A Non-Revisiting Particle Swarm Optimization

Chi Kin Chow and Shiu Yin Yuen

Abstract – **In this article, a non-revisiting particle swarm optimization (NrPSO) is proposed. NrPSO is an integration of the non-revisiting scheme and a standard particle swarm optimization (PSO). It guarantees that all updated positions are not evaluated before. This property leads to two advantages: 1) it undisputedly reduces the computation cost on evaluating a time consuming and expensive objective function and 2) It helps prevent premature convergence. The non-revisiting scheme acts as a self-adaptive mutation. Particles genericly switch between local search and global search. In addition, since the adaptive mutation scheme of NrPSO involves no parameter, comparing with other variants of PSO which involve at least two performance sensitive parameters, the performance of NrPSO is more reliable. The simulation results show that NrPSO outperforms four variants of PSOs on optimizing both uni-modal and multi-modal functions with dimensions up to 40. We also illustrate that the overhead and archive size of NrPSO are insignificant. Thus NrPSO is practical for real world applications. In addition, it is shown that the performance of NrPSO is insensitive to the specific chosen values of parameters.**

I. Introduction

Particle Swarm Optimization (PSO) [1] [2] is an evolutionary computation technique inspired by swarm intelligence phenomena such as birds flocking and fish schooling. For a *D*-dimensional problem, each particle in a swarm in PSO is represented by 1) its position $\mathbf{x} = [x_1, x_2, \ldots, x_D]$, 2) its velocity $\mathbf{v} = [v_1, v_2, \ldots, v_D]$, and 3) its best visited position so far $\mathbf{b} = [b_1, b_2, \ldots, b_D]$. Besides, the swarm also records the historical best position $\mathbf{g}$ found by the whole swarm. During iterations, the i^{th} position component x_i and the i^{th} velocity component v_i of a particle are updated according to the following equations:

$$v_i \leftarrow wv_i + c_1 \times \eta_1 \times (b_i - x_i) + c_2 \times \eta_2 \times (g_i - x_i) \quad (1)$$

$$x_i \leftarrow x_i + v_i \quad (2)$$

where c_1 and c_2 are the relative influence of the cognitive and social components respectively; w is the inertia weight; η_1 and η_2 are uniformly distributed random variables in the range [0,1].

Compared to other evolutionary computation algorithms, such as Genetic algorithm, PSO has shown a faster convergence speed on some problems [3] [4]. In addition, it is easier to implement with less parameters to adjust. However, PSO suffers from premature convergence and stagnate at local optimal solutions. Angeline [5] showed that though PSO may outperform other evolutionary algorithms in the early iterations, its performance may not be competitive as the number of generations is increased. Particles converge to a position while their velocities tend to zero. Thus, no more significant fitness improvement can be made. In order to overcome the premature convergence of PSO, it is important to increase the diversity of particles. Recently, several researchers proposed adding swarm diversity for avoidance of premature convergence, such as solving the collision and clustering between two particles [6], random re-initialization [7], and mutation with small probability [8].

A dissipative particle swarm optimization (DPSO) [7] introduced random mutation that helps particles to escape from local minima. Its formula is described as follows:

$$\text{If } \eta_3 < C_v \text{ then } v_i = \eta_4 \times V_{max} / C_m \quad (3)$$

where η_3 and η_4 are uniformly distributed random variables in the range [0,1]; C_v is the mutation rate to control the velocity; C_m is a constant to control the extent of mutation; and V_{max} is the maximum velocity.

Particle swarm optimization with spatial particle extension (SEPSO) [6] introduced the spatial particle extension model to increase the diversity when particles start to cluster. Two particles collide when their distance is smaller than a given radius r. If collision occurs, the corresponding particles bounce off by adjusting their velocities.

Liu *et al.* in [9] proposed a PSO with mutation (PSOMS) that prevents premature convergence. According to the averaged similarity between each particle and the historical best particle explored by the swarm, the clustering degree of particle swarm is computed to measure the swarm diversity. The position of a particle is re-initialized if:

$$\text{If } \eta_4 < \alpha \times c(t) \times s(i, g) \quad (4)$$

where α is a predefined constant, $c(t)$ is the collectivity of t^{th} generation and $s(i, g)$ denotes the similarity of the i^{th} particle to the current best particle.

Though the modified PSO models [6] [7] [8] weaken the chance of premature convergence, a proper selection of parameters (i.e. C_v and C_m in DPSO, r in SEPSO and α in PSOMS) is critical to the performance. To be a truly adaptive PSO, the particle exploration must be parameter free. In this article, we proposed integrating the non-revisiting scheme suggested by Yuen and Chow [10] with a standard PSO, which results in a non-revisiting PSO (NrPSO). Since all updated positions are guaranteed to be

Chi Kin Chow and Shiu Yin Yuen are with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, China (e-mail: {chowchi, kelviny}@cityu.edu.hk).

