Adaptive Particle Swarm Optimization

Zhi-Hui Zhan, *Student Member, IEEE*, Jun Zhang, *Senior Member, IEEE*, Yun Li, *Member, IEEE*, and Henry Shu-Hung Chung, *Senior Member, IEEE*

***Abstract*—An adaptive particle swarm optimization (APSO) that features better search efficiency than classical particle swarm optimization (PSO) is presented. More importantly, it can perform a global search over the entire search space with faster convergence speed. The APSO consists of two main steps. First, by evaluating the population distribution and particle fitness, a real-time evolutionary state estimation procedure is performed to identify one of the following four defined evolutionary states, including *exploration*, *exploitation*, *convergence*, and *jumping out* in each generation. It enables the automatic control of inertia weight, acceleration coefficients, and other algorithmic parameters at run time to improve the search efficiency and convergence speed. Then, an elitist learning strategy is performed when the evolutionary state is classified as *convergence* state. The strategy will act on the globally best particle to jump out of the likely local optima. The APSO has comprehensively been evaluated on 12 unimodal and multimodal benchmark functions. The effects of parameter adaptation and elitist learning will be studied. Results show that APSO substantially enhances the performance of the PSO paradigm in terms of convergence speed, global optimality, solution accuracy, and algorithm reliability. As APSO introduces two new parameters to the PSO paradigm only, it does not introduce an additional design or implementation complexity.**

***Index Terms*—Adaptive particle swarm optimization (APSO), evolutionary computation, global optimization, particle swarm optimization (PSO).**

I. Introduction

PARTICLE swarm optimization (PSO), which was introduced by Kennedy and Eberhart in 1995 [1], [2], is one of the most important swarm intelligence paradigms [3]. The PSO uses a simple mechanism that mimics swarm behavior in birds flocking and fish schooling to guide the particles to search for globally optimal solutions. As PSO is easy to implement, it has rapidly progressed in recent years and with many successful applications seen in solving real-world optimization problems [4]–[10].

Manuscript received July 31, 2008; revised November 7, 2008 and January 21, 2009. First published April 7, 2009; current version published November 18, 2009. This work was supported in part by the National Science Foundation (NSF) of China under Project 60573066, the NSF of Guangdong under Project 5003346, the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, P.R. China, and the NSFC Joint Fund with Guangdong under Key Project U0835002. This paper was recommended by Associate Editor Q. Zhang.

Z.-H. Zhan and J. Zhang are with the Department of Computer Science, Sun Yat-Sen University, Guangzhou 510275, China (e-mail: junzhang@ieee.org).

Y. Li is with the Department of Electronics and Electrical Engineering, University of Glasgow, G12 8LT Glasgow, U.K., and also with the University of Electronic Science and Technology of China (UESTC), Chengdu 610054, China.

H. S.-H. Chung is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong.

Digital Object Identifier 10.1109/TSMCB.2009.2015956

However, similar to other evolutionary computation algorithms, the PSO is also a population-based iterative algorithm. Hence, the algorithm can computationally be inefficient as measured by the number of function evaluations (FEs) required [11]. Further, the standard PSO algorithm can easily get trapped in the local optima when solving complex multimodal problems [12]. These weaknesses have restricted wider applications of the PSO [5].

Therefore, accelerating convergence speed and avoiding the local optima have become the two most important and appealing goals in PSO research. A number of variant PSO algorithms have, hence, been proposed to achieve these two goals [8], [9], [11], [12]. In this development, control of algorithm parameters and combination with auxiliary search operators have become two of the three most salient and promising approaches (the other being improving the topological structure) [10]. However, so far, it is seen to be difficult to simultaneously achieve both goals. For example, the comprehensive-learning PSO (CLPSO) in [12] focuses on avoiding the local optima, but brings in a slower convergence as a result.

To achieve both goals, adaptive PSO (APSO) is formulated in this paper by developing a systematic parameter adaptation scheme and an elitist learning strategy (ELS). To enable adaptation, an evolutionary state estimation (ESE) technique is first devised. Hence, adaptive parameter control strategies can be developed based on the identified evolutionary state and by making use of existing research results on inertia weight [13]–[16] and acceleration coefficients [17]–[20].

The time-varying controlling strategies proposed for the PSO parameters so far are based on the generation number in the PSO iterations using either linear [13], [18] or nonlinear [15] rules. Some strategies adjust the parameters with a fuzzy system using fitness feedback [16], [17]. Some use a self-adaptive method by encoding the parameters into the particles and optimizing them together with the position during run time [19], [20]. Although these generation-number-based strategies have improved the algorithm, they may run into the risk of inappropriately adjusting the parameters because no information on the evolutionary state that reflects the population and fitness diversity is identified or utilized. To improve efficiency and to accelerate the search process, it is vital to determine the evolutionary state and the best values for the parameters.

