+	+	+	+
R-CPSO − SPSO	∼	∼	∼	−	+	+	∼	−	−
CPSO − PSO	+	+	+	+	+	+	+	+	+
CPSO − SPSO	+	+	+	+	+	+	∼	∼	+
CPSO − R-CPSO	+	+	+	+	+	+	∼	+	+

The performance of CPSO is compared with the simple PSO model (PSO) and the speciation based PSO (SPSO) proposed by Parrott and Li [17]. In order to test the effect of the VAR scheme, a corresponding algorithm called R-CPSO with $\gamma_{ij} = rand(0, 1)$ is involved in the experiments, which implies a random exploration within a constrained space. For all PSO models, the learning factors $c_1 = c_2 = 2.0$, the inertia weight $\omega = \omega_{max} - (\omega_{max} - \omega_{min}) * iter/iter_{max}$ ($\omega_{max} = 0.7$, $\omega_{min} = 0.5$, $iter_{max}$ and $iter$ are the max number of iterations and the current iteration respectively). Parameters in SPSO are set as [17]: $P_{max} = 20$ and $r = 20$. Parameters in CPSO are set as follows: the length of edges $L = 0.01 \times (X_{max} - X_{min}) = 1$, the extension factor $\eta = 1.25$, the identification factor $\beta = 0.5$, implying that the ingredients of fitness and distance in Eq. (7) have an equal strength. The total number of particles is set to 50 for PSO and SPSO and is set to 20 for CPSO and R-CPSO, where half of particles in the initial swarm are selected to construct compound particles, to ensure the fairness of comparisons between algorithms.

For environmental dynamics parameters, we set $s \in \{0.05, 0.5, 1.0\}$ and $\tau \in \{10, 50, 100\}$, where τ determines the speed of change (i.e., the environment changes every τ generations). This gives 9 different scenarios, i.e., 9 pairs of (s, τ). For each scenario, 20 random instances were created and the results were averaged over the 20 runs. For each run the error concerned with the best-of-period fitness was record every iteration [15]. The mean error of an algorithm is calculated as follows:

$$E_{mean} = \frac{1}{G} \sum_{i=1}^{G} (\frac{1}{N} \sum_{j=1}^{N} e_{ij}), \tag{11}$$

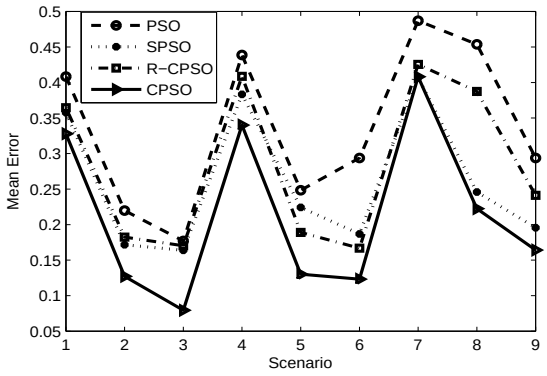

Fig. 3. The mean error of PSOs on different dynamic problems.

where $G = 10$ is the total number of changes for a run, this way, the number of iterations $Iter_{max} = 10\tau$, $N = 20$ is the total number of runs, e_{ij} is the error of the last iteration at the i-th change in the j-th run.

The experimental results are plotted in Fig. 3 for different dynamic problems, which are indexed by the combination of the pairs (s, τ), as shown in Table 1. The statistical test results of comparing algorithms by one-tailed t-test with 38 degrees of freedom at a 0.05 level of significance are given in Table 2, and the t-test result regarding $Alg.\ 1 - Alg.\ 2$ is shown as "+" or "$\sim$" when $Alg.\ 1$ is significantly better than or statistically equivalent to $Alg.\ 2$ respectively.

From Fig. 3 and Table 2, it can be seen that both SPSO and CPSO significantly outperform PSO on the dynamic problems with different environmental dynamics, CPSO outperforms SPSO on most dynamic problems, and CPSO performs significantly better than R-CPSO. This result validates our expectation of CPSO. The better performance of CPSO is because compound particles in CPSO integrate valuable information effectively, and comparing with the random exploration scheme in a guideless way within R-PSO, the VAR scheme has an intensive exploration ability and helps the compound particles to search for more optimal solutions continuously rather than converging into a solution ahead. Furthermore, the information preserving scheme in the process of Integral Movement can improve the exploitation ability of CPSO.

In order to further investigate the performance of CPSO, the dynamic performance of algorithms PSO, SPSO, and CPSO regarding the offline performance defined in [5] with $s = 1.0$ and different speed of changes is plotted in Fig. 4. From Fig. 4, some conclusions similar to the previous ones can be drawn, and, furthermore, CPSO is only beaten by SPSO occasionally. This is because when $s = 1.0$, the problem endures severe changes and hence some previous information may not be valid. However, when the environment changes slowly, e.g., when $\tau = 100$, CPSO performs better due to the VAR scheme.

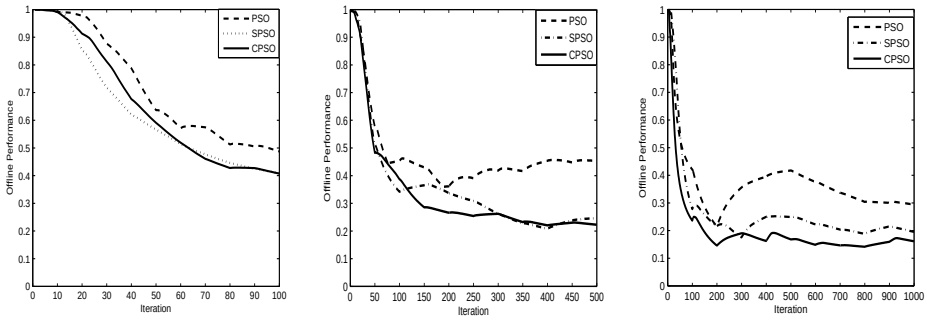

Fig. 4. Dynamic performance of PSOs on dynamic problems with $s = 1.0$: (a) $\tau = 10$, (b) $\tau = 50$, and (c) $\tau = 100$

5 Conclusions

This paper introduces a new kind of particles, called "compound particles", together with some specialized operating mechanisms into PSO for DOPs. The compound particles introduced can aggregate more valuable information with a simple configuration, which is an improvement over traditional PSO. A velocity-anisotropic reflection method is proposed to construct new compound particles, which can drive particles to search for a better solution especially in a new environment. Furthermore, the information preserving and anti-convergence strategies are also applied to the motion of compound particles. Experimental study over a benchmark dynamic problem shows that the proposed schemes efficiently improve the performance of PSO in dynamic environments.

For future work, it would be valuable to investigate the effect of different modifications to CPSO for DOPs. The VAR mechanism in CPSO are quite general and hence can be integrated to other optimization methods to improve their capability in dynamic environments. This is another interesting research for the future work.

Acknowledgments

The work by Lili Liu and Dingwei Wang was supported by the Key Program of the National Natural Science Foundation (NNSF) of China under Grant No. 70431003 and Grant No. 70671020, the Science Fund for Creative Research Group of NNSF of China under Grant No. 60521003 and the National Science and Technology Support Plan of China under Grant No. 2006BAH02A09. The work by Shengxiang Yang was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant No. EP/E060722/1.

References

1. B. Brandstätter and U. Baumgartner. Particle swarm optimization: Mass spring system analogon. *IEEE Trans. on Magnetics*, **38**(2): 997–1000, 2002.
2. T. M. Blackwell and P. J. Bentley. Dynamic search with charged swarms. *Proc. of the 2002 Genetic and Evol. Comput. Conf.*, pp. 19–26, 2002.
3. T. M. Blackwell. Swarms in dynamic environments. *Proc. of the 2003 Genetic and Evol. Comput. Conf.*, LNCS vol. 2723, pp. 1–12, 2003.
4. T. M. Blackwell and J. Branke. Multi-swarm optimization in dynamic environments. In G. R. Raidl et al. (eds), *Applications of Evolutionary Computing*, LNCS vol. 3005, pp. 489–500, 2004.
5. J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, 2002.
6. A. Carlisle and G. Dozier. Adapting particle swarm optimization to dynamic environments. *Proc. of the 2000 Int. Conf. on Artif. Intell.*, pp. 429-434, 2000.
7. C. D. Chio, A. Moraglio, and R. Poli. Geometric particle swarm optimisation on binary and real spaces: from theory to practice. *Proc. of the 2007 GECCO conference companion on Genetic and evolutionary computation*, pp. 2659-2666, 2007.
8. R. C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. *Proc. of the 2001 IEEE Congress on Evol. Comput.*, pp. 94–100, 2001.
9. F. S. Galasso. *Structure and Properties of Inorganic Solids*, Pergamon Press, 1970.
10. X. Hu and R. C. Eberhart. Adaptive particle swarm optimization: Detection and response to dynamic systems. *Proc. of the 2002 IEEE Congress on Evol. Comput.*, pp. 1666–1670, 2002.
11. S. K. Jain. *Linear Algebra*. Thomson, 4th edition,pp. 12-20, 1994.
12. Y. Jin and J. Branke. Evolutionary optimization in uncertain environments: a survey. *IEEE Trans. on Evol. Comput.*, **9**(3): 303–317, June 2005.
13. J. Kennedy and R. Eberhart. Particle Swarm Optimization. *Proc. of the 1995 IEEE Int. Conf. on Neural Networks*, vol. 4, pp. 1942-1948, 1995.
14. J. Kennedy. Stereotyping: Improving particle swarm performance with cluster analysis. *Proc. of the 2000 IEEE Congress on Evol. Comput.*, pp. 1507–1512, 2000.
15. X. Li and K. H. Dam. Comparing particle swarms for tracking extrema in dynamic environments. *Proc. of the 2003 IEEE Congress on Evol. Comput.*, pp. 1772–1779, 2003.
16. S. Nakahata, K. Sogabe, T. Matsuura, and A. Yamakawa. One role of titanium compound particles in aluminium nitride sintered body. *Journal of Materials Science*, **32**(7), pp. 1873-1876, 1997.
17. D. Parrott and X. Li. A particle swarm model for tacking multiple peaks in a dynamic environment using speciation. *Proc. of the 2004 Genetic and Evol. Comput. Conf.*, pp. 98–103, 2004.
18. K. E. Parsopoulos and M. N. Vrahatis. On the computation of all global minimizers through particle swarm optimization. *IEEE Trans. on Evol.Comput.*, **8**(3): 211–224, June 2004.
19. S. Yang. Population-based incremental learning with memory scheme for changing environments. *Proc. of the 2005 Genetic and Evol. Comput. Conference*, vol. 1, pp. 711–718, 2005.
20. S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Computing*, **9**(11): 815-834, 2005.