978-1-4244-1823-7/08/$25.00©2008 IEEE

novel – they are non-revisited before, faster convergence speed is expected. Moreover, the NrPSO reduces not only computation cost but also the evaluation cost for a large variety of applications [11]-[14]. Furthermore, due to the nature of the non-revisiting scheme, the non-revisited position self switch between local search and global search.

This paper is organized as follows: Section II reports the PSO of which the updated positions are non-revisited. Section III presents the simulation setup. Section IV reports the simulations results. A conclusion is drawn in Section V.

II. NON-REVISITING PSO (NRPSO)

A. Non-Revisiting Scheme

The non-revisiting scheme is proposed by Yuen and Chow in [10]. It is originally applied to genetic algorithm (GA); the non-revisiting GA (NrGA) prevents from solution re-evaluation. In addition, the scheme also acts as a parameter-less mutation operation. The NrGA is found to be more robust than GA.

Definition 1: Revisits
Suppose Q is a set of evaluated solutions, the solution $\mathbf{x}$ is a revisit if $\mathbf{x} \in Q$. □

The non-revisiting scheme stores all visited solutions $\{\mathbf{s_i}\}$ by a tree-structure archive, namely binary space partitioning (BSP) tree. During iterations, the search space is being partitioned into a set of regions H. The regions are non-overlapping, i.e. $h_i \cap h_j = \varnothing$ for all $h_i, h_j \in H$ and $h_i \neq h_j$, and each of them consists of one evaluated solution. In view of the BSP tree, a node represents a region $h \in H$. Thus, the tree grows along with the iterations. Since the tree construction depends on the sequence of solution set, the BSP tree is a random tree and its topology is different from trial to trial.

The scheme is analogous to a black box function (fig. 1). The input $\mathbf{x}$ of that function can be any point in the search space. If $\mathbf{x}$ is a *revisit*, that function outputs solution $\mathbf{r}$ such that 1) $\mathbf{r} \neq \mathbf{x}$, 2) $\mathbf{x}, \mathbf{r} \in h_x \in H$ and 3) $\mathbf{r}$ is the nearest neighbor of $\mathbf{x}$. Otherwise, $\mathbf{r}$ is assigned as $\mathbf{x}$. Since the size of $h \in H$ gradually decreases along with the iterations, and $\mathbf{r}$ is randomly selected from h_x where $\mathbf{x} \in h_x$, the expected distance between $\mathbf{x}$ and $\mathbf{r}$ becomes smaller.

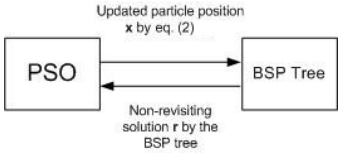

Fig. 1 Communication between PSO and the BSP tree.

B. Particle acceleration by the non-revisiting scheme

The position of a particle $\mathbf{r}$ returned by the non-revisiting scheme can be expressed as:

$$r_i^{(t)} = x_i^{(t)} + \Delta x_i^{(t)} \quad (5)$$

where x_i is the i^{th} position component updated by eq. (2), the superscript (t) represents the t^{th} iteration. Since the adjustment of $\mathbf{x}$: $\Delta\mathbf{x}$ is computed based on the archive A (revisit set) and the updated $\mathbf{x}$, it can be written as a function of $\mathbf{x}$ and A:

$$\Delta x_i^{(t)} = \psi(x_i^{(t)}, A^{(t-1)}) \quad (6)$$

According to eqs. (1) - (2), $\Delta x_i^{(t)}$ can be further expressed as a function $x_i^{(t-1)}$, $v_i^{(t-1)}$, $b_i^{(t-1)}$, $g_i^{(t-1)}$ and $A^{(t-1)}$ (eq. (7)):

$$\Delta x_i^{(t)} = V_r^{(t)} = \psi(x_i^{(t-1)}, v_i^{(t-1)}, b_i^{(t-1)}, g_i^{(t-1)}, A^{(t-1)}) \quad (7)$$

As $b_i^{(t-1)}$ and $g_i^{(t-1)}$ are elements of $A^{(t-1)}$, eq. (7) can be simplified as:

$$\Delta x_i^{(t)} = V_r^{(t)} = \psi(x_i^{(t-1)}, v_i^{(t-1)}, A^{(t-1)}) \quad (8)$$

By substituting eq. (8) into eqs. (1) and (2), it is observed in eq. (9) that the non-revisiting scheme acts as an accelerator to particles. The acceleration is $\psi(.)$:

$$v_i^{(t)} = wv_i^{(t-1)} + c_1 \times \eta_1 \times (b_i^{(t-1)} - x_i^{(t-1)}) + c_2 \times \eta_2 \times (g_i^{(t-1)} - x_i^{(t-1)}) + \psi(x_i^{(t-1)}, v_i^{(t-1)}, A^{(t-1)}) \quad (9)$$

Fig. 2 illustrates an example of particle acceleration by the non-revisiting scheme. The sign • represents the particle position while the grey region represents the positions that have been evaluated. When the updated position $\mathbf{x}^{(t)}$ falls into the grey region, an acceleration V_r is applied to the particle and it escapes from that region.