To avoid possible local optima in the convergence state, combinations with auxiliary techniques have been developed elsewhere by introducing operators such as selection [21], crossover [22], mutation [23], local search [24], reset [25], [26], reinitialization [27], [28], etc., into PSO. These hybrid operations are usually implemented in every generation [21]–[23] or at a prefixed interval [24] or are controlled by adaptive

1083-4419/$25.00 © 2009 IEEE

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

strategies using stagnated generations as a trigger [25]–[28]. While these methods have brought improvements in PSO, the performance may further be enhanced if the auxiliary operations are adaptively performed with a systematic treatment according to the evolutionary state. For example, the mutation, reset, and reinitialization operations can be more pertinent when the algorithm has converged to a local optimum rather than when it is exploring.

Extending from the existing parameter setting techniques on inertia weight [13]–[16] and acceleration coefficients [17]–[20], this paper develops a systematic adaptation scheme. The PSO parameters are not only controlled by ESE but also taking the different effects of these parameters in different states into account. In addition, departing from mutation [23], reset [25], [26], or reinitialization [27], [28] operations, the ELS is proposed in this paper to perform only on the globally best particle and only in a convergence state. This is not only because the convergence state needs the ELS most but also because of a very low computational overhead. Further, the adaptive ELS will maintain population diversity for jumping out of the potential local optima. Moreover, tests are to be carried out on various topological structures in the PSO paradigm to verify the effectiveness of the APSO and to more comprehensively compare with other improved PSO algorithms.

In Section II, the PSO and its developments are briefly reviewed. Section III presents the ESE approach in detail. The APSO algorithm is proposed in Section IV through the developments of an adaptive parameter control strategy and the ELS. Section V experimentally compares the APSO with various existing PSO algorithms using a set of benchmark functions. Discussions and further investigations on the APSO are made in Section VI. Finally, conclusions are drawn in Section VII.

II. PSO and Its Developments

A. PSO Framework

In PSO, a swarm of particles are represented as potential solutions, and each particle i is associated with two vectors, i.e., the velocity vector $\boldsymbol{V}_i = [v_i^1, v_i^2, \ldots, v_i^D]$ and the position vector $\boldsymbol{X}_i = [x_i^1, x_i^2, \ldots, x_i^D]$, where D stands for the dimensions of the solution space. The velocity and the position of each particle are initialized by random vectors within the corresponding ranges. During the evolutionary process, the velocity and position of particle i on dimension d are updated as

$$v_i^d = \omega v_i^d + c_1 rand_1^d \left(pBest_i^d - x_i^d\right) + c_2 rand_2^d \left(nBest^d - x_i^d\right) \quad (1)$$

$$x_i^d = x_i^d + v_i^d \quad (2)$$

where ω is the inertia weight [13], c_1 and c_2 are the acceleration coefficients [2], and $rand_1^d$ and $rand_2^d$ are two uniformly distributed random numbers independently generated within [0, 1] for the dth dimension [1]. In (1), $pBest_i$ is the position with the best fitness found so far for the ith particle, and $nBest$ is the best position in the neighborhood. In the literature, instead of using $nBest$, $gBest$ may be used in the global-version PSO, whereas $lBest$ may be used in the local-version PSO (LPSO).

A user-specified parameter $V_{\max}^d \in \Re^+$ is applied to clamp the maximum velocity of each particle on the dth dimension. Thus, if the magnitude of the updated velocity $|v_i^d|$ exceeds $V_{\max}^d$, then v_i^d is assigned the value $sign(v_i^d)V_{\max}^d$. In this paper, the maximum velocity $V_{\max}$ is set to 20% of the search range, as proposed in [4].

B. Current Developments of the PSO

Given its simple concept and effectiveness, the PSO has become a popular optimizer and has widely been applied in practical problem solving. Thus, theoretical studies and performance improvements of the algorithm have become important and attractive. Convergence analysis and stability studies have been reported by Clerc and Kennedy [29], Trelea [30], Yasuda *et al.* [31], Kadirkamanathan *et al.* [32], and van den Bergh and Engelbrecht [33]. Meanwhile, much research on performance improvements has been reported, including parameter studies, combination with auxiliary operations, and topological structures [4], [5], [10].

The inertia weight ω in (1) was introduced by Shi and Eberhart [13]. They proposed an ω linearly decreasing with the iterative generations as

$$\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min})\frac{g}{G} \quad (3)$$

where g is the generation index representing the current number of evolutionary generations, and G is a predefined maximum number of generations. Here, the maximal and minimal weights $\omega_{\max}$ and $\omega_{\min}$ are usually set to 0.9 and 0.4, respectively [13], [14].

In addition, a fuzzy adaptive ω was proposed in [16], and a random version setting ω to $0.5 + random(0,1)/2$ was experimented in [34] for dynamic system optimization. As this random ω has an expectation of 0.75, it has a similar idea as Clerc's constriction factor [28], [29]. The constriction factor has been introduced into PSO for analyzing the convergence behavior, i.e., by modifying (1) to

$$v_i^d = \chi\left[v_i^d + c_1 rand_1^d \left(pBest_i^d - x_i^d\right) + c_2 rand_2^d \left(nBest^d - x_i^d\right)\right] \quad (4)$$

where the constriction factor

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|} \quad (5a)$$

is set to 0.729 with

$$\varphi = c_1 + c_2 = 4.1 \quad (5b)$$

where c_1 and c_2 are both set to 2.05 [29]. Mathematically, the constriction factor is equivalent to the inertia weight, as Eberhart and Shi pointed out in [35]. In this paper, we focus on the PSO with an inertia weight and use a global version of PSO (GPSO) [13] to denote the traditional global-version PSO with an inertia weight as given by (3).

In addition to the inertia weight and the constriction factor, the acceleration coefficients c_1 and c_2 are also important

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

parameters in PSO. In Kennedy's two extreme cases [36], i.e., the "social-only" model and the "cognitive-only" model, experiments have shown that both acceleration coefficients are essential to the success of PSO. Kennedy and Eberhart [1] suggested a fixed value of 2.0, and this configuration has been adopted by many other researchers. Suganthan [37] showed that using ad hoc values of c_1 and c_2 rather than a fixed value of 2.0 for different problems could yield better performance. Ratnaweera *et al.* [18] proposed a PSO algorithm with linearly time-varying acceleration coefficients (HPSO-TVAC), where a larger c_1 and a smaller c_2 were set at the beginning and were gradually reversed during the search. Among these three methods, the HPSO-TVAC shows the best overall performance [18]. This may be owing to the time-varying c_1 and c_2 that can balance the global and local search abilities, which implies that adaptation of c_1 and c_2 can be promising in enhancing the PSO performance. Hence, this paper will further investigate the effects of c_1 and c_2 and develop an optimal adaptation strategy according to ESE.

Another active research trend in PSO is hybrid PSO, which combines PSO with other evolutionary paradigms. Angeline [21] first introduced into PSO a selection operation similar to that in a genetic algorithm (GA). Hybridization of GA and PSO has been used in [38] for recurrent artificial neural network design. In addition to the normal GA operators, e.g., selection [21], crossover [22], and mutation [23], other techniques such as local search [24] and differential evolution [39] have been used to combine with PSO. Cooperative approach [40], self-organizing hierarchical technique [18], deflection, stretching, and repulsion techniques [41] have also been hybridized with traditional PSO to enhance performance. Inspired by biology, some researchers introduced niche [42], [43] and speciation [44] techniques into PSO to prevent the swarm from crowding too closely and to locate as many optimal solutions as possible.

In addition to research on parameter control and auxiliary techniques, PSO topological structures are also widely studied. The LPSO with a ring topological structure and the von Neumann topological structure PSO (VPSO) have been proposed by Kennedy and Mendes [45], [46] to enhance the performance in solving multimodal problems. Further, dynamically changing neighborhood structures have been proposed by Suganthan [37], Hu and Eberhart [47], and Liang and Suganthan [48] to avoid the deficiencies of fixed neighborhoods. Moreover, in the "fully informed particle swarm" (FIPS) algorithm [49], the information of the entire neighborhood is used to guide the particles. The CLPSO in [12] lets the particle use different $pBest$'s to update its flying on different dimensions for improved performance in multimodal applications.

III. ESE FOR PSO

To more objectively and optimally control the PSO, this section develops an ESE approach. During a PSO process, the population distribution characteristics vary not only with the generation number but also with the evolutionary state. For example, at an early stage, the particles may be scattered in various areas, and, hence, the population distribution is dispersive. As the evolutionary process goes on, however, particles would cluster together and converge to a locally or globally optimal area. Hence, the population distribution information would be different from that in the early stage. Therefore, how to detect the different population distribution information and how to use this information to estimate the evolutionary state would be a significant and promising research topic in PSO. The notion of evolutionary states was first introduced in [50] and [51], where a clustering analysis technique was used to determine the states. This section extends this technique to systematic ESE with a fuzzy classification option.

A. Population Distribution Information in PSO

In this section, the population distribution characteristics in a PSO process are first investigated so as to formulate an ESE approach. For this, a total of 12 commonly used test functions [12], [53], [54] are adopted to later benchmark the performance in this paper (including the tests in Section IV-B on the effects of parameter adaptation, the benchmark experiments in Section V, and the merit and sensitivity investigations in Section VI). These functions are summarized in Table I, where D represents the number of dimensions of the test function, and Column 6 defines an "acceptance" value to gauge whether a solution found by the nondeterministic PSO would be acceptable or not.

To illustrate the dynamics of particle distribution in the PSO process, we herein take a time-varying 2-D Sphere function

$$f_1(x-r)=(x_1-r)^2+(x_2-r)^2, \qquad x_i\in[-10,10] \quad (6)$$

as an example, where r is initialized to -5 and shifts to 5 at the fiftieth generation in a 100-generation optimization process. That is, the theoretical minimum of f_1 shifts from $(-5, -5)$ to (5, 5) half way in the search process. Using a GPSO [13] with 100 particles to solve this minimization problem, the population distributions in various running phases were observed, as shown in Fig. 1.

It can be seen in Fig. 1(a) that following the initialization, the particles start to explore throughout the search space without an evident control center. Then, the learning mechanisms of the PSO pull many particles to swarm together toward the optimal region, as seen in Fig. 1(b). Then, the population converges to the best particle [in Fig. 1(c)]. At the fiftieth generation, the bottom of the sphere is shifted from $(-5, -5)$ to (5, 5). It is seen in Fig. 1(d) that a new leader quickly emerges somewhat far away from the current clustering swarm. It leads the swarm to jump out of the previous optimal region to the new region [Fig. 1(e)], forming a second convergence [Fig. 1(f)]. From this simple investigation, it can be seen that the population distribution information can significantly vary during the run time, and that the PSO has the ability to adapt to a time-varying environment.

B. ESE

Based on the search behaviors and the population distribution characteristics of the PSO, an ESE approach is developed in

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE I
TWELVE TEST FUNCTIONS USED IN THIS PAPER, THE FIRST SIX BEING UNIMODAL AND THE REMAINING BEING MULTIMODAL

	Test function	D	Search Space	Global $f_{\min}$	Acceptance	Name of function
Unimodal	$f_1(x)=\sum_{i=1}^{D} x_i^2$	30	$[-100,100]^D$	0	0.01	Sphere [53]
	$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	30	$[-10,10]^D$	0	0.01	Schwefel's P2.22 [53]
	$f_3(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$	30	$[-100,100]^D$	0	100	Quadric [53]
	$f_4(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	30	$[-10,10]^D$	0	100	Rosenbrock [53]
	$f_5(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	30	$[-100,100]^D$	0	0	Step [53]
	$f_6(x)=\sum_{i=1}^{D} ix_i^4+random[0,1)$	30	$[-1.28,1.28]^D$	0	0.01	Quadric Noise [53]
Multimodal	$f_7(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{x_i})$	30	$[-500,500]^D$	-12569.5	-10000	Schwefel [53]
	$f_8(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	30	$[-5.12,5.12]^D$	0	50	Rastrigin [53]
	$f_9(x)=\sum_{i=1}^{D}[y_i^2-10\cos(2\pi y_i)+10]$ where $y_i=\begin{cases} x_i & \lvert x_i\rvert<0.5 \\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\geq 0.5 \end{cases}$	30	$[-5.12,5.12]^D$	0	50	Noncontinuous Rastrigin [12]
	$f_{10}(x)=-20\exp(-0.2\sqrt{1/D\sum_{i=1}^{D}x_i^2})-\exp(1/D\sum_{i=1}^{D}\cos 2\pi x_i)+20+e$	30	$[-32,32]^D$	0	0.01	Ackley [53]
	$f_{11}(x)=1/4000\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(x_i/\sqrt{i})+1$	30	$[-600,600]^D$	0	0.01	Griewank [53]
	$f_{12}(x)=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ where $y_i=1+\frac{1}{4}(x_i+1)$, $u(x_i,a,k,m)=\begin{cases} k(x_i-a)^m, & x_i>a \\ 0, & -a\leq x_i\leq a \\ k(-x_i-a)^m, & x_i<-a \end{cases}$	30	$[-50,50]^D$	0	0.01	Generalized Penalized [53]

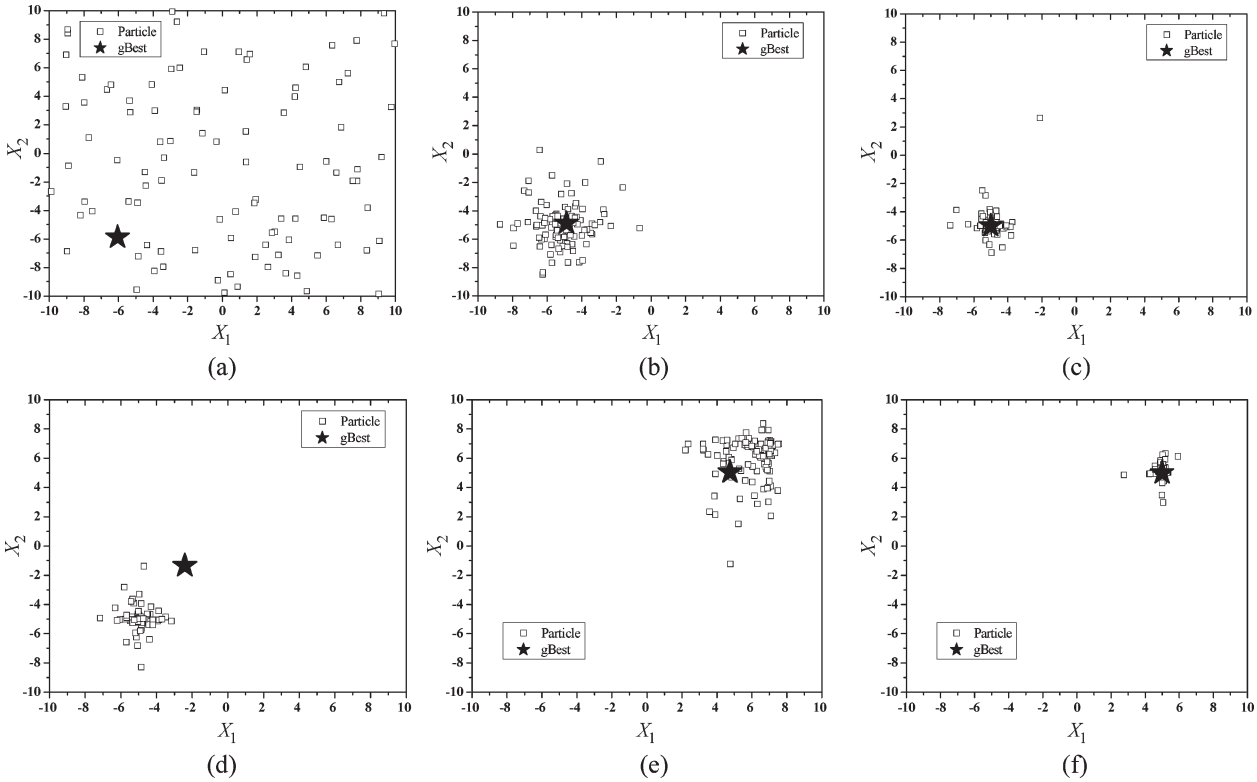


Fig. 1. Population distribution observed at various stages in a PSO process. (a) Generation = 1. (b) Generation = 25. (c) Generation = 49. (d) Generation = 50. (e) Generation = 60. (f) Generation = 80.

this section. The distribution information in Fig. 1 can be formulated as illustrated in Fig. 2 by calculating the mean distance of each particle to all the other particles. It is reasonable to expect that the mean distance from the globally best particle to other particles would be minimal in the convergence state since the global best tends to be surrounded by the swarm. In contrast, this mean distance would be maximal in the jumping-out state, because the global best is likely to be away from the crowding swarm. Therefore, the ESE approach will take into account the population distribution information in every generation, as detailed in the following steps.

Step 1: At the current position, calculate the mean distance of each particle i to all the other particles. For

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

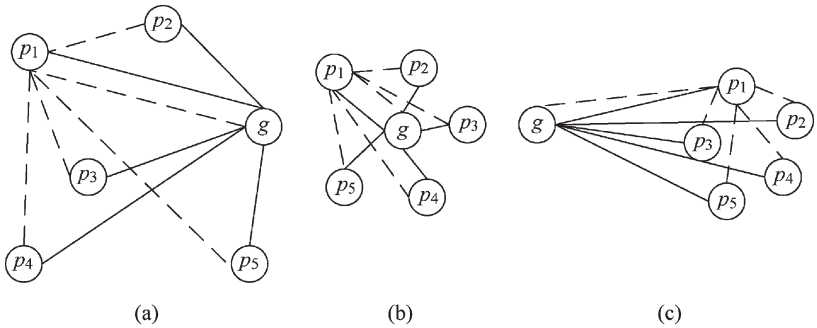


Fig. 2. PSO population distribution information quantified by an evolutionary factor f. (a) $d_g \approx d_{p_i}$ exploring. (b) $d_g \ll d_{p_i}$ exploiting converging. (c) $d_g \gg d_{p_i}$ jumping out.

example, this mean distance can be measured using an Euclidian metric

$$d_i = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} \left(x_i^k - x_j^k\right)^2} \quad (7)$$

where N and D are the population size and the number of dimensions, respectively.

Step 2: Denote d_i of the globally best particle as d_g. Compare all d_i's, and determine the maximum and minimum distances $d_{\max}$ and $d_{\min}$. Compute an "evolutionary factor" f as defined by

$$f = \frac{d_g - d_{\min}}{d_{\max} - d_{\min}} \in [0, 1]. \quad (8)$$

Take the time-varying f_1 minimization process shown in Fig. 1 as an example to illustrate the variations of f. The dynamics of f has been recorded and are shown in Fig. 3(a). As can be seen, the exploration phase (for about 6 generations) exhibits a large f, followed by a rapidly decreasing f during the exploitation phase (until about Generation 25), and a vanishing f during the convergence phase until the environment changes. When the search target shifts at Generation 50, the PSO is seen to be able to jump out, resulting in the largest value of f, followed by exploration and exploitation again until another convergence emerges.

For generality, this experiment was repeated on f_2, f_4, and f_7. Here, the unimodal functions f_2 and f_4 are also time varying as (6). The results are also plotted in Fig. 3, which shows similar patterns to that from f_1. It can be seen that the values of f, as derived from the population characteristics, robustly reveal the state that the PSO is in at run time.

Step 3: Classify f into one of the four sets S_1, S_2, S_3, and S_4, which represent the states of exploration, exploitation, convergence, and jumping out, respectively. These sets can be simple crisp intervals for a rigid classification. For example, referring to Fig. 3, $f = 0.5 \in [0.5, 0.7)$ can be classified by S_1 to signal that PSO would be in the state of exploration. However, analysis from Figs. 1–3 suggests that the state transition would be nondeterministic and fuzzy, and that different algorithms or applications could exhibit different characters of the transition. It is, hence, recommended that the fuzzy classification be adopted. Therefore, sets S_1, S_2, S_3, and S_4 are assigned the fuzzy membership functions depicted in Fig. 4, which are derived from Figs. 1–3 and their empirical studies. The key to fuzzy classification is overlap memberships. The formulation for numerical implementation of the classification is as follows.

Case (a)—Exploration: A medium to large value of f represents S_1, whose membership function is defined as

$$\mu_{S_1}(f) = \begin{cases} 0, & 0 \le f \le 0.4 \\ 5 \times f - 2, & 0.4 < f \le 0.6 \\ 1, & 0.6 < f \le 0.7 \\ -10 \times f + 8, & 0.7 < f \le 0.8 \\ 0, & 0.8 < f \le 1. \end{cases} \quad (9a)$$

Case (b)—Exploitation: A shrunk value of f represents S_2, whose membership function is defined as

$$\mu_{S_2}(f) = \begin{cases} 0, & 0 \le f \le 0.2 \\ 10 \times f - 2, & 0.2 < f \le 0.3 \\ 1, & 0.3 < f \le 0.4 \\ -5 \times f + 3, & 0.4 < f \le 0.6 \\ 0, & 0.6 < f \le 1. \end{cases} \quad (9b)$$

Case (c)—Convergence: A minimal value of f represents S_3, whose membership function is defined as

$$\mu_{S_3}(f) = \begin{cases} 1, & 0 \le f \le 0.1 \\ -5 \times f + 1.5, & 0.1 < f \le 0.3 \\ 0, & 0.3 < f \le 1. \end{cases} \quad (9c)$$

Case (d)—Jumping Out: When PSO is jumping out of a local optimum, the globally best particle is distinctively away from the swarming cluster, as shown in Fig. 2(c). Hence, the largest values of f reflect S_4, whose membership function is, thus, defined as

$$\mu_{S_4}(f) = \begin{cases} 0, & 0 \le f \le 0.7 \\ 5 \times f - 3.5, & 0.7 < f \le 0.9 \\ 1, & 0.9 < f \le 1. \end{cases} \quad (9d)$$

Therefore, at a transitional period, two memberships will be activated, and f can be classified to either state. For a final

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

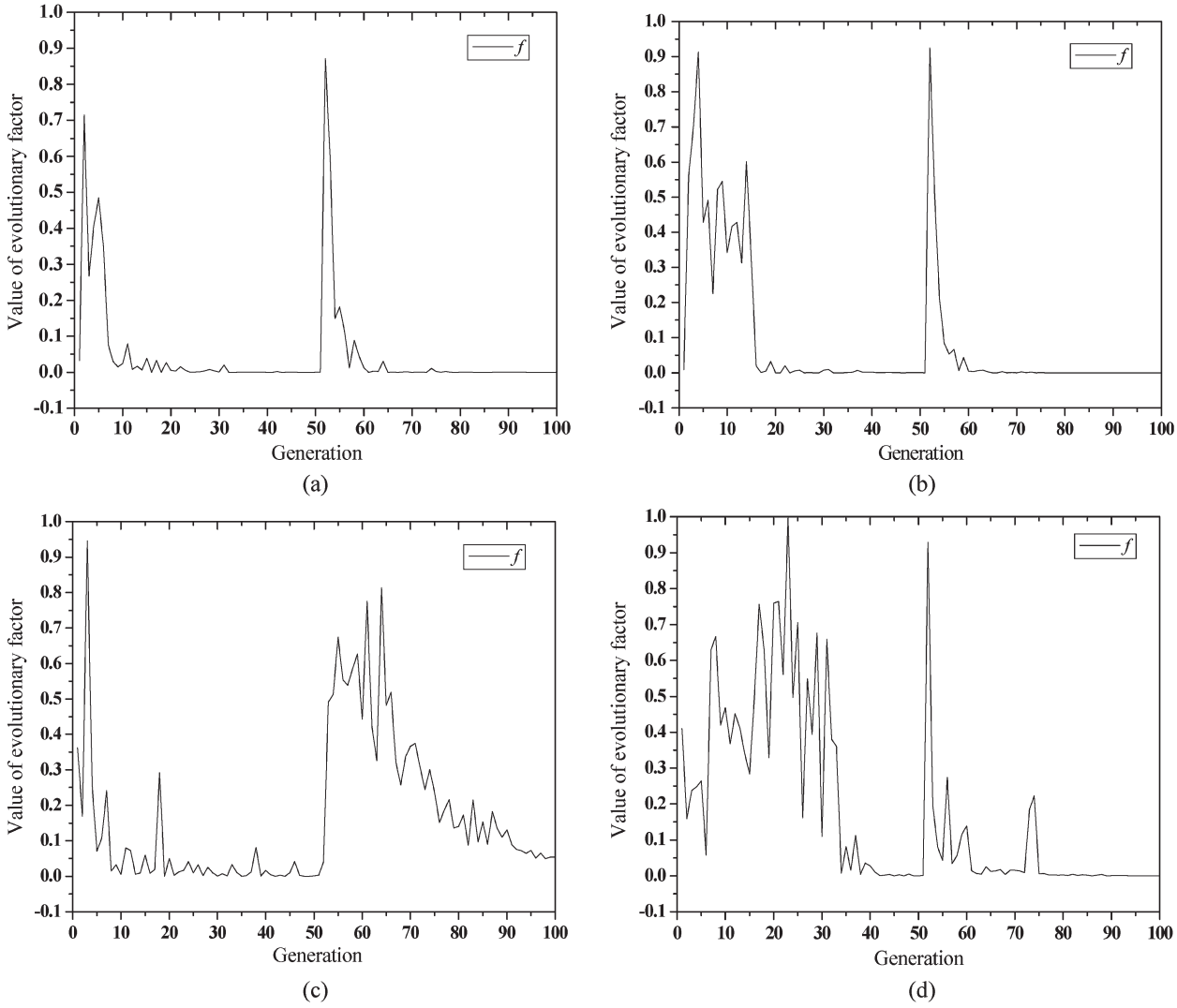


Fig. 3. Evolutionary state information robustly revealed by f at run time. (a) f from time-varying Sphere function f_1. (b) f from time varying Schwefel's function f_2. (c) f from time-varying Rosenbrock's function f_4. (d) f from multimodal function f_7.

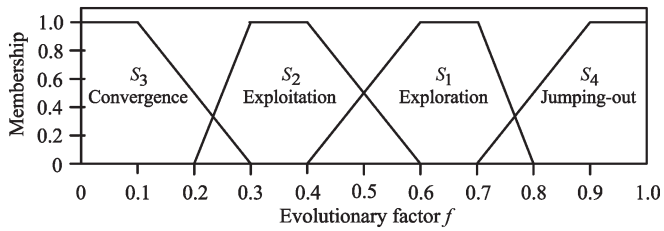


Fig. 4. Fuzzy membership functions for the four evolutionary states.

classification, either of the two most commonly used defuzzification techniques, i.e., the "singleton" or the "centroid" method [55], may be used here. The singleton method is adopted in this paper, since it is more efficient than the centroid and is simple to implement in conjunction with a state transition rule base.

Similar to most fuzzy logic schemes, the decision-making rule base here also involves both the state and the "change of state" variables in a 2-D table. The change of state is reflected by the PSO sequence $S_1 \Rightarrow S_2 \Rightarrow S_3 \Rightarrow S_4 \Rightarrow S_1 \ldots$, as observed in Figs. 1–3. Hence, for example, an f evaluated to 0.45 has both a degree of membership for S_1 and another degree of membership for S_2, which indicate that the PSO is in a transitional period between S_1 and S_2. Using the singleton method alone without the rule base would classify f to S_2, since $\mu_{S_2}(f) > \mu_{S_1}(f)$. However, with the rule base, the singleton will single out S_1 over S_2 if the previous state is S_4, because the rule base (containing the change sequence) determines decision making at defuzzification. If the previous state is S_1, then f is also classified to S_1 for the sake of classification stability, that is, not to excessively switch the state indicator. However, if the previous state is either S_2 or S_3, then the singleton with the rule table will classify f to S_2.

IV. APSO

A. Adaptive Control of PSO Parameters

1) Adaptation of the Inertia Weight: The inertia weight ω in PSO is used to balance the global and local search capabilities. Many researchers have advocated that the value of ω should be large in the exploration state and small in the exploitation state [4], [13], [14]. However, it is not necessarily correct to decrease ω purely with time [14]. The evolutionary factor f

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

shares some characteristics with the inertia weight ω in that f is also relatively large during the exploration state and becomes relatively small in the convergence state. Hence, it would be beneficial to allow ω to follow the evolutionary states using a sigmoid mapping $\omega(f) : \Re^+ \mapsto \Re^+$

$$\omega(f) = \frac{1}{1 + 1.5e^{-2.6f}} \in [0.4, 0.9] \qquad \forall f \in [0, 1]. \tag{10}$$

In this paper, ω is initialized to 0.9. As ω is not necessarily monotonic with time, but monotonic with f, ω will, thus, adapt to the search environment characterized by f. In a jumping-out or exploration state, the large f and ω will benefit the global search, as referenced earlier. Conversely, when f is small, an exploitation or convergence state is detected, and, hence, ω decreases to benefit the local search.

2) Control of the Acceleration Coefficients: Adaptive control can be devised for the acceleration coefficients based on the following notion. Parameter c_1 represents the "self-cognition" that pulls the particle to its own historical best position, helping explore local niches and maintaining the diversity of the swarm. Parameter c_2 represents the "social influence" that pushes the swarm to converge to the current globally best region, helping with fast convergence [4], [18]. These are two different learning mechanisms and should be given different treatments in different evolutionary states [51]. In this paper, the acceleration coefficients are both initialized to 2.0 and adaptively controlled according to the evolutionary state, with strategies developed as follows.

Strategy 1—Increasing c_1 and Decreasing c_2 in an Exploration State: It is important to explore as many optima as possible in the exploration state. Hence, increasing c_1 and decreasing c_2 can help particles explore individually and achieve their own historical best positions, rather than crowd around the current best particle that is likely to be associated with a local optimum.

Strategy 2—Increasing c_1 Slightly and Decreasing c_2 Slightly in an Exploitation State: In this state, the particles are making use of local information and grouping toward possible local optimal niches indicated by the historical best position of each particle. Hence, increasing c_1 slowly and maintaining a relatively large value can emphasize the search and exploitation around $pBest_i$. In the mean time, the globally best particle does not always locate the global optimal region at this stage yet. Therefore, decreasing c_2 slowly and maintaining a small value can avoid the deception of a local optimum. Further, an exploitation state is more likely to occur after an exploration state and before a convergence state. Hence, changing directions for c_1 and c_2 should slightly be altered from the exploration state to the convergence state.

Strategy 3—Increasing c_1 Slightly and Increasing c_2 Slightly in a Convergence State: In the convergence state, the swarm seems to find the globally optimal region, and, hence, the influence of c_2 should be emphasized to lead other particles to the probable globally optimal region. Thus, the value of c_2 should be increased. On the other hand, the value of c_1 should be decreased to let the swarm converge fast. However, extensive experiments on optimizing the 12 benchmark functions given in Table I revealed that such a strategy would prematurely saturate the two parameters to their lower and upper bounds, respectively. The consequence is that the swarm will strongly be attracted by the current best region, causing premature convergence, which is harmful if the current best region is a local optimum. To avoid this, both c_1 and c_2 are slightly increased.

Note that, slightly increasing both acceleration parameters will eventually have the same desired effect as reducing c_1 and increasing c_2, because their values will be drawn to around 2.0 due to an upper bound of 4.0 for the sum of c_1 and c_2 (refer to (12) discussed in the following section).

Strategy 4—Decreasing c_1 and Increasing c_2 in a Jumping-Out State: When the globally best particle is jumping out of local optimum toward a better optimum, it is likely to be far away from the crowding cluster. As soon as this new region is

TABLE II
STRATEGIES FOR THE CONTROL OF c_1 AND c_2

State	Strategy	c_1	c_2
Exploration	Strategy 1	Increase	Decrease
Exploitation	Strategy 2	Increase slightly	Decrease slightly
Convergence	Strategy 3	Increase slightly	Increase slightly
Jumping-out	Strategy 4	Decrease	Increase

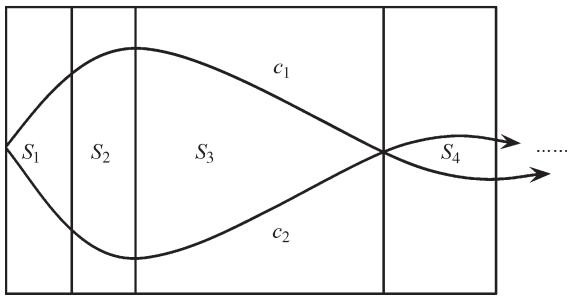


Fig. 5. End results of acceleration coefficient adