Particle Swarm Optimizer with Aging Operator for Multimodal Function Optimization

BoJiang [1], Ning Wang [1]*, Xiaodong Li [2],

[1] *National Laboratory of Industrial Control, Zhejiang University, No.38 Zheda Road, Hangzhou,Zhejiang,310027, China*
E-mail: bjiang,nwang@iipc.zju.edu.cn
[2] *School of Computer Science and Information Technology, RMIT University, GPO Box 2476, Melbourne,3001, Victoria, Australia.*
E-mail: xiaodong.li@rmit.edu.au

Received 22 May 2012

Accepted 9 January 2013

Abstract

This paper proposes a new scheme for preventing a Particle Swarm Optimizer from premature convergence on multimodal optimization problems. Instead of only using fitness evaluation, we use a new index called particle age to guide population towards more promising region of the search space. The particle age is a measure of how long each particle moves towards a better solution. The main novelty of the proposed method is to let each particle learn from not only neighbours with better fitness values but also the neighbours whose fitness values are updated more frequently. To achieve this, we design a comprehensive age-based learning strategy, in which age is used for excluding old particles, selecting learning exemplars and deciding mutation strength and inertial weight for each particle. Experiments were conducted on 15 multimodal test functions to assess the performance of this new strategy in comparison with 7 state-of-the-art PSOs from the literature. The experimental results show the good performance of the proposed algorithm in solving multimodal functions when compared with several existing PSO variants.

Keywords: soft computing; function optimization; particle swarm optimizer (PSO); swarm intelligence.

1. Introduction

Particle Swarm Optimizer (PSO) has been shown to be very successful in solving complex and challenging optimization problems [1,2]. However, a common problem often experienced when applying PSO to multimodal function optimization is that the particle population loses diversity too rapidly before it converges to some reasonable solutions [3,4,5,6,7], which is commonly referred as premature convergence.

Various schemes have been proposed to enhance PSO to cope with premature convergence. These methods can be divided into two categories, one employing population diversity maintenance and another adopting some mechanism to escape local optima. More specifically, methods for diversity maintanance include those using parameter adjustment [8,9,10], mutation [3,11,12], improved topology structure [5,7,13,14] and multi-population [4,13,15,16]. Although these works keep a reasonable balance between diversity and convergence, it is still far from optimal. When this occurs, sometimes the only choice is to

*Corresponding author.

restart [17]. Restarting strategy does help to improve the performance of PSO [18], however one problem is to decide when the entire population needs restarting.

An alternative strategy to restart the entire swarm is to restart part of the population regularly in conjunction with the use of an aging operator [19]. Basically the aging operator works by accumulating its ages, an individual with its age exceeding the maximal age τ is removed from the current population and a new randomly generated individual is inserted. τ is predefined by an user to determine the lifespan of each individual. The aging operator maintains population diversity through consecutive individual replacement during the search process. Compared to restarting the entire population, one advantage of aging operator is that new individuals coexist with old population and can learn useful information from them. It was previously demonstrated that aging operator can achieve performance improvements that restarts cannot[20]. In the past decade, the aging operator has been employed in many evolutionary algorithms (EAs) to control ways of how a population is generated [21,22,23]. Two typical aging operators are evolutionary aging[24] and static pure aging[21,22]. Jansen's work[25] shows the static pure aging can help escape from local optima and evolutionary aging is more effective on optimize functions with plateau .

The aim of this paper is to propose a new age-based search strategy to improve the performance of PSO for solving multimodal problems. We propose four core age-based operators, which are particle replacement, neighbour selection, hypermutation and inertial weight adjustment. This new algorithm differs from the existing age-based EAs and PSOs in the following aspects.

(1) We propose a new and PSO-oriented age definition that can be used for detecting both fitness stagnation and oscillation.

(2) The age index is used not only for excluding particles but also for constructing a comprehensive age-based learning strategy.

(3) Instead of the *gbest* population topology, an age-based population topology is used to select neighbours for each particles, to improve the performance of PSO on multimodal problems in particular.

(4) A new age-based hypermutation is developed to decide the mutation strength for selected particles according to their age.

(5) A new age-based adaptive scheme is used to dynamically determine the inertial weight for each particle. In this scheme, particles with different ages have different inertial weights.

The rest of this paper is organized as follows. The related work about PSO and aging operators are reviewed in Section 2. In Section 3, we investigate two important phenomena of PSO to explain why the proposed definition of particle age is useful. Then, we present the framework of the proposed age-based learning strategy in Section 4. Experimental results are shown in Section 5. The discussions and conclusion are presented in Section 6.

2. Related Works

2.1. PSO

PSO was first proposed by Kennedy and Eberhart in 1995 [26]. Each particle i is composed of two vectors that are position vector $\mathbf{x}_i^t = \left[x_{i,1}^t, x_{i,2}^t, ..., x_{i,D}^t\right]$ and velocity vector $\mathbf{v}_i^t = \left[v_{i,1}^t, v_{i,2}^t, ..., v_{i,D}^t\right]$, where D denotes the dimension of search space and t is the generation. Each particle's personal best position (*pbest*) $\mathbf{p}_i^t = \left[p_{i,1}^t, p_{i,2}^t, ..., p_{i,D}^t\right]$ and neighbor's best position (*nbest*) $\mathbf{p}_n^t = \left[p_{n,1}^t, p_{n,2}^t, ..., p_{n,D}^t\right]$ are used for communicating information about the good positions in the search space to each particle. $\mathbf{x}_i^t$ and $\mathbf{v}_i^t$ are updated as follows[9]:

$$\mathbf{v}_i^{t+1} = \omega \mathbf{v}_i^t + c_1 r_1 \left(\mathbf{p}_i^t - \mathbf{x}_i^t\right) + c_2 r_2 \left(\mathbf{p}_n^t - \mathbf{x}_i^t\right) \quad (1)$$

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1} \quad (2)$$

where ω is called inertia weight and c_1 and c_2 are acceleration constants. r_1 and r_2 are two random numbers in the range $[0,1]$. The ω dampens the influence of previous velocity to the current velocity and can be interpreted as the fluidity of the medium where each particle moves[27]. c_1 and c_2 control the weight of stochastic acceleration terms that pull each particle toward *pbest* and *nbest*, respectively[5]. To better control the velocity, a positive value V_{max} is used to clamp each particle's velocity on each dimension within $[-V_{max}, V_{max}]$. According to population topology structure, there are two kinds of PSO, global best version PSO (*gbest*PSO) and local best version PSO (*lbest*PSO). In *gbest*PSO, each particle chooses the global best position of the current population as its *nbest*. In *lbest*PSO, each particle selects the best position in its neighbourhood defined by local population topology as *nbest*.

2.2. Diversity maintenance methods in PSO

Many works have been reported on the issue of maintaining population diversity of PSO and some of the representative works are reviewed here. PSO's search behaviour is highly influenced by two parameters, i.e. inertial weight and acceleration constants, so the easiest method used to keep effective diversity during search may be adjusting the two parameters dynamically. For example, a time-decreasing inertia weight[9], different inertial weight for each particle [28], time-varying acceleration constants [8,10]. Another important parameter is population size. For example, adjusting population size according to the status of the global best position [29]. In addition, incremental social learning is also introduced to construct a growing population size [6]. Besides these learning parameter, population topology also plays an important role in maintaining diversity. Many static *lbest* topology have been proposed to improve PSO's performance on multimodal functions, such as ring topology[16], von Neuman topology [30] and fully informed structure[31]. Eberhart and Kennedy [32] has shown that a static *lbest* ring topology converges more slowly than *gbest* topology but performs better on multimodal functions. Recently, some dynamic topologies have been developed, for example random topology [13], comprehensive learning [5] and orthogonal learning [7].

Some effective diversity maintaining methods used in other EAs are also introduced to PSO. Mutation is one of them. Several mutation strategies have been proposed, such as Gaussian mutation, Cauchy mutation and hypermutation [12]. Besides mutation, multi-population methods have also been introduced to PSO for multimodal function optimization. The first category of multi-population method in PSO evolves multiple subpopulations in parallel and each subpopulation searches a different region of the landscape. This method is often used to search the landscape of a multimodal function or deal with dynamic optimization problems. Cluster-based PSO[33,34] and niching PSO [16,35] are two representative multipopulation based methods. Another multipopulation based method is cooperative coevolution PSO, which splits solution vectors into multiple smaller vectors and each of these smaller search spaces is searched by a separated population. Bergh and Engelbrecht [4] introduced cooperative coevolution into PSO, then Li and Yao [36] improved it to optimize large-scale continuous problems.

2.3. Aging operators

If premature convergence happened, the only way is to restart the entire population to search once again till the terminal condition is achieved. Instead of restarting all individuals, aging operator selects part of individuals to restart every time according to the age of an individual. In a way, aging operator can be seen as a steady-state version of the restart method.

Most of the current aging operators are used to remove particles with poor performance. Two frequently used aging operators are evolutionary aging [23,37,24,38] and static pure aging [22,21,39,40]. The evolutionary aging is often used in EAs and the static pure aging is used in artificial immune systems(AISs). What is in common between these is that each individual with its age exceeding maximal age τ is replaced by a new randomly generated individual. The main difference between the two aging operators lies in the definition of an individual's age. In the evolutionary aging, each new offspring generated through crossover or mutation is assigned age 0 and the age of each remaining individual is increased by one in each generation [23,24]. The parent individuals are included in the remaining individuals if the parents can exist with their offspring [24]. In the static pure aging, each new offspring is assigned age 0 only if its fitness is better than its parents fitness, otherwise it inherits its parents age [21]. It is obvious that the individual age defined in evolutionary aging does not depend on individual fitness but measures how long the individual survives. In contrast, the age in static pure aging represents how long an individual has not reproduced a better offspring. For a minimization problem, the definition of age in the two aging operators are as follows [25]:

Definition 1 (evolutionary age) $\boldsymbol{y}.age = 0$.

Definition 2 (static pure age) *if* $f(\boldsymbol{y}) < f(\boldsymbol{x})$ *then* $\boldsymbol{y}.age = 0$ *else* $\boldsymbol{y}.age = \boldsymbol{x}.age$.

where **x** and **y** represent parent individual and child individual, respectively. Jasen's recent work [25] compares the above two aging operators and shows that: the static pure aging can recognize local optima while the evolutionary aging fails. On the other hand the evolutionary aging is able to optimize functions with plateau but the static pure aging can't. To improve the performance of static pure aging on plateau, a new aging called genotypic aging was also proposed [25]. The age in genotypic aging is defined as follows:

Definition 3 (genotypic age) *if* $f(\boldsymbol{y}) \leqslant f(\boldsymbol{x}) \wedge \boldsymbol{y} \neq \boldsymbol{x}$ *then* $\boldsymbol{y}.age = 0$ *else* $\boldsymbol{y}.age = \boldsymbol{x}.age$

Different from static pure aging, offspring with the same fitness but in different place with respect to its parent, is assigned age 0 in genotypic aging. It is obvious that this mechanism allows for the random walk on plateau. Besides the above three agings, Hornby [41] proposed a new aging, called the age-layered population structure (ALPS). In ALPS, randomly created individuals start with age 0 and the age is then increased by 1 in each generation if an individual is used for producing an offspring. Individuals created through mutation and crossover start with age 1 plus the maximal age of their parents.

3. Proposed Definition of Particle Age

3.1. *Fitness stagnation and oscillation*

Before defining new age for PSO, let us first consider the drawback of the original PSO, which will be used to explain why the existing age definitions are unsuitable for PSO. To do this, consider the original PSO with *gbest* topology for multimodal funtion minimization. In Eq.(1) and (2), we eliminate the velocity term and transform them to the format as follows:

$$\mathbf{x}_i^{t+1} + (\phi_1 + \phi_2 - \omega - 1)\mathbf{x}_i^t + \omega\mathbf{x}_i^{t-1} = \phi_1\mathbf{p}_i^t + \phi_2\mathbf{p}_g^t \tag{3}$$

where $\phi_1 = c_1 r_1$ and $\phi_2 = c_2 r_2$. A particular solution of this second-order difference equation Eq. (3) is:

$$\mathbf{o}_i^t = \frac{\phi_1\mathbf{p}_i^t + \phi_2\mathbf{p}_g^t}{\phi_1 + \phi_2} \tag{4}$$

Furthermore, existing work has proved the following equation under certain conditions[42]:

$$\lim_{t\to\infty}\mathbf{x}_i^t = E(\mathbf{o}_i^t) = \frac{c_1\mathbf{p}_i^t + c_2\mathbf{p}_g^t}{c_1 + c_2} \tag{5}$$

This means that, particle i will converge to or oscillates around an equilibrium point $E(\mathbf{o}_i^t)$, which is a weighted average of $\mathbf{p}_i^t$ and $\mathbf{p}_g^t$ if t is big enough. In the initial stage, due to the $|\boldsymbol{gbest} - \mathbf{x}|$ is significantly larger than $|\boldsymbol{pbest} - \mathbf{x}|$, each particle is attracted toward *gbest* position because its big influence. As search continues, the gap between $|\boldsymbol{gbest} - \mathbf{x}|$ and $|\boldsymbol{pbest} - \mathbf{x}|$ is reduced gradually, the velocity of each particle becomes small. Under this circumstance, if *gbest* and *pbest* are on the same valley of fitness landscape, the particle is still attracted to move toward *gbest* direction. However, it brings a problem that is if the *gbest* is a local optima, the particle may be impossible to jump out of this local optima once its *pbest* has moved into the same area with the *gbest* [5]. We call this fitness stagnation, because all individuals are trapped in the same valley or peak and cannot jump out. Another situation is that if the fitness of *gbest* and *pbest* are very close and they are in two different valleys of the function landscape, they would make the particle oscillate [16]between them. Oscillation is also regarded as a kind of premature as *gbest* and *pbest* are not changed during oscillation. This is called fitness oscillation.

When the fitness stagnation takes place, both the particle's fitness value and its *pbest* fail to further improve. However, when the fitness oscillation happens, particle's fitness value is fluctuated but its *pbest* dose not change. So all the above three existing aging operators fail to detect fitness oscillation because no *pbest* information is used in them.

3.2. Particle age

To recognize the fitness oscillation of PSO, we propose a new definition of age for particles in PSO. The new age is defined as follows:

Definition 4 (particle age) *if* $f(\boldsymbol{x}_i^t) < f(\boldsymbol{p}_i^t)$ *then* $\boldsymbol{x}_i^{t+1}.age = 0$ *else* $\boldsymbol{x}_i^{t+1}.age = \boldsymbol{x}_i^t.age + 1$

In *particle age*, each new particle begins from age 0 and the change of its age depends on whether it can find a better *pbest* after one iteration. If a particle find a better *pbest* to replace the current one, its age becomes 0. On the other hand, if it fails to update its *pbest*, its age is increased by 1 in this iteration.

A distinct feature of *particle age* is that it utilizes particle's best historical information and particle's current information to determine the age for each particle. It can recognize the two kinds of premature convergence, because no matter which of them happens it is impossible to change a particle's pbest. Thus it causes a linear increase on age of the particle. When the sharp increase is detected, some remedial measures, such as adjusting parameters and restarting, can be utilised to allow the particle to escape from the current local optima. An example of the change of particle age during optimization is shown in Fig.1. The objective particle was randomly selected from 40 particles used in a basic PSO for optimizing 30-D Rastrigin function [5]. From Fig.1, this particle was relatively easy to update its *pbest* at the earlier stage of search but difficult to find a better *pbest* after about 1500 generations. This may be caused by the lose of population diversity after certain number of generations. Furthermore, this tendency appeared on other multimodal functions such as Rosenbrock, Griewank, Ackley, Weierstrass and Schwefel function[5] when the original PSO was used.

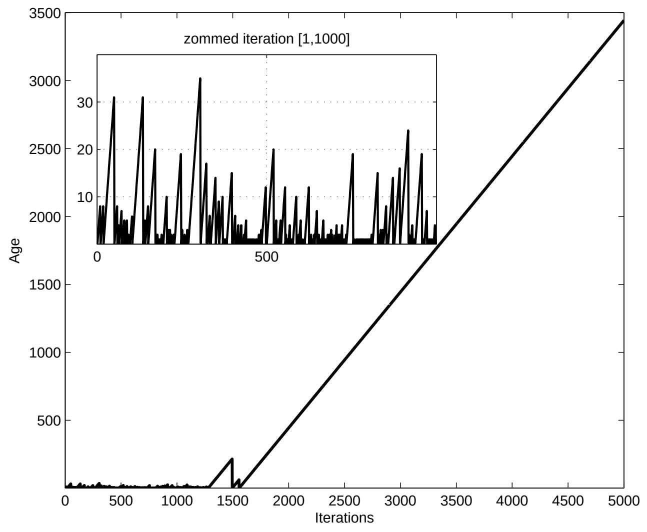

Fig. 1. Example of the change in age of a particle when using basic PSO to optimize 30-D Rastrigin function.

4. Particle Swarm Optimizer with Age

4.1. Framework of PSOA

As defined above, the age of each particle shows whether the space around it is a promising region for itself or other particles, in order to search for better solutions. A too old particle means that the direction it lies is hopeless and we do not hope any particles search toward it. In other words, it is possible for population to find a better solution if more particles are used to search around a particle whose age is zero. According to this basic guideline, we propose the following three learning principles: First, any particle whose age exceeding the permitted maximal age is replaced by a new random particle. Second, each particle learns information from the younger particles. Third, to avoid being replaced, a high probability should be assigned to the older particle so that it can update its *pbest*.

PSOA starts from a randomly generated initial swarm with a random topology. The framework of PSOA is given in Algorithm 1. Four main components of PSOA, i.e. *particles replacement*, *neighbours selection*, *hypermutation* and *inertial weight adjustment*, which are the four age-based operators are described in the following sections.

4.2. Particle replacement

The original aging operator that excludes particles with their age exceeding τ is used in PSOA to manage population dynamically. We call it particle replacement in this paper. At the end of each generation, the age of each particle is computed according to the *particle age* defined in **Definition 4**. Specifically, if particle i find a new position that is strictly better than its current *pbest* its age α_i is changed to

Algorithm 1 The framework of the proposed PSOA

```
Initialization:
 1: Randomly generate an initial swarm $S$ with size $P$;
 2: Set the fitness evaluations counter $fevals = 0$;
 3: Set the age of initial swarm $\alpha_i = 0 (i = 1, 2, ..., P)$;
Iterations:
 4: while stop condition is not satisfied do
 5:     if regular interval achieves then
 6:         $ParticlesReplacement(S, \alpha, \tau, fevals)$;
 7:         $NeighbourSelection(S, \alpha)$;
 8:     end if
 9:     for $i = 1 : P$ do
10:         $InertiaWeightAdjustment(S, \omega, \alpha)$;
11:         Update velocity and position according (1) and (2) ;
12:     end for
13:     if mutation condition is satisfied then
14:         $Hypermutation(S, a)$;
15:     end if
16:     $FitnessEvaluation(S, fevals)$;
17:     Update age according the Definition 4
18: end while
```

0 ($\alpha_i = 0$), else α_i is increased by one ($\alpha_i = \alpha_i + 1$) . When the latter happens, α_i will be compared with τ. If $\alpha_i > \tau$, the particle i is excluded from the current population. Then, a new particle with age 0 is randomly generated in the search space and inserted into population. The above process is executed on each particle in the population.

4.3. Neighbour selection

Population topology structure has significant influence on the performance of PSO for multimodal function optimization. Instead of using *gbest* topology, an age based population topology with K neighbours is proposed in this paper. In this age topology, each particle first randomly selects K particles in population as its neighbours then chooses the particle with the best fitness among the K neighbours as its *nbest*. At regular intervals, this procedure is repeated to select new neighbours for each particle. One important question is how to select the K neighbours from the population. In existing work [13], the K neighbours for each particle is randomly selected from the entire population. In PSOA, each particle selects ones that not older than itself as its neighbours. One benefit of this strategy is improving the chance to share information with good particles and reduce the probability to search on poor directions. Specifically, we select neighbours for particle i as follows.