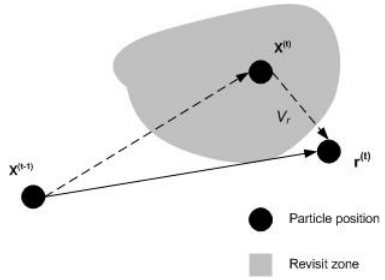

Fig. 2 An example of the particle acceleration by the non-revisiting scheme.

Definition 2: *Nearest neighbor*
Note that PSO must be discretized for non-revisiting scheme to work. In NrPSO, the search space S is in the form of a grid. Every solution occupies a unique cell in S. Solution $\mathbf{n} \in S$ is a nearest neighbor of solution $\mathbf{x} \in S$ if the cell of $\mathbf{n}$ is adjacent to the cell of $\mathbf{x}$. For a D-dimensional function, every solution, except those at the boundary of the search space, has $2D$ nearest neighbors. □

Suppose $Z \subset S$ is a revisit set (revisit zone in S), $B \subset Z$ is a

subset of Z such that every $\mathbf{b} \in \mathbf{B}$ must has at least one *nearest neighbor* $\mathbf{n}$ which is not a revisit, i.e. $\mathbf{n} \notin Z$. According to the non-revisiting scheme, the distance between $\mathbf{x}$ and $\mathbf{r}$ must be larger than $|\mathbf{x} - \mathbf{b_0}|$ where

$$\mathbf{b_0} = \arg\min_{\mathbf{b} \in B} |\mathbf{x} - \mathbf{b}| \quad (10)$$

Thus the magnitude of V_r depends upon the size of the revisit zone Z that $\mathbf{x}$ belongs to, i.e. $\mathbf{x} \in R$. Since the size of Z is related to the size of optimal basin, NrPSO adaptively switches between escaping from local optima and fine searching solutions. Moreover, while the swarm converges to a position $\mathbf{x_0}$ (premature convergence), the region around $\mathbf{x_0}$ forms a revisit zone that accelerates particles to guide the swarm to move away from $\mathbf{x_0}$.

C. Redirected Particle Position

Considering the case that particle $\mathbf{p}$ is moving towards a visited position $\mathbf{s}$, as discussed in the previous section, the non-revisiting scheme forces the particle to a new position $\mathbf{r}$ where 1) $\mathbf{r}$ is never visited before and 2) $\mathbf{r}$ is a nearby position of $\mathbf{s}$. It is analogous to view $\mathbf{s}$ as an *obstacle* in the search space. These obstacles (revisit set) applies a reaction force $\Delta\mathbf{x}$ to $\mathbf{p}$ such that the resultant position of $\mathbf{p}$ is $\mathbf{r} = \mathbf{x} + \Delta\mathbf{x}$. Since the particle receives reaction force only when it collides with the obstacle, the force $\Delta\mathbf{x}$ should not be included in the velocity updating equation of the NrPSO. In addition, $\Delta\mathbf{x}$ is a random variable viewed from the particle, the equation is normally the same as that of a standard PSO (eq. (1)).

D. Mechanism of NrPSO

In general, a standard PSO (SPSO) and its variants (i.e. DPSO, SEPSO, PSOMS) can be easily modified as NrPSO by including the black box function (fig. 1) since the non-revisiting scheme is independent of the operations in PSO. In this article, a standard PSO is used and modified as NrPSO. Fig. 3 shows the pseudo code of NrPSO. The procedure of NrPSO is similar to those of standard PSO except for the following three extra steps in each iteration:

1. After updating $\mathbf{x_i}$ and $\mathbf{v_i}$, the redirected position $\mathbf{r_i}$ of $\mathbf{x_i}$ is computed by the non-revisiting scheme.
2. When $\mathbf{r_i}$ is evaluated, it is regarded as a revisit. Thus, the BSP tree should be updated in order to include $\mathbf{r_i}$ into the revisit pool.
3. Instead of $\mathbf{x_i}$, $\mathbf{r_i}$ is used to update $\mathbf{b_i}$ and $\mathbf{g}$.

III. Simulation Setup

A. Test function set

In this article, a real valued function set $\mathbf{F} = [f_1(\mathbf{x}), f_2(\mathbf{x}),\ldots, f_{14}(\mathbf{x})]$ consisting of 14 functions are employed to illustrate the performance of the proposed non-revisiting scheme. The details (i.e. X – search space, D – function dimension, $\mathbf{x_0}$ – optimal solution, y_0 – optimal fitness) of the functions are listed in Table II. The first six functions are simple unimodal functions whereas the remaining eight functions are multimodal functions designed with a considerable amount of local minima. Meanwhile, the dimensions of the first ten functions are adjustable while the dimensions of the remaining four functions are fixed at two. All functions with the exceptions of f_9, f_{11}, f_{12} and f_{13}, have the global minimum at the origin or very close to the origin. Simulations were carried out to find the global minimum of each function.

B. Setup of test algorithms

To evaluate the impact of NrPSO, we compare its optimal fitness with those found by SPSO, DPSO, SEPSO and PSOMS. For all test algorithms, the values of c_1, c_2 are set to 0.8. The inertia w is linearly decreasing from 1 to 0. Suppose $X = \prod_{i=1,\ldots,D}[L_i, U_i]$ is the search space of a D-dimensional objective function, the maximum velocity V_{max} is set to $0.1R$ where $R = max(U_i - L_i)$. For NrPSO, the axis resolutions d of the first 10 functions are chosen as 100 whereas the values of d are 2,000 for the remaining four. The parameters used in DPSO, SEPSO and PSOMS are assigned as those suggested their original works: the parameters C_v and C_m of DPSO are chosen to be 0.001 and 0.002 respectively. For SEPSO, the radius r is assigned to $0.005RD^{0.5}$ and simple velocity line bouncing with bouncing factor -1 is used. For PSOMS, the parameters d_{min}, d_{max}, β and α are set to $0.001RD^{0.5}$, $0.01RD^{0.5}$, 1 and 3 respectively.

C. Simulation settings

For all simulations, the swarm size is set to 200 and the search is terminated after 200 generations. All test functions with the exceptions of f_{11}, f_{12}, f_{13} and f_{14}, which is two-dimensional, are tested with dimension 40. Since the test algorithms are stochastic, their performance on each test function are concluded by 100 independent runs. All simulations are performed on a PC with 3.2GHz CPU and 1GB memory. The algorithms are implemented in C language.

- Initialize particle positions $\{\mathbf{x_i}\}$
- Initialize iteration index t as zero.
- Evaluate $\{\mathbf{x_i}\} \rightarrow \{f(\mathbf{x_i})\}$

while $t < t_{max}$ and $f(\mathbf{g}) > f_{min}$

for $i = 1$ *to* N

- Update $\mathbf{v_i}$ for all i according to the eq. (1)
- Update $\mathbf{x_i}$ for all i according to the eq. (2)
- Compute the redirected position $\mathbf{r_i}$.
- Evaluate: $\mathbf{r_i} \rightarrow f(\mathbf{r_i})$
- Record $[\mathbf{r_i}, f(\mathbf{r_i})]$ in the BSP tree
- *if* $f(\mathbf{r_i}) < f(\mathbf{b_i})$ *then* $\mathbf{b_i} \leftarrow \mathbf{r_i}$

Next i

- *if* $f(\mathbf{o}) < f(\mathbf{g})$ *then* $\mathbf{g} \leftarrow \mathbf{o}$ where $\mathbf{o} = \arg\min_{\mathbf{s} \in \{\mathbf{b_i}\} \cup f(\mathbf{s})}$

- $t \leftarrow t + 1$

loop

Fig. 3 The pseudo code for NrPSO.

IV. Simulation Results

Initially, we observe the performance in terms of accuracy (quality of the averaged optimal fitness) for 100 trials of NrPSO in comparison with SPSO, DPSO, SEPSO and PSOMS. The contribution of integrating the non-revisiting scheme is observed. To be a practical solution to real world problems, the processing time and the archive size of NrPSO should be within a reasonable range. The average processing time and the worst case archive size are also observed. Finally, the stability of the test algorithms is observed. The averaged and the standard deviation (inside brackets) of the optimal fitness for 100 trials are presented in Table III. In Table IV, the averaged processing time of the algorithms are presented. Table V lists the averaged archive sizes for 100 trials. Table VI lists the stability data of the algorithms.

A. Accuracy

Table III lists the average and the standard deviation of the optimal fitness for 100 trials. The algorithm with bold average fitness value represented that it performs the best among all algorithms. It is clear from the table that NrPSO performs better than other PSO models for all unimodal functions except f_4, which shows the superiority of the neighbor picking scheme of NrPSO. Moreover, NrPSO has improved the accuracy significantly when compared with PSO models on optimizing multimodal functions: $f_7 - f_{14}$. The reason is that the non-revisiting scheme memorizes all visited local optima. This substantially helps to escape from the local basins. In general, a consistent performance of the NrPSO has been observed for all benchmarks considered in this investigation.

B. Processing time

During the search process, an algorithm spends its computation on either solution generation or function evaluation. Different algorithms use different strategies to guess a solution, so the corresponding processing time is different. For SEPSO and PSOMS, the mutual distance among the swarm has to be computed in each iteration. The NrPSO spends most computations on accessing the BSP tree (i.e. search, insert and prune nodes). As a practical solution to real world applications, the processing time of NrPSO should be within an acceptable range. In this section, the computation load of an algorithm, in term of processing time, is studied.

Table IV lists the processing time amongst the test algorithms. Seen from the table, NrPSO is faster than SEPSO for all simulations. Though the maximum overhead of NrPSO compared to SPSO, DPSO and SEPSO is 1.5 seconds, its improvement is significant. Furthermore, it should be emphasized that, for real world applications such as surface registration [11], optimized design and energy management of heating, ventilating and air conditioning systems [12]-[14], function evaluations are much more time consuming than solution generation. For these classes of functions, the overhead rate of NrPSO is insignificant.

C. Archive size

The variations of the archive size of the BSP tree for 100 trials are observed and the results are listed in Table V. The archive size is presented as the number of the BSP tree nodes. In general, the archive size increases along with the function dimension. One reason is that the increment on function dimension leads to a significant growth on the size of search space, which reduces the chance of tree node pruning. Apart from the increment of function dimension, the nature of an objective function also affects the archive size. The fitness landscape of the Schwefel's problem 2.21 composes of piecewisely flat regions. These regions mislead the solution replacement strategy and NrPSO more likely acts as a random non-revisiting search. Thus, the chance of tree pruning is reduced. For the Quaric function, the random element introduces many small oscillations (local optimum) into the fitness landscape. In addition, these local optima can hardly be merged to reduce the archive size as they are sparsely distributed on the landscape. The upper bound of the archive size in this test is 40,000. By comparing using a PC that is commonly configured with 1GMB memory, these 40,000 units are just 0.096% of the memory (assuming that each BSP tree node is represented by 24 bytes).

D. Stability of NrPSO

In this article, the stability of an algorithm is defined as the influence of changing parameter values to algorithms accuracy. It is in terms of quality of the averages and the standard deviation of the optimal fitness using different parameter values. Table I lists the varied parameters of the algorithms and the corresponding ranges. In this experiment, the axis resolution d of NrPSO is varied from 60 to 100. C_v in DPSO is varied from $0.0005R$ to $0.002R$. r in SEPSO is varied from $0.0005R$ to $0.002R$. α in PSOMS is varied from 1 to 10. The remaining parameters of the test algorithms are maintained as the same in the accuracy test.

TABLE I
THE ADJUSTED PARAMETERS OF THE ALGORITHMS AND THE CORRESPONDING RANGES IN STABILIT TEST

	parameter	range
NrPSO	d	f_1-f_{10}: [60, 100] and f_{11}-f_{14}: [1800, 2200]
DPSO	C_v	[0.0005, 0.002]
SEPSO	r	[$0.005R$, $0.02R$]
PSOMS	α	[1, 10]

Table VI lists the result for 100 trials. Seen from the table, the results indicate that the effect of d on the optimal fitness is insignificant for all test functions. On the other hand, the choice of C_v, r, and α are critical factors for those functions. In conclusion, it is clear from the results that the averaged optimal fitness is only slightly dependent on the axis

resolution for all the test functions. Therefore, a proper selection of the axis resolution is not a key factor for most problems (at least for the 14 benchmark functions), and the use of the proposed scheme can be identified as a good strategy to overcome the difficulties of selecting a proper mutation parameter.

V. Conclusions

Premature convergence or lack of diversity is a major problem of particle swarm optimization (PSO). To tackle it, a variety of modifications on PSO including collision and clustering between particles, random re-initialization and mutation with small probability are suggested. In this article, a non-revisiting particle swarm optimization (NrPSO) is proposed. NrPSO is an integration of the non-revisiting scheme [10] and a standard PSO. It guarantees that all updated positions are *novel* – they have not been evaluated before. This property leads to two advantages: 1) it obviously reduces the computation cost on evaluating time-consuming objective functions such as surface registration, optimized design and energy management of heating, ventilating and air conditioning systems; 2) It helps prevent premature convergence; 3) The nature of the non-revisiting scheme acts as a self-adaptive mutation. It provides a generic guidance to particles to search either locally or globally. 4) In addition, since the adaptive mutation scheme of NrPSO involves no parameter, comparing with other PSO models involving at least two performance sensitive parameters, the performance of NrPSO is more reliable. The simulation results show that NrPSO outperforms four PSO models on optimizing both uni-modal and multi-modal functions with dimensions up to 40. 5) We also illustrate that NrPSO is practical for real world applications as its overhead and archive size are insignificant. 6) In addition, it is shown that the performance of NrPSO is insensitive to the grid resolution parameter of the problem.

Acknowledgment

The work described in this article was supported by a grant from CityU (7001859).

References

[1] R. Eberhart, J. Kennedy. "A new optimizer using particle swarm theory," *Proc. Int. Symposium on Micro Machine and Human Science*, 1995, pp. 39 - 43.

[2] R. Eberhart, J. Kennedy, "Particle Swarm Optimization," in *Proc. IEEE Int. Conf. On Neural Networks*, 1995, pp. 1942 - 1948.

[3] Y. Shi, R. Eberhart. "A modified particle swarm optimizer," in *Proc. of IEEE Int. Conf. on Evolutionary Computation*, 1988, pp. 69 - 73.

[4] R. Eberhart, Y. Shi, "Particle swarm optimization: developments, applications and resources," *Proc. IEEE Int. Conf. on Evolutionary Computation,* 2001, pp. 81 – 86.

[5] P. Angeline, "Evolutionary optimization versus particle swarm optimization: philosophy and performance difference," in *Proc. of the Evolutionary Programming Conference,* 1998, pp. 169 - 173.

[6] T. Krink, J. S. Vesterstrom, J. Riget, "Particle swarm optimization with spatial particle extension," in *Proc. IEEE Int. Conf. on Evolutionary Computation,* 2002, pp. 1474 - 1497.

[7] X. F. Xie, W. J. Zhang, Z. L. Yang, "A dissipative particle swarm optimization," in *Proc. IEEE Int. Conf. on Evolutionary Computation,* 2002, pp. 1666 - 1670.

[8] A. Ratnaweera, S. K. Halgamuge, H. C. Watson, "Self-Organizing hierarchical particle swarm optimizer with timevarying acceleration coefficients," *IEEE Trans. on Evolutionary Computation,* 2004, vol. 8, no. 3, pp. 240 - 255.

[9] J. Liu, X. Fan and Z. Qu, "An Improved Particle Swarm Optimization with Mutation Based on Similarity," *Proc. IEEE Int. Conf. on Natural Computation*, 2007, pp. 824 – 828.

[10] S. Y. Yuen and C. K. Chow, "A Non-revisiting Genetic Algorithm," in *Proc. of IEEE CEC Conf.*, pp. 4583 – 4590, 2007.

[11] C. K. Chow, H. T. Tsui and T. Lee, "Surface registration using a dynamic genetic algorithm," *Pattern Recognition*, vol. 37, no. 1, pp. 105-117, 2004.

[12] K. F. Fong, V. I. Hanby and T. T. Chow, "HVAC system optimization for energy management by evolutionary programming," *Energy and Buildings*, vol. 38, pp. 220- 231, 2006.

[13] K. F. Fong, T. T. Chow and V. I. Hanby, "Development of optimal design of solar water heating system by using evolutionary algorithm," *Journal of Solar Energy Engineering*, vol. 129, no. 4, pp. 499-501, 2007.

[14] K. F. Fong, "Optimized design and energy management of heating, ventilating and air conditioning systems by evolutionary algorithm," PhD thesis, De Montfort University, UK, 2006.

TABLE II
THE DETAILS OF THE FOURTEEN TEST FUNCTIONS

		X	D	$\mathbf{x_0}$	y_0
f_1: Spherical model	$\sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	40	[0,…,0]	0
f_2: Schwefel's problem 2.22	$\sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	40	[0,…,0]	0
f_3: Schwefel's problem 1.2	$\sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^D$	40	[0,…,0]	0
f_4: Schwefel's problem 2.21	$\max_{i \in [1,D]} \lvert x_i \rvert$	$[-100, 100]^D$	40	[0,…,0]	0
f_5: Rosenbrock's function	$\sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-29, 31]^D$	40	[0,…,0]	0
f_6: Quaric function	$\sum_{i=1}^{D} (ix^4 + random[0,1])$	$[-1.28, 1.25]^D$	40	[0,…,0]	0
f_7: Rastrigin's function	$\sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$[-5.12, 5.12]^D$	40	[0,…,0]	0

f_8: Griewank function	$\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\frac{x_i}{\sqrt{i}} + 1$	$[-600, 600]^D$	40	$[0,\ldots,0]$	0
f_9: Schwefel's problem 2.26	$-\sum_{i=1}^{D} x_i \sin\sqrt{\lvert x_i \rvert}$	$[-500, 500]^D$	40	[420.9687, 420.9687]	-418.9829D
f_{10}: Ackley	$-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$	$[-32, 32]^D$	40	$[0,\ldots,0]$	0
f_{11}: Shekel's Foxholes	$\left[\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{i,j})^6}\right]^{-1}$ $\{a_{i,j}\} = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \cdots & 32 & 32 & 32 \end{bmatrix}$	[-98,34]	2	[-32,…,-32]	1
f_{12}: Six-Hump Camel-Back	$4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	[-4.91017, 5.0893] × [-5.7126, 4.2874]	2	[0.08983,-0.7126], [-0.08983,0.7126]	-1.0316285
f_{13}: Branin	$(x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	[-8.142, 6.858] × [-12.275, 2.725]	2	[-3.142,12.275], [3.142,2.275], [9.425,2.425]	0.398
f_{14}: Goldstein-Price	g(**x**) × h(**x**) *where* $g(\mathbf{x}) = 1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)$ $h(\mathbf{x}) = 30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)$	[-2,2] ×[-3,1]	2	[0,-1]	3

TABLE III
THE AVERAGED BEST FITNESS FOUND BY NRPSO, SPSO, DPSO, SEPSO AND PSOMS

function		mean	stdev.	function		mean	stdev.	function		mean	stdev.
	NrPSO	**4.700**	2.478		NrPSO	**14.335**	0.993		NrPSO	**0.998**	0.006
	SPSO	230.148	7.708		SPSO	15.014	0.881		SPSO	**0.998**	0.006
f_1	DPSO	220.830	5.907	f_6	DPSO	14.688	0.823	f_{11}	DPSO	**0.998**	0.006
	SEPSO	216.760	7.223		SEPSO	15.217	0.992		SEPSO	**0.998**	0.006
	PSOMS	246.545	8.046		PSOMS	15.326	0.939		PSOMS	1.031	0.422
	NrPSO	**0.025**	0.315		NrPSO	**67.496**	4.181		NrPSO	**-1.032**	0.006
	SPSO	60.102	8.705		SPSO	198.674	4.939		SPSO	**-1.032**	0.008
f_2	DPSO	33.283	7.222	f_7	DPSO	230.250	5.017	f_{12}	DPSO	**-1.032**	0.012
	SEPSO	71.803	9.018		SEPSO	207.040	5.069		SEPSO	**-1.032**	0.009
	PSOMS	31.884	6.785		PSOMS	211.737	4.931		PSOMS	**-1.032**	0.010
	NrPSO	**2407.500**	29.134		NrPSO	**0.887**	0.708		NrPSO	**0.398**	0.006
	SPSO	5643.846	37.322		SPSO	9.920	1.602		SPSO	**0.398**	0.016
f_3	DPSO	4279.550	34.570	f_8	DPSO	8.842	1.451	f_{13}	DPSO	**0.399**	0.040
	SEPSO	5393.726	36.798		SEPSO	9.590	1.438		SEPSO	**0.398**	0.011
	PSOMS	5700.470	37.300		PSOMS	10.550	1.779		PSOMS	**0.398**	0.018
	NrPSO	24.750	1.901		NrPSO	**-9276.4**	32.357		NrPSO	**3.000**	0.006
	SPSO	18.782	1.639		SPSO	-7218.9	29.202		SPSO	**3.000**	0.020
f_4	DPSO	**13.645**	1.330	f_9	DPSO	-4614.0	26.126	f_{14}	DPSO	3.001	0.036
	SEPSO	18.407	1.874		SEPSO	-7128.1	29.451		SEPSO	3.001	0.030
	PSOMS	18.656	1.739		PSOMS	-7203.1	26.464		PSOMS	**3.000**	0.021
	NrPSO	**1571.67**	31.581		NrPSO	**0.048**	1.519				
	SPSO	51273.57	144.731		SPSO	9.144	0.840				
f_5	DPSO	46195.23	118.383	f_{10}	DPSO	7.886	0.737				
	SEPSO	60907.34	169.193		SEPSO	8.917	0.842				
	PSOMS	68209.62	176.291		PSOMS	9.322	0.969				

TABLE IV
THE AVERAGED PROCESSING TIME (IN SEC.) OF NRPSO, SPSO, DPSO, SEPSO AND PSOMS

function		mean	stdev.	function		mean	stdev.	function		mean	stdev.
f_1	NrPSO	1.733	0.165	f_6	NrPSO	1.767	0.150	f_{11}	NrPSO	0.267	0.165

	SPSO	0.200	< 0.001		SPSO	0.200	< 0.001		SPSO	0.200	< 0.001
	DPSO	0.200	< 0.001		DPSO	0.233	< 0.001		DPSO	0.233	< 0.001
	SEPSO	7.900	< 0.001		SEPSO	8.000	< 0.001		SEPSO	0.600	< 0.001
	PSOMS	0.367	< 0.001		PSOMS	0.400	< 0.001		PSOMS	0.200	< 0.001
	NrPSO	1.667	0.187		NrPSO	1.900	0.148		NrPSO	0.100	0.148
	SPSO	0.033	< 0.001		SPSO	0.233	< 0.001		SPSO	0.033	< 0.001
f_2	DPSO	0.067	< 0.001	f_7	DPSO	0.267	< 0.001	f_{12}	DPSO	0.033	< 0.001
	SEPSO	7.733	< 0.001		SEPSO	8.000	< 0.001		SEPSO	0.400	< 0.001
	PSOMS	0.200	< 0.001		PSOMS	0.400	< 0.001		PSOMS	0.033	< 0.001
	NrPSO	2.000	0.006		NrPSO	1.767	0.150		NrPSO	0.067	0.004
	SPSO	0.433	< 0.001		SPSO	0.267	< 0.001		SPSO	< 0.001	< 0.001
f_3	DPSO	0.433	< 0.001	f_8	DPSO	0.300	< 0.001	f_{13}	DPSO	0.033	< 0.001
	SEPSO	8.200	< 0.001		SEPSO	8.000	< 0.001		SEPSO	0.400	< 0.001
	PSOMS	0.600	< 0.001		PSOMS	0.400	< 0.001		PSOMS	0.000	< 0.001
	NrPSO	1.667	0.187		NrPSO	1.667	0.187		NrPSO	0.100	0.028
	SPSO	0.033	< 0.001		SPSO	0.100	< 0.001		SPSO	0.033	< 0.001
f_4	DPSO	0.067	< 0.001	f_9	DPSO	0.100	< 0.001	f_{14}	DPSO	0.033	< 0.001
	SEPSO	7.833	< 0.001		SEPSO	7.867	< 0.001		SEPSO	0.400	< 0.001
	PSOMS	0.200	< 0.001		PSOMS	0.233	< 0.001		PSOMS	0.067	< 0.001
	NrPSO	2.067	0.199		NrPSO	1.700	0.177				
	SPSO	0.500	< 0.001		SPSO	0.267	< 0.001				
f_5	DPSO	0.567	< 0.001	f_{10}	DPSO	0.267	< 0.001				
	SEPSO	8.367	< 0.001		SEPSO	8.000	< 0.001				
	PSOMS	0.733	< 0.001		PSOMS	0.433	< 0.001				

TABLE V
THE AVERAGED ARCHIVE SIZE OF NRPSO

function	mean	stdev.	function	mean	stdev.	function	mean	stdev.	function	mean	stdev.
f_1	38760	877	f_5	39943	239	f_9	39422	661	f_{13}	5976	125
f_2	37435	1144	f_6	40000	< 0.1	f_{10}	38838	700	f_{14}	5535	154
f_3	40000	< 0.1	f_7	39304	613	f_{12}	8860	1521			
f_4	40000	< 0.1	f_8	39778	366	f_{12}	6160	1071			

TABLE VI
THE STABILIES THE NRPSO, DPSO, SEPSO AND PSOMS

function		mean	stdev.	function		mean	stdev.	function		mean	stdev.
	NrPSO	5.31	0.67		NrPSO	15.18	0.97		NrPSO	1.16	0.19
f_1	DPSO	346.89	132.55	f_6	DPSO	23.26	9.47	f_{11}	DPSO	1.26	0.28
	SEPSO	264.38	52.30		SEPSO	20.07	5.59		SEPSO	1.27	0.27
	PSOMS	329.85	87.87		PSOMS	19.86	4.90		PSOMS	1.41	0.39
	NrPSO	0.03	0.01		NrPSO	86.16	22.14		NrPSO	-1.28	0.28
f_2	DPSO	48.22	17.85	f_7	DPSO	297.40	78.80	f_{12}	DPSO	-1.55	0.60
	SEPSO	107.40	40.36		SEPSO	252.84	50.22		SEPSO	-1.38	0.40
	PSOMS	43.79	13.87		PSOMS	317.13	109.60		PSOMS	-1.57	0.60
	NrPSO	2483.24	75.91		NrPSO	1.13	0.28		NrPSO	0.48	0.10
f_3	DPSO	5889.88	1617.37	f_8	DPSO	12.59	4.02	f_{13}	DPSO	0.51	0.13
	SEPSO	7695.46	2415.21		SEPSO	12.66	3.10		SEPSO	0.58	0.18
	PSOMS	7486.99	1881.23		PSOMS	13.14	2.66		PSOMS	0.61	0.23
	NrPSO	28.95	4.79		NrPSO	-11090.3	2142.12		NrPSO	3.45	0.48
f_4	DPSO	17.86	4.73	f_9	DPSO	-7059.1	2704.05	f_{14}	DPSO	3.84	0.94
	SEPSO	22.92	5.31		SEPSO	-11400.5	4707.87		SEPSO	4.77	1.92
	PSOMS	27.99	10.54		PSOMS	-11361.3	4230.39		PSOMS	4.53	1.66
	NrPSO	1808.39	265.24		NrPSO	0.05	0.00				
f_5	DPSO	63934.25	20942.20	f_{10}	DPSO	10.98	3.41				
	SEPSO	94438.08	38544.21		SEPSO	10.81	1.93				
	PSOMS	99434.88	34894.23		PSOMS	12.34	3.44				