(1) We first find out all particles except i with their age not exceeding the age of particle i (α_i). The number of particles is N.

(2) If $N \geqslant K$, randomly choose K particles from the N individuals as the neighbours of particle i. If $N < K$, choose the global best particle of current population as its only neighbour.

(3) If $N \geqslant K$, compare the fitness value of these K particles' *pbest* and select the best one as the *nbest* of particle i.

This procedure is executed at every *age-gap* generations. For example, if the value of *age-gap* is 15, the topology of each particle is reconstructed at generation 15, 30, 45, ... To leave enough time for

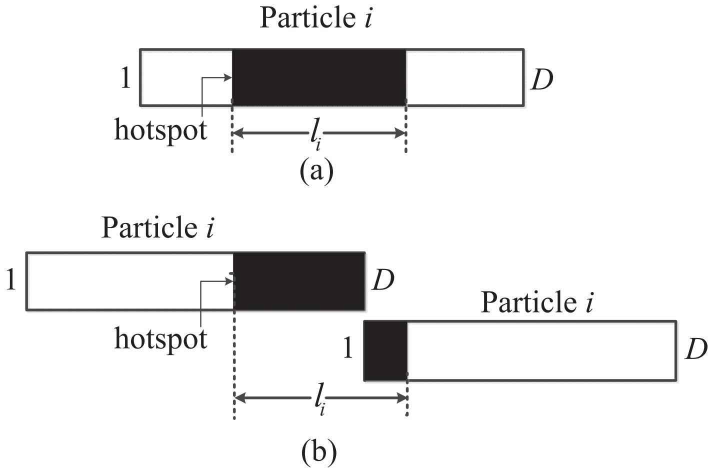

Figure 2: Determine mutation region in different situation. (a) the length after the hotspot is longer than l_i. (b)the length after the hotspot is shorter than l_i

each particle to learn more information from good neighbours, the *age-gap* is set to the maximal age τ in this paper. The main difference between the proposed topology and other random topology [13] is that it uses age to decide the constitution of neighbourhoods but not randomly selects them from the entire population. Since age represents the ability of a particle to update its *pbest*, searching around young particles may provide more promising solutions.

4.4. Hypermutation

Hypermutation is a basic mechanism of cell immune response and has been simulated in artificial immune systems (AISs) for machine learning and optimization[43]. Its distinct feature is that it makes all dimensions in a contiguous region of vector to be mutated. Our preliminary work [12] implemented this idea in PSO and found its good performance on multimodal problems. Instead of using a random mutation strength as in [12], we use the particle age to determine the range of the contiguous region. We hope the older particles have the higher mutation strength to push them to a new position, since they have stagnated or oscillated for a long time.

For particle i with age α_i, if it is selected to be mutated, its mutation length l_i is as follow.

$$l_i = \begin{cases} \lceil 1 + (\frac{D}{2} - 2) * \frac{\alpha_i}{\tau + 1} \rceil & \alpha_i \leqslant \tau \\ \frac{D}{2} & \alpha_i > \tau \end{cases} \quad (6)$$

In (6), D is the particle dimensions and $\lceil\rceil$ is ceiling function. If $\alpha_i = 0$, only one dimension of particle i is chosen to be mutated and this is same with one point mutation. With the increase of α_i, the mutation length l_i is also increased from one to $\frac{D}{2}$. If $\alpha_i > \tau$, the mutation length is $D/2$, which is the highest mutation strength allowed in PSOA. In this paper, the mutated particles are not considered as new particles because at least half of these particles' dimensions information are retained. As a result, the age of them remain unchanged. For particle i, a hotspot (mutation point) is first randomly selected within the dimension of it to implement the hypermutation. If the distance between hotspot and the end of the vector is longer than l_i, then all l_i dimensions that from the hotspot onward are mutated (Fig.2(a)). If the distance between them is shorter than l_i, the remainder dimensions are used from the beginning of vector i (Fig.2(b)).

4.5. *Inertia weight adjustment*

Although particle replacement can discard aged particles and introduce new particles to replace them, it may lose useful information in the discarded particles. Furthermore, adding new particles needs additional fitness evaluations. In order to make the best use of the existing particles, it may be better to keep them through finding better *pbest*. So, it is reasonable to equip the old particles with better local search capability. To do this, we build a connection between each particle's age and inertia weight to decrease ω when it gets old. The inertia weight for particle i (ω_i) is changed as follows.

$$\omega_i = \begin{cases} 0.729(1-\dfrac{\alpha_i}{\tau+1}) & \alpha_i \leqslant \tau \\ 0 & \alpha_i > \tau \end{cases} \qquad (7)$$

In PSOA, ω is initialized to 0.729 which can keep the search balanced between global best and personal best position[27]. From (7), the ω_i of particles with age 0 are assigned 0.729. The ω_i is decreased by $1/(\tau+1)$ when the age of particle i (α_i) is increased by one till the α_i is larger than τ. Due to the particle replacement is performed at every age-gap generations, some particles with their age exceeding τ may be kept in population. For these particles, we assign them age 0 to give them the highest local search abilities to update their *pbest* and they will be excluded in the next iteration if this is failed.

4.6. *Setting the maximal age τ*

To select an appropriate τ, six 30-D multimodal functions are used to investigate the influence of τ. The six function are Rosenbrock, Griewank, Ackley, Rastrigin, Weierstrass and Schwefel function[5]. First, we use the basic *gbest*PSO to run it 30 times on all functions. Table 1 shows the average and standard deviation of the age of the entire population during optimization process. The data before the termination of the PSOA is used for statistical analysis. It is obvious that the population ages in the six functions are very close and their mean value is about seven. It means that each particle spends average about 7 iterations to find a better *pbest* when dealing the six functions. Therefore, the lower limit should be larger than seven to leave enough time for most particles to search. Based on this, we run PSO with particle replacement on the six functions using 15 different τ from 7 to 21. The experiments is also run 30 times on each level and the average of the final fitness value are plotted in Fig.3. Four test functions, including Ackley, Rastrigin, Weistrass and Schwefel functions, are very sensitive to τ and three of them achieve the best results when τ is 19. The other two test functions, i.e. Rosenbrock and Griewank functions also get good result when τ is around 19. So, the maximal age τ is set at 19 for all benchmark functions used in this paper.

Table 1. Statistic results of α for the six test functions.

	f_1	f_2	f_3	f_4	f_5	f_6
α	7.9±4.1	5.7±1.6	5.4±1.3	9.0±4.1	4.8±1.1	7.1±2.5

5. Experiments Results

The experiments conducted in this paper are divided into two parts: the first part is to investigate the respective influence of the four proposed schemes to PSO and the second part compares PSOA with other well-known PSOs.

5.1. *Test functions*

In order to test the PSOA on multimodal functions and compare it to other algorithms, we choose fifteen widely used multimodal test functions from [5,44,45]. The fifteen test functions are listed in Table 2, where the dimension D is set to 30 in this paper. The x_{min} for the functions $f_1 - f_5$ and $f_7 - f_{11}$ is $\{0\}^D$, for the functions f_6 and f_{12} is $\{420.96\}^D$ and for the functions $f_{13} - f_{15}$ is $\{f\_bias\}^D$. The column 'Range' represents the initialization range and search range.

The function $f_1 - f_6$ are six basic multimodal functions. The Rosenbrock function f_1 is unimodal in a 2-D or 3-D search space but can be seen as multimodal function in high dimension space[7,46]. In the six functions, the f_1 and f_6 are nonseparable and the other four are separable. The functions $f_7 - f_{12}$ are the rotated version of the functions $f_1 - f_6$. The last three functions $f_{13} - f_{15}$ are three more difficult shifted rotated multimodal functions, which are proposed in CEC 2005[45]. To make problems non-

Table 2: Benchmark Test Functions

Function[a]	Range	f_{min}	Accuracy
$f_1 = \sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	$[-30,30]^D$	0	1.0e+02
$f_2 = \sum_{i=1}^{D}\frac{1}{4000}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	$[-600,600]^D$	0	1.0e-06
$f_3 = 20+e-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))$	$[-30,30]^D$	0	1.0e-06
$f_4 = \sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i))+10D$	$[-5.12,5.12]^D$	0	1.0e-06
$f_5 = \sum_{i=1}^{D}(\sum_{k=0}^{20}[0.5^k\cos(2\pi 0.3^k(x_i+0.5))])-D\sum_{k=0}^{20}0.5^k\cos(\pi 0.3^k)$	$[-0.5,0.5]^D$	0	1.0e-06
$f_6 = 418.9829D+\sum_{i=1}^{D}[-x_i\sin(\sqrt{\lvert x_i\rvert})]$	$[-500,500]^D$	0	2.0e+03
f_7[b]$= \sum_{i=1}^{D-1}[100(y_{i+1}-y_i^2)^2+(y_i-1)^2]$	$[-100,100]^D$	0	1.0e-02
f_8[b]$= \sum_{i=1}^{D}(y_i^2-10\cos(2\pi y_i))+10D$	$[-5.12,5.12]^D$	0	1.0e+02
f_9[b]$= 20+e-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}y_i^2})-exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi y_i))$,	$[-30,30]^D$	0	1.0e-06
f_{10}[b]$= \sum_{i=1}^{D}(\sum_{k=0}^{20}[0.5^k\cos(2\pi 0.3^k(y_i+0.5))])-D\sum_{k=0}^{20}0.5^k\cos(\pi 0.3^k)$,	$[-0.5,0.5]^D$	0	1.0e-06
f_{11}[b]$= \sum_{i=1}^{D}\frac{1}{4000}y_i^2-\prod_{i=1}^{D}\cos(\frac{y_i}{\sqrt{i}})+1$	$[-600,600]^D$	0	1.0e-06
f_{12}[b]$= 418.9829D+\sum_{i=1}^{D}[-y_i\sin(\sqrt{\lvert y_i\rvert})]$,	$[-500,500]^D$	0	5.0e+03
f_{13}[c]$= \sum_{i=1}^{D}\frac{1}{4000}z_i^2-\prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}})+1+f\_bias, f\_bias=-180$,	$[-600,600]^D$	-180	1.0e-01
f_{14}[c]$= \sum_{i=1}^{D}(z_i^2-10\cos(2\pi z_i))+10D+f\_bias, f\_bias=-330$	$[-5,5]^D$	-330	1.0e-01
f_{15}[c]$= \sum_{i=1}^{D}(\sum_{k=0}^{20}[0.5^k\cos(2\pi 0.3^k(z_i+0.5))])-D\sum_{k=0}^{20}0.5^k\cos(\pi 0.3^k)$ $+f\_bias, f\_bias=90$	$[-0.5,0.5]^D$	90	1.0e-01

[a] f_1 Rosenbrock function, f_2 Griewank function, f_3 Ackley function, f_4 Rastrigin function, f_5 Weiestrass function, f_6 Schwefel function, f_7 Rotated Rosenbrock function, f_8 Rotated Rastrigin function, f_9 Rotated Ackley function, f_{10} Rotated Weiestrass function, f_{11} Rotated Griewank function, f_{12} Rotated Schwefel function, f_{13} Shifted Rotated Griewank function, f_{14} Shifted Rotated Rastrigin function, f_{15} Shifted Rotated Weiestrass function;
[b] **y=M*****x**,**M** is an orthogonal matrix;
[c] **z=(x-o)*****M**, **M** is an orthogonal matrix and **o** is the shifted global optimum.

Table 3: Nine PSOs used for comparision

Name	Parameters Setting
GPSO	$\omega=0.729, c_1=c_2=1.49445, V_{max}=0.5*R$[a]$, P_m=0.1$.
SPSO07	$\omega=1/(2*\ln(2)), c_1=c_2=0.5+\ln(2), K=3$.
FIPS	$\omega=0.729, \sum c=4.1, V_{max}=0.5*R$.
DMS-PSO	$\omega=0.9\sim 0.2, c_1=c_2=2.05, V_{max}=0.2*R, m=3, R=5$.
CLPSO	$\omega=0.9\sim 0.4, c=1.49445, V_{max}=0.2*R, m=7$.
IPSO	$\omega=0.729, c_1=c_2=1.49445, V_{max}=0.5*R, ps_{min}=1, ps_{max}=1000$.
EPUS-PSO	$\omega=1/(2*\ln(2)), c_1=c_2=0.5+\ln(2), k=3, ps_{min}=1, ps_{max}=20$.
PSOA	$\omega=0.729, c_1=c_2=1.49445, \tau=19, K=3, P_m=0.2$.

[a] R means the search range.

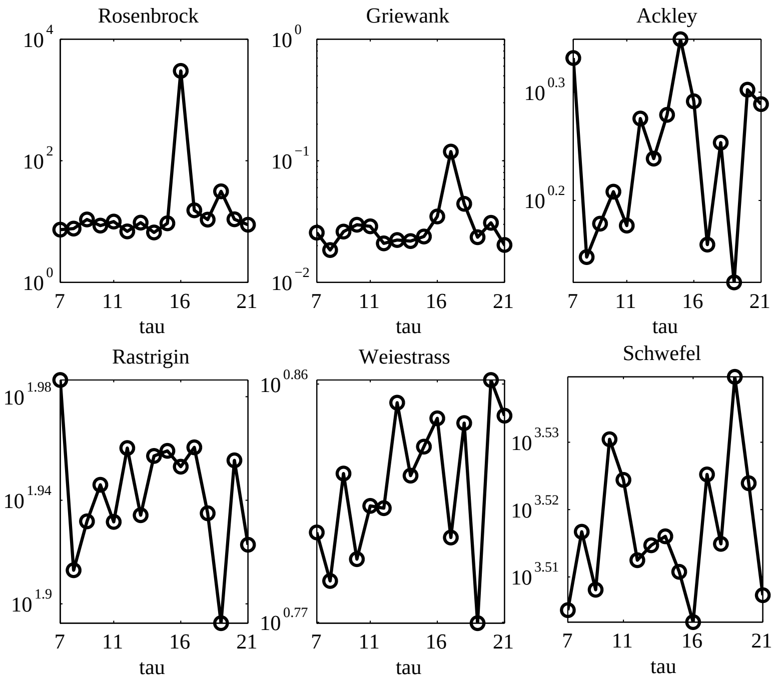

Figure 3: The variance of the final fitness value of PSO against the parameter τ (tau) on the six benchmark functions.

separable, an orthogonal matrix $\mathbf{M}$ is used to rotate coordinate. The original vector $\mathbf{x}$ is multiplied by $\mathbf{M}$ to generate a vector $\mathbf{y} = \mathbf{M} * \mathbf{x}$. In this paper, the orthogonal matrix $\mathbf{M}$ is generated through Salomon's method[5,29,47].

Table 2 also shows the search range of each dimension of the particle (column 2), the global optimal fitness f_{min}(column 3) and the acceptable accuracy for each function (column 4). In our following experiment, the initialization range is the same with the search range. The accuracy index [7] is used to measure the desired accuracy for each function.

5.2. Experimental setup

Experiments will be conducted to compare PSOA with seven existing PSO variants on the fifteen test functions. Table 3 lists the algorithms name and parameters configuration. All parameter settings are based on the suggestions in the corresponding references. The first PSO is Gaussian PSO (GPSO) where a Gaussian mutation is used [48]. The second is Clerc's standard PSO 2007 version (SPSO07) [14], where a random topology is used. The third one is fully informed particle swarm (FIPS)[31] where each particle is related to all other particles of the population. The fourth one is dynamic multi-swarm PSO (DMS-PSO)[13] where a dynamic random topology is applied to organize learning among particles. The fifth one is comprehensive learning PSO (CLPSO) [5] that uses all others personal best position to update a particles velocity. The sixth one is incremental PSO (IPSO) [6] which uses the incremental social learning to control the population size dynamically. The last PSO for comparisons is the efficient population utilization strategy PSO (EPUS-PSO) [29] in which a dynamic population size is also employed.

All algorithms were programmed and compiled under Matlab R2011b on an Intel Core i5-760 2.8GHz computer running Microsoft Windows 7. In the following experiments, the population size is set to 40 for all algorithms except IPSO and EPUS-PSO. In IPSO and EPUS-PSO, the minimum population size of both is set to 1 and the maximum population size is set to 1000 and 20 respectively, according to their original setting in[6,29]. Furthermore, all the algorithms uses the same maximum number of fitness evaluations (FEs) 2.0e+05 in each run for all functions. In experimental setup, each algorithm is run with the same parameter setting across all test functions. To verify the effectiveness of the proposed algorithms, each algorithm is run 30 times independently for every function and the mean value and standard deviation value are calculated. Besides, two additional indexes success rate(SR) and convergence speed(CS) are used for algorithm comparisons. The SR index is the percentage of runs that reach the desired accuracy and the CS is measured on the mean number of FEs required to reach an acceptable solution among successful runs[7]. Note that, in the all following experimental results tables,the better results on each index are in bold.

5.3. Effects of the proposed strategies

In this section, we investigate whether the proposed strategies, i.e. particle replacement, neighbours selection, hypermutation and inertia weight adjustment, can significantly improve PSO 's performance on multimodal function. To do this, we add each of the three ideas to basic PSO one by one to produce four different algorithm as shown in Table 4. We named them PSOA, PSOA1, PSOA2 and PSOA3 respectively. Then we run basic PSO, PSOA and the three PSOA variants on the fifteen test functions listed in Table 2, 30 times independently.

Table 4. Relation between the four different PSOA versions.

Name	Content
PSOA	PSO + *ParticleReplace*
PSOA1	PSOA + *NeighborSelection*
PSOA2	PSOA1 + *Hypermutation*
PSOA3	PSOA2 + *InertiaWeightAdjust*

Table 5 compares the experimental results of the five algorithms. The better results on each test function are highlighted in bold. It is noticeable that the performance fluctuates a lot when different aging operators are used. Firstly, the basic PSO gave better results than any PSOA variants on f_1. For PSOA, there was no significant enhancement on all test functions compared to basic PSO though particle replacement was used in it. On the contrary, PSOA using an age based random topology (PSOA1) got a better performance than PSO and PSOA on most functions except f_1 and f_7. In addition, PSOA1 gave the best results on the function f_{15}. Furthermore, PSOA2 gave a better solution than PSOA1 on all functions except f_{13}-f_{15}. On the function f_8, f_{10}, f_{11} and f_{12}, PSOA2 achieved the best results among the five PSOs. Finally, PSOA3 achieved the best accuracy on the function $f_2 - f_7$, f_9, f_{13} and f_{14}. Table 5 also compares the *t*-test results among the five PSOs. The first group of *t*-test is between PSOA and PSO, it can be noted from the results (B vs A column) that PSOA performed worse than PSO on all of the fifteen functions. It is obvious that PSOA1 was significantly better than PSOA on 12 out of the 15 benchmarks. Furthermore, PSOA2 outperformed PSOA1 on nine functions. From results of the last group (PSOA3 vs PSOA2), PSOA3 performed significantly better than PSOA2 on ten functions.

Table 6 compares the success rate and convergence speed of PSO and the four PSOA variants. On success rate measure, the basic PSO only performed well on the function f_1 and the PSOA totally converged to the desired accuracy on the function f_1 and f_{12} in the 30 runs. Although the average success rate of PSOA1 (55.3%) was significant better than PSO and PSOA, it only gave 100% success rate on the function f_3 and f_{12}. Compared to PSOA1, PSOA2 showed better results of success rate on nine test functions, i.e. $f_1 - f_5$ and $f_8 - f_{11}$. Finally, PSOA3 got the best average success rate (75.3%) among the five PSO variants and was converged on nine test functions. Another measure is about the convergence speed in the successful runs. On this measure, PSO, PSOA and PSOA3 gave better results on one, four and seven test functions, respectively.

From the statistical results shown in Table 5, it seems that the particle replacement can not improve the performance of PSO. This is understandable because there is no additional mechanism to allow other trapped particles to learn the information in the new ones. On the contrary, the new particles are easily attracted by these particles in local optima. This conjecture is verified in the results of PSOA1. When the age-based topology is added to PSOA, its solution accuracy, converge speed and success rate on most test functions were all enhanced substantially. This is because the new born and young particles can provide more promising information than the *gbest*. More important, the age-based population topology provides the chance for trapped particles to learn these information and escape from local optima. Fig.4 compares the change of population diversity and fitness value when applying PSO, PSOA and PSOA1 on Rastrigin functions f_4. The results are the average value of 30 independent runs and the diversity is computed according to Morrison and De Jong's moment of inertia diversity measure [49]. From Fig.4, it is clearly observed that PSOA failed to maintain diversity during search but PSOA1 was able to keep an effective diversity in a high level. It is noted that the similar tendency happened on all of the other fourteen test functions.

5.4. *Comparisons with other PSOs*

In this section, the PSOA2 and PSOA3 are compared with other seven improved PSO variants, which are listed in Table 3. Table 7 shows the average value and standard deviation results of the nine PSOs. PSOA2 gave the better results on f_8, f_{10}, f_{11} and f_{15}. PSOA3 performed the best on the eight out of the fifteen functions, i.e. $f_2 - f_5$, f_7,f_9 and $f_{13} - f_{14}$. FIPS got better results on the function f_{12} and CLPSO performed very well on the function f_1 and f_6. Overall, it can be observed that PSOA2 and PSOA3 gave the better results on most of the test functions.

Table 8 lists the t-test results at the confidence level of 5% between PSOA3 and the other seven PSO variants (except PSOA2). "+" and"-" indicate that PSOA3 is significantly better and worse than the compared algorithm, respectively. "$\approx$" indicates the difference is not significant. From Table 8, it is obvious that PSOA3 was significantly better than all the other seven PSO variants on the function f_2, f_4, f_5, f_8 and f_{15}. On the function f_3, f_9, f_{12} and f_{13}, PSOA3 outperformed six out of the seven compared PSOs. On the function f_1, f_{10} and f_{14}, it beat five out the seven algorithms. In addition, it performed better on four of the functions on the function f_7 and f_{11}. Finally, it performed poorly on the Schwefel function f_6. From another perspective, for GPSO and DMS-PSO, it performed significantly better than them on most of the test functions and worse than them on the function f_6. For SPSO07, IPSO and EPUS-PSO, they failed to performed well on any of the test functions. For FIPS, the proposed algorithm was better than it on twelve out of the fifteen functions but defeated on the function f_6 and f_{12}. For CLPSO, the proposed algorithm also beat it on twelve test functions and performed bad on three functions, i.e. f_1,f_6 and f_{11}.

Table 9 compares the success rate and converge speed in the successful runs of the nine PSO variants. The results show that PSOA2 and PSOA3 achieved the highest average success rate 70.7% and 75.3% respectively, DMS-PSO ranked second at 62.7%, followed by CLPSO, SPSO07, GPSO(FIPS), IPSO and EPUS-PSO. Moreover, PSOA3 converged very fast on most of the functions and gave the best performance on the function f_1, f_4, f_5 and $f_8 - f_{10}$. GPSO was fastest on f_6 and f_{12} and SPSO07 performed fastest on f_{13}. In addition, IPSO is very fast on f_3 and EPUS-PSO performed well on f_2 and f_{11}. Fig.5 and 6 show the convergence plot of PSOA3, DMS-PSO, CLPSO and SPSO07, which all performed very well on the fifteen test functions in the seven compared PSOs. It can be noted that PSOA3 converged fast to the best solution on most of the test functions, especially on $f_2 - f_5$, f_8, f_9, f_{12} and f_{15}. Overall, compared to the other seven PSOs, PSOA2 and PSOA3 show better performance both on the success rate and the convergence speed.

6. Conclusion

In this paper, we presented a novel PSOA to cope with premature converge when solving multimodal functions. Firstly, we gave a new definition of age for PSO, which used the update of *pbest* of each particle to determine its age. One useful feature of *particle age* is that it can recognize both fitness stagnation and fitness oscillation. Based on this, we first introduced the original aging operator (particle replacement) into PSO. However, many valuable information in the new and young particles were not fully utilized. To make use of these information, three age related operators were proposed in this paper, i.e. age based neighbourhood selection, hyper-

Table 5: Results with four variations of PSOA and PSO on the 15 test functions

Function		PSO	PSOA	PSOA1	PSOA2	PSOA3	*t*-Test[a]			
		A	B	C	D	E	B vs A[b]	C vs B	D vs C	E vs D
f_1	avg.	**7.85e+00**	9.48e+00	3.14e+01	2.19e+01	1.84e+01	$\approx$	$-$	$\approx$	$+$
	std.	**5.70e+00**	1.59e+01	3.31e+01	2.31e-01	3.60e-01				
f_2	avg.	1.23e-02	6.12e-02	7.14e-03	1.18e-15	**0.00e+00**	$\approx$	$\approx$	$+$	$+$
	std.	1.63e-02	1.87e-01	1.05e-02	1.25e-15	**0.00e+00**				
f_3	avg.	2.22e+00	1.49e+00	1.80e-12	1.64e-12	**7.70e-15**	$\approx$	$+$	$\approx$	$+$
	std.	2.47e+00	8.56e-01	1.83e-12	1.75e-12	**1.35e-15**				
f_4	avg.	8.22e+01	9.15e+01	2.41e+01	1.17e-14	**0.00e+00**	$\approx$	$+$	$+$	$+$
	avg.	1.79e+01	2.70e+01	1.78e+01	7.81e-15	**0.00e+00**				
f_5	avg.	6.90e+00	7.08e+00	1.84e-01	1.84e-13	**0.00e+00**	$\approx$	$+$	$+$	$+$
	std.	2.83e+00	3.05e+00	4.84e-01	1.54e-13	**0.00e+00**				
f_6	avg.	3.30e+03	3.35e+03	2.08e+03	2.40e+03	**1.94e+03**	$\approx$	$+$	$\approx$	$+$
	std.	6.13e+02	5.87e+02	6.05e+02	7.36e+02	**5.22e+02**				
f_7	avg.	2.53e+01	2.59e+01	6.08e+01	2.34e+01	**2.17e+01**	$\approx$	$-$	$+$	$+$
	std.	2.52e+01	2.23e+01	4.42e+01	2.87e-01	**1.41e+00**				
f_8	avg.	9.38e+01	1.01e+02	6.48e+01	**1.34e-14**	1.03e+01	$\approx$	$+$	$+$	$-$
	std.	2.67e+01	3.07e+01	1.80e+01	**7.53e-15**	9.29e+00				
f_9	avg	2.17e+00	2.29e+00	1.97e-01	1.52e-12	**6.87e-15**	$\approx$	$+$	$+$	$+$
	std.	7.41e-01	8.67e-01	5.20e-01	2.37e-12	**1.30e-15**				
f_{10}	avg.	1.41e+01	1.40e+01	4.98e+00	**4.44e-06**	1.77e-01	$\approx$	$+$	$+$	$\approx$
	std.	2.91e+00	3.67e+00	2.40e+00	**1.07e-05**	7.98e-01				
f_{11}	avg.	2.21e-02	1.68e-02	4.43e-03	**2.83e-15**	2.14e-03	$\approx$	$+$	$+$	$-$
	std.	2.37e-02	2.15e-02	7.15e-03	**2.04e-15**	5.00e-03				
f_{12}	avg.	3.54e+03	3.60e+03	2.44e+03	**1.54e+03**	1.93e+03	$\approx$	$+$	$+$	$-$
	std.	8.09e+02	6.64e+02	6.58e+02	**2.79e+02**	4.41e+02				
f_{13}	avg.	-156.203	-158.085	-179.967	-178.261	**-179.980**	$\approx$	$+$	$-$	$+$
	std.	6.97e+01	4.50e+01	3.75e-02	4.34e+00	**1.28e-02**				
f_{14}	avg.	-199.865	-209.418	-273.955	-264.267	**-289.445**	$\approx$	$+$	$-$	$+$
	std.	4.18e+01	2.98e+01	1.44e+01	1.97e+01	**1.16e+01**				
f_{15}	avg.	118.101	118.070	**110.204**	110.343	110.419	$\approx$	$+$	$\approx$	$\approx$
	std.	3.59e+00	3.95e+00	**3.76e+00**	4.31e+00	3.69e+00				
Total		1	0	1	4	**9**	0+	12+	9+	10+

[a] A *t*-test with significance level of 0.05 and freedom of 58 degrees was conducted;

[b] 'B vs A' means a *t*-test is executed between A and B, '$+$' and '$-$' indicate B is significantly better and worse than A, respectively. '$\approx$' means the difference between B and A is not statistically significant.

Table 6: Success rate and convergence speed in successful runs of the five PSOs

Function	PSO		PSOA		PSOA1		PSOA2		PSOA3	
	SR	CS	SR	CS	SR	CS	SR	CS	SR	CS
f_1	**100%**	1.61e+04	**100%**	1.79e+04	97%	4.10e+04	**100%**	1.42e+04	**100%**	**9.09e+03**
f_2	40%	**2.10e+04**	13%	2.25e+04	57%	4.16e+04	**100%**	3.57e+04	**100%**	2.16e+04
f_3	10%	3.22e+04	13%	3.26e+04	**100%**	5.90e+04	**100%**	4.91e+04	**100%**	**2.99e+04**
f_4	0%	$\times$[b]	0%	$\times$	0%	$\times$	**100%**	3.91e+04	**100%**	**2.86e+04**
f_5	0%	$\times$	0%	$\times$	83%	1.02e+05	**100%**	5.64e+04	**100%**	**3.72e+04**
f_6	3%	4.32e+03	0%	$\times$	53%	6.34e+04	27%	9.16e+04	**57%**	**5.51e+04**
f_8	70%	7.26e+03	50%	**6.94e+03**	97%	1.96e+04	**100%**	1.61e+04	**100%**	8.58e+04
f_9	3%	3.22e+04	0%	$\times$	83%	7.06e+04	**100%**	4.98e+04	**100%**	**2.98e+04**
f_{10}	0%	$\times$	0%	$\times$	0%	$\times$	70%	7.37e+04	**90%**	**4.48e+04**
f_{11}	30%	2.15e+04	23%	**2.03e+04**	67%	4.23e+04	**100%**	3.37e+04	83%	2.07e+04
f_{12}	93%	2.52e+03	**100%**	**2.42e+03**	**100%**	4.68e+03	**100%**	4.86e+03	**100%**	4.43e+03
f_{13}	23%	3.12e+04	37%	**3.00e+04**	93%	5.91e+04	63%	1.21e+05	**100%**	8.19e+04
avg.[a]	24.6%		22.4%		55.3%		70.7%		**75.3%**	

[a] results for the function f_7, f_{14} and f_{15} are not shown in this table because none of the eight PSOs is able to achieve the desired accuracy on them;

[b] '$\times$' means the corresponding algorithm fails to converge to the desired region once during the 30 runs.

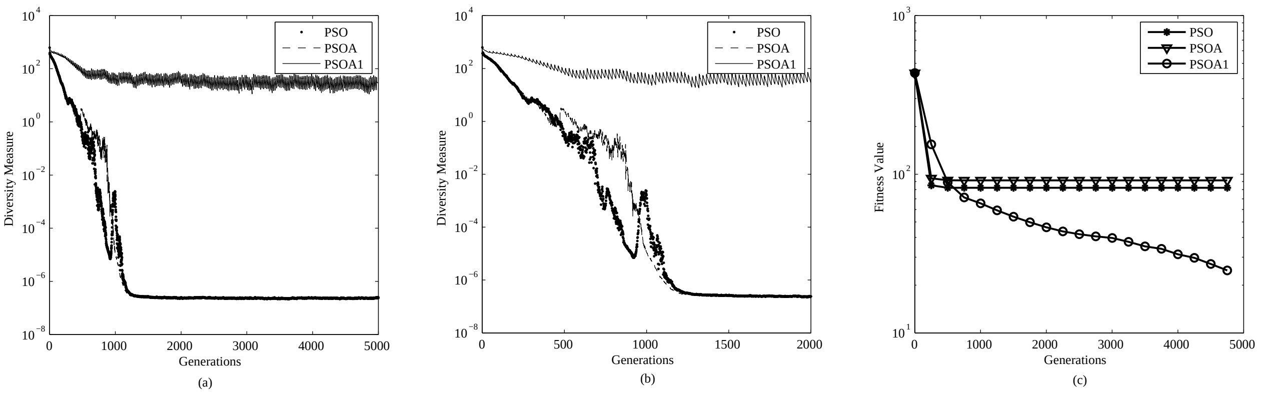

Figure 4: Comparision of PSO, PSOA and PSOA1's diversity and convergence on the function Rastrigin f_4. (a) diversity plot. (b) enlarged view of the diversity plot between 1 and 2000 generations. (c) convergence plot.

Table 7: Results with PSOA and the compared eight PSOs on the 15 functions with 30-D

Function		GPSO	SPSO07	FIPS	DMS-PSO	CLPSO	IPSO	EPUS-PSO	PSOA2	PSOA3
f_1	avg.	3.56e+01	4.04e+01	2.72e+01	3.06e+01	**7.58e+00**	4.36e+01	3.50e+01	2.19e+01	1.84e+01
	std.	3.16e+01	3.10e+01	1.12e+01	2.03e+01	**7.94e+00**	3.80e+01	3.80e+01	2.31e-01	3.60e-01
f_2	avg.	8.54e-03	5.50e-03	7.64e-05	4.81e-08	2.02e-09	9.02e-03	1.56e-02	1.18e-15	**0.00e+00**
	std.	9.68e-03	7.56e-03	3.31e-04	2.62e-07	2.85e-09	1.32e-02	1.67e-02	1.25e-15	**0.00e+00**
f_3	avg.	6.30e-09	3.10e-02	3.64e-07	6.55e-14	1.54e-06	7.65e-08	2.62e+01	1.64e-12	**7.70e-15**
	std.	6.29e-09	1.70e-01	9.97e-08	2.95e-14	5.26e-07	1.28e-07	6.80e-01	1.75e-12	**1.35e-15**
f_4	avg.	5.06e+00	3.69e+01	6.99e+01	2.22e+01	9.69e-09	2.17e+01	6.36e+01	1.17e-14	**0.00e+00**
	std.	5.98e+00	1.04e+01	1.05e+01	2.24e+00	7.31e-09	1.15e+01	2.07e+01	7.81e-15	**0.00e+00**
f_5	avg.	4.43e-04	1.34e-01	7.11e-02	1.97e-14	1.13e-07	2.31e-01	1.51e+01	1.84e-13	**0.00e+00**
	std.	3.86e-04	2.63e-01	7.87e-02	1.18e-14	6.82e-08	5.28e-01	2.48e+01	1.54e-13	**0.00e+00**
f_6	avg.	1.62e+03	3.22e+03	6.98e+02	1.62e+03	**3.82e-04**	2.16e+03	1.22e+03	2.40e+03	1.94e+03
	std.	2.92e+02	8.06e+02	4.56e+02	3.26e+02	**1.68e-09**	4.12e+02	2.20e+03	7.36e+02	5.22e+02
f_7	avg.	7.22e+01	2.38e+01	2.60e+01	2.53e+01	8.85e+01	3.93e+01	2.56e+01	2.34e+01	**2.16e+01**
	std.	1.37e+02	1.49e+01	7.88e-01	1.72e+00	2.76e+01	2.98e+01	1.37e+01	2.87e-01	**1.41e+00**
f_8	avg.	5.14e+01	4.44e+01	1.29e+02	3.40e+01	3.83e+01	6.06e+01	8.80e+01	**1.34e-14**	1.03e+01
	std.	1.68e+01	1.08e+01	1.52e+01	3.97e+00	5.75e+00	1.82e+01	3.43e+01	**7.53e-15**	9.29e+00
f_9	avg.	7.83e-01	6.21e-02	5.02e-07	7.41e-14	5.06e-05	9.49e-01	4.44e+00	1.52e-12	**6.87e-15**
	std.	7.79e-01	2.36e-01	1.54e-07	4.75e-14	3.28e-05	7.46e-01	1.20e+00	2.73e-12	**1.30e-15**
f_{10}	avg.	1.03e+01	1.52e+00	3.33e-01	6.85e-02	2.96e+00	9.47e+00	1.80e+01	**4.44e-06**	1.78e-01
	std.	3.09e+00	1.95e+00	6.44e-01	5.81e-02	8.01e-01	2.76e+00	3.36e+00	**1.07e-15**	7.99e-01
f_{11}	avg.	1.37e-02	6.73e-03	6.73e-03	3.89e-06	4.76e-05	1.26e-02	1.07e-02	**2.83e-15**	2.13e-03
	std.	1.66e-02	8.54e-03	9.53e-05	1.88e-05	4.31e-05	1.13e-02	1.44e-02	**2.04e-15**	5.00e-03
f_{12}	avg.	2.88e+03	2.61e+03	**1.10e+03**	4.88e+03	2.66e+03	3.74e+03	4.32e+03	1.54e+03	1.93e+03
	std.	6.99e+02	6.37e+02	**8.24e+02**	2.78e+02	2.53e+02	6.75e+02	7.40e+02	2.79e+02	4.41e+02
f_{13}	avg.	-179.980	-179.978	-179.924	-179.926	-178.323	-179.979	-179.971	-178.261	**-179.984**
	std.	1.55e-02	1.62e-02	4.28e-02	1.03e-01	2.71e-01	5.02e-01	2.51e-02	4.34e+00	**1.28e-02**
f_{14}	avg.	-214.017	-282.550	-147.447	-274.534	-218.698	-212.691	-70.730	-264.267	**-289.445**
	std.	4.21e+01	1.64e+01	1.04e+01	5.38e+00	1.60e+01	4.00e+01	8.69e+02	1.91e+01	**1.16e+01**
f_{15}	avg.	115.034	122.385	129.161	117.199	117.781	113.874	124.348	**110.343**	110.429
	std.	3.95e+00	2.12e+00	1.17e+00	1.17e+00	2.12e+00	3.54e+00	3.61e+00	**4.31e+00**	3.69e+00
	Total	0	0	1	0	2	0	0	4	**8**

Table 8: *t*-test results between PSOA3 and GPSO, SPSO07, FIPS, DMS-PSO, CLPSO, IPSO and EPUS-PSO

PSOA3 Versus	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	Total		
																$+$	$\approx$	$-$
GPSO	$+$	$+$	$+$	$+$	$+$	$-$	$\approx$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	13	1	1
SPSO07	$+$	$+$	$\approx$	$+$	$+$	$+$	$\approx$	$+$	$\approx$	$+$	$\approx$	$+$	$+$	$\approx$	$+$	10	5	0
FIPS	$+$	$+$	$+$	$+$	$+$	$-$	$+$	$+$	$+$	$\approx$	$+$	$-$	$+$	$+$	$+$	12	1	2
DMS-PSO	$+$	$+$	$+$	$+$	$+$	$-$	$+$	$+$	$+$	$\approx$	$\approx$	$+$	$+$	$+$	$+$	12	2	1
CLPSO	$-$	$+$	$+$	$+$	$+$	$-$	$+$	$+$	$+$	$+$	$-$	$+$	$+$	$+$	$+$	12	0	3
IPSO	$+$	$+$	$+$	$+$	$+$	$\approx$	$+$	$+$	$+$	$+$	$+$	$+$	$\approx$	$+$	$+$	13	2	0
EPUS-PSO	$\approx$	$+$	$+$	$+$	$+$	$\approx$	$\approx$	$+$	$+$	$+$	$+$	$+$	$+$	$\approx$	$+$	11	4	0
$+$	5	7	6	7	7	1	4	7	6	5	4	6	6	5	7	83		
Total $\approx$	1	0	1	0	0	2	3	0	1	2	3	0	1	2	0		15	
$-$	1	0	0	0	0	4	0	0	0	0	1	1	0	0	0			7

Table 9: Comparisons of success rate(SR)and convergence speed (CS) in successful runs on the 15 functions with 30-D

Function		GPSO	SPSO07	FIPS	DMS-PSO	CLPSO	IPSO	EPUS-PSO	PSOA2	PSOA3
f_1	SR	**100%**	**100%**	**100%**	97%	**100%**	97%	93%	**100%**	**100%**
	CS	3.41e+04	1.41e+04	6.26e+04	1.24e+05	9.92e+04	3.98e+04	2.32e+04	1.42e+04	**9.09e+03**
f_2	SR	37%	57%	67%	97%	**100%**	53%	20%	**100%**	**100%**
	CS	1.55e+05	2.20e+04	1.54e+05	1.45e+05	1.60e+05	8.83e+04	**1.60e+04**	3.57e+04	2.16e+04
f_3	SR	**100%**	97%	**100%**	**100%**	23%	**100%**	0%	**100%**	**100%**
	CS	1.84e+05	3.15e+04	1.89e+05	1.51e+05	1.98e+05	**1.52e+04**	$\times$[b]	4.91e+04	2.99e+04
f_4	SR	7%	0%	0%	0%	**100%**	0%	0%	**100%**	**100%**
	CS	1.76e+05	$\times$	$\times$	$\times$	1.75e+05	$\times$	$\times$	3.91e+04	**2.87e+04**
f_5	SR	0%	20%	0%	**100%**	**100%**	0%	0%	**100%**	**100%**
	CS	$\times$	4.55e+05	$\times$	1.67e+05	1.89e+05	$\times$	$\times$	5.64e+04	**3.72e+04**
f_6	SR	87%	0%	97%	**100%**	**100%**	37%	17%	27%	57%
	CS	**2.11e+04**	$\times$	1.04e+05	1.19e+05	2.97e+04	9.60e+04	6.05e+04	9.16e+04	5.51e+04
f_8	SR	**100%**	**100%**	3%	**100%**	**100%**	**100%**	70%	**100%**	**100%**
	CS	9.54e+03	2.07e+04	1.92e+05	2.93e+04	7.03e+04	1.41e+04	6.96e+03	1.61e+04	**8.58e+03**
f_9	SR	47%	93%	**100%**	**100%**	0%	27%	0%	**100%**	**100%**
	CS	1.83e+05	3.19e+04	1.92e+05	1.60e+05	$\times$	1.57e+05	$\times$	4.98e+04	**2.98e+04**
f_{10}	SR	0%	0%	0%	0%	0%	0%	0%	70%	**90%**
	CS	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	7.37e+04	**4.48e+04**
f_{11}	SR	37%	53%	70%	90%	0%	30%	43%	**100%**	83%
	CS	1.37e+05	2.35e+04	1.46e+05	1.58e+05	$\times$	7.03e+04	**1.56e+04**	3.37e+04	2.07e+04
f_{12}	SR	**100%**	**100%**	**100%**	77%	**100%**	**100%**	40%	**100%**	**100%**
	CS	**3.26e+03**	8.11e+03	1.87e+04	1.22e+05	3.35e+04	2.07e+04	3.09e+04	4.86e+03	4.44e+03
f_{13}	SR	**100%**	**100%**	77%	80%	0%	80%	97%	63%	**100%**
	CS	1.13e+05	**2.95e+04**	1.75e+05	1.78e+05	$\times$	9.07e+04	3.19e+04	1.21e+05	8.19e+04
	avg.SR[a]	47.6%	48.0%	47.6%	62.7%	48.2%	41.6%	25.3%	70.7%	**75.3%**

[a] results for the function f_7, f_{14} and f_{15} are not shown in this table because none of the eight PSOs is able to achieve the desired accuracy on them;

[b] '$\times$' means the corresponding algorithm fails to converge to the desired region once during the 30 runs.

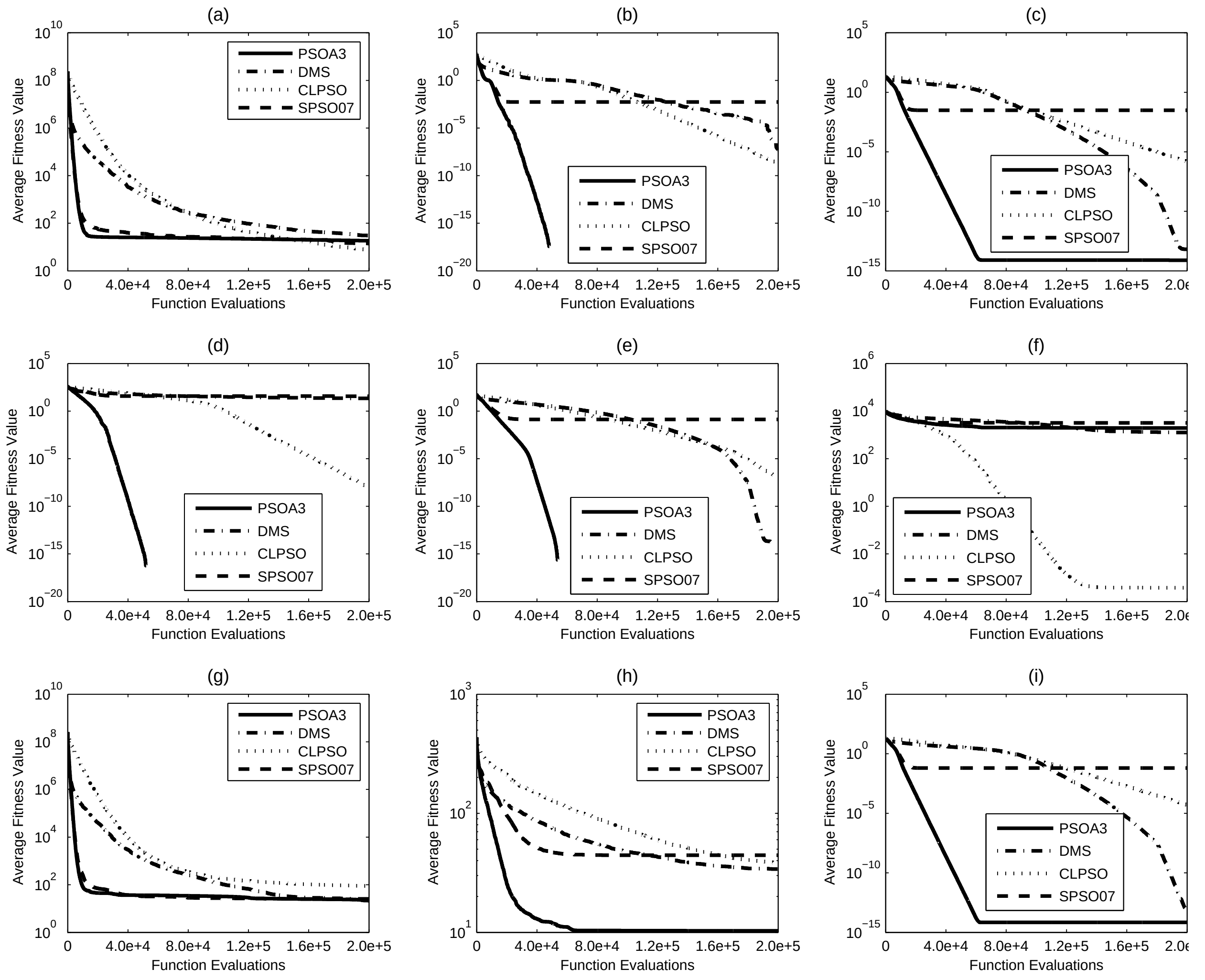

Figure 5: Average best fitness value for PSOA3, DMS-PSO,CLPSO and SPSO07 on the test functions.(a) f_1. (b) f_2. (c) f_3. (d) f_4. (e) f_5. (f) f_6. (g) f_7. (h) f_8. (i) f_9.

mutation and inertia weight adjustment.

Experiments were conducted on 15 multimodal test functions. From the results, we can summarize some distinct features of PSOA variants as follows.

(1) It is found that the isolated particle replacement operator fails to improve PSO significantly on both population diversity and converge accuracy. This is because the added particles are easily attracted by *gbest* particle if there is no other strategy to stop them. On the other hand, the particle replacement operator is necessary for the proposed strategy because it can provide the enough age diversity that is the foundation of the other three operators.

(2) Age based neighbours selection is able to maintain effective population diversity and improve convergence results. From the particle age definition, the young particles are more promising than the old ones. Selecting particles with same or smaller age as neighbours distinguishes it from the other neighbourhood topologies.

(3) Age based hypermutation and inertial weight

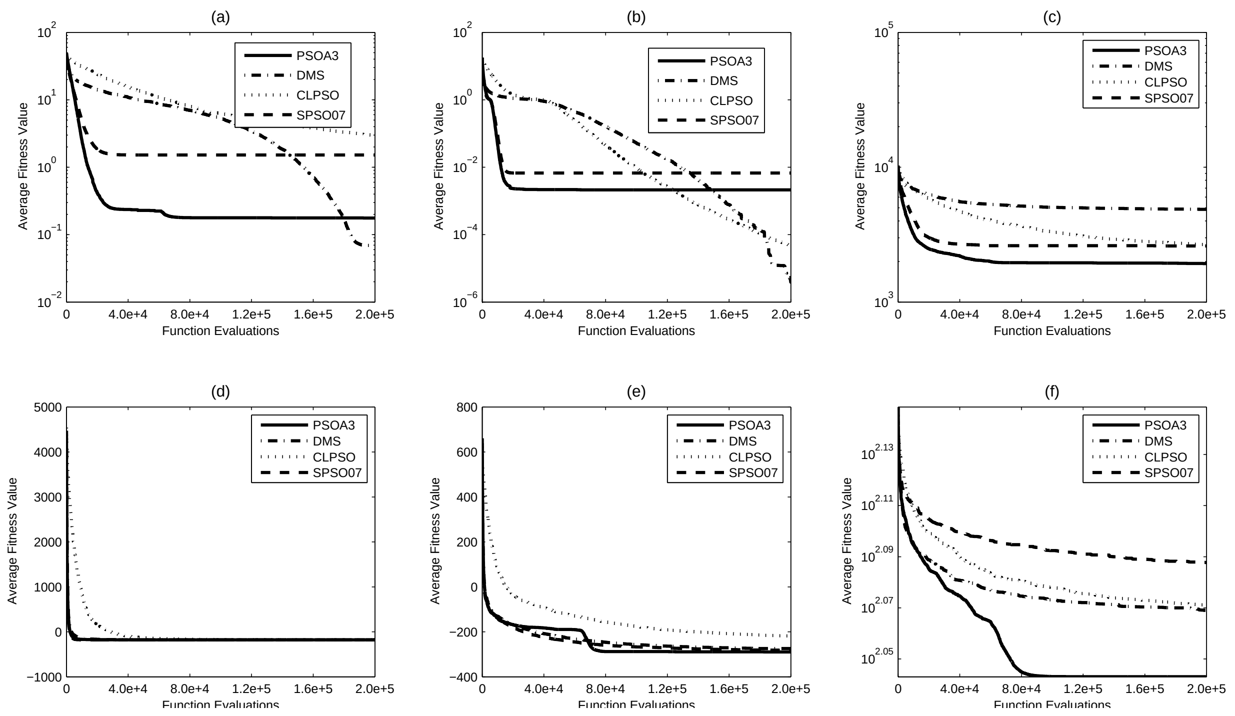

Figure 6: Average best fitness value for PSOA3, DMS-PSO,CLPSO and SPSO07 on the test functions.(a) f_{10}. (b) f_{11}. (c) f_{12}. (d) f_{13}. (e) f_{14}. (d) f_{15}.

decreasing give the older particles more chances to generate better *pbest* so that they can avoid to be discarded. Older particles play a very significant role in retaining effective particles and thereby reducing fitness evaluations that should have to be computed. In addition, hypermutation makes PSOA insensitive to the different initialization condition.

(4) The proposed three PSOAs are less sensitive to coordinate rotation and shift than the other seven compared PSOs. However, they may not be efficient in optimizing nonseparable problems with deep local optima far from the global optima, such as the function f_6.

It can be seen that the proposed three aging operators can prevent particles from getting stuck by the local optima, maintain effectively population diversity, guarantee robustness and improve performance of PSO. In future, we will apply the proposed algorithm to solve some real-world optimization problems, such as data clustering problems and image segmentation problems.

Acknowledgments

The authors would like to thank the two anonymous reviewers for their constructive and valuable suggestions and comments. This work is partly supported by the National Natural Science Foundation of China under grant no. 60974100 and 61070135.

References

1. M. Clerc and J. Kennedy, “The particle swarm - explosion, stability, and convergence in a multidimensional complex space,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, 2002.
2. N. W. Yaping Du and J. Zhang, “An optimum design method based on pso algorithm for neuron controllers,” in *Proc. World Congr. on Intell. Control and Autom.*, vol. 3, 2004, pp. 2617–2621.
3. P. S. Andrews, “An investigation into mutation operators for particle swarm optimization,” in *Proc. IEEE*

Congr. Evol. Comput., 2006, pp. 1044–1051.
4. F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, 2004.
5. J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, 2006.
6. M. A. M. de Oca, T. Stutzle, K. Van den Enden, and M. Dorigo, "Incremental social learning in particle swarms," *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 41, no. 2, pp. 368–384, 2011.
7. Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 832–847, 2011.
8. A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240 – 255, 2004.
9. Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, 1998, pp. 69 –73.
10. Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 39, no. 6, pp. 1362–1381, 2009.
11. J. Sun, W. Fang, V. Palade, X. Wu, and W. Xu, "Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point," *Appl. Math. Comput.*, vol. 218, no. 7, pp. 3763–3775, Dec. 2011.
12. B. Jiang, N. Wang, and X. He, "Asynchronous particle swarm optimizer with relearning strategy," in *Proc. Annu. Conf. IEEE Ind. Electron. Soci.* IEEE, 2011, pp. 2341–2346.
13. J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proc. IEEE Swarm Intell. Symp.*, 2005, pp. 124–129.
14. M.Clerc. Standard particle swarm optimisation from 2006 to 2011. [Online]. Available: www.particleswarm.info
15. Y. Jiang, T. Hu, C. Huang, and X. Wu, "An improved particle swarm optimization algorithm," *Appl. Math. Comput.*, vol. 193, no. 1, pp. 231–239, Oct. 2007.
16. X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, 2010.
17. G. S. Hornby, "A steady-state version of the age-layered population structure ea," in *Genetic Programming Theory and Practice VII*, 2010, pp. 87–102.
18. J. Garca-Nieto and E. Alba, "Restart particle swarm optimization with velocity modulation: a scalability test," *Soft Comput.*, vol. 15, pp. 2221–2232, 2011.
19. C. Horoba, T. Jansen, and C. Zarges, "Maximal age in randomized search heuristics with aging," in *Proc. Genet. Evol. Comput. Conf.*, 2009, pp. 803–810.
20. T. Jansen and C. Zarges, "Aging beyond restarts," in *Proc. Genet. Evol. Comput. Conf.*, 2010, pp. 705–712.
21. V. Cutello, G. Nicosia, and M. Pavone, "An immune algorithm with stochastic aging and kullback entropy for the chromatic number problem," *J. Comb. Optim.*, vol. 14, pp. 9–33, 2007.
22. V. Cutello, G. Nicosia, M. Pavone, and J. Timmis, "An immune algorithm for protein structure prediction on lattice models," *IEEE Trans. Evol. Comput.*, vol. 11, no. 1, pp. 101–117, 2007.
23. H.-P. Schwefel and G. Rudolph, "Contemporary evolution strategies," in *Proc. Eur. Conf. Artif. Life*. Springer-Verlag, 1995, pp. 893–907.
24. N. Kubota and T. Fukuda, "Genetic algorithms with age structure," *Soft Comput.*, vol. 1, pp. 155–161, 1997.
25. T. Jansen and C. Zarges, "Comparing different aging operators," in *Artif. Immun. Syst.*, ser. LNCS, 2009, vol. 5666, pp. 95–108.
26. J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4, 1995, pp. 1942 –1948 vol.4.
27. R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization an overview," *Swarm. Intell.*, vol. 1, pp. 33–57, 2007.
28. X. Yang, J. Yuan, J. Yuan, and H. Mao, "A modified particle swarm optimizer with dynamic adaptation," *Appl. Math. Comput.*, vol. 189, no. 2, pp. 1205–1213, Jun. 2007.
29. S.-T. Hsieh, T.-Y. Sun, C.-C. Liu, and S.-J. Tsai, "Efficient population utilization strategy for particle swarm optimizer," *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 39, no. 2, pp. 444–456, 2009.
30. J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, vol. 2, 2002, pp. 1671 –1676.
31. R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, 2004.
32. R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. and Hum. Sci.*, 1995, pp. 39–43.
33. J. Kennedy, "Stereotyping: improving particle swarm performance with cluster analysis," in *Proc. IEEE Congr. Evol. Comput.*, vol. 2, 2000, pp. 1507–1512.
34. S. Yang and C. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 959–974, 2010.
35. S. Bird and X. Li, "Adaptively choosing niching parameters in a pso," in *Proc. Genet. Evo. Comput.*

Conf., 2006, pp. 3–10.

36. X. Li and X. Yao, “Cooperatively coevolving particle swarms for large scale optimization,” *IEEE Trans. Evol. Comput.*, no. 99, 2011, early Access.
37. A. Ghosh, S. Tsutsui, and H. Tanaka, “Individual aging in genetic algorithms,” in *Aust. and New Zealand Conf. Intell. Inf. Syst.*, 1996, pp. 276 –279.
38. D.-H. Choi, “Cooperative mutation based evolutionary programming for continuous function optimization,” *Oper. Res. Lett.*, vol. 30, no. 3, pp. 195–201, 2002.
39. V. Cutello, G. Nicosia, and M. Pavone, “A hybrid immune algorithm with information gain for the graph coloring problem,” in *Proc. Genet. Evo. Comput. Conf.*, ser. LNCS, 2003, vol. 2723, pp. 171–182.
40. G. Stracquadanio, C. Drago, V. Romano, and G. Nicosia, “An immunological algorithm for doping profile optimization in semiconductors design,” in *Artif. Immun. Syst.*, ser. LNCS, 2010, vol. 6209, pp. 213–222.
41. G. Hornby, “Alps: the age-layered population structure for reducing the problem of premature convergence.” in *Proc. Genet. Evol. Comput. Conf.*, 2006, pp. 815–822.
42. F. van den Bergh and A. Engelbrecht, “A study of particle swarm optimization particle trajectories,” *Inf. Sci.*, vol. 176, no. 8, pp. 937 – 971, 2006.
43. L. N. de Castro and F. J. Von Zuben, “Learning and optimization using the clonal selection principle,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 3, pp. 239–251, 2002.
44. A. Auger, N. Hansen, J. Perez Zerpa, R. Ros, and M. Schoenauer, “Experimental comparisons of derivative free optimization algorithms,” in *Proc. 8th Int. Symp. Exp. Algorithms*, ser. LNCS, vol. 5526. Springer, 2009, pp. 3–15.
45. P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, “Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization,” Nanyang Technological University, Singapore, Tech. Rep., 2005.
46. Y.-W. Shang and Y.-H. Qiu, “A note on the extended rosenbrock function,” *Evol. Comput.*, vol. 14, no. 1, pp. 119–126, 2006.
47. R. Salomon, “Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions - a survey of some theoretical and practical aspects of genetic algorithms,” *BioSystems*, vol. 39, pp. 263–278, 1995.
48. N. Higashi and H. Iba, “Particle swarm optimization with gaussian mutation,” in *Proc. IEEE Swarm Intell. Symp.*, 2003, pp. 72–79.
49. R. W. Morrison and K. A. D. Jong, “Measurement of population diversity,” in *Proc. Artif. Evolution*, 2002, pp. 31–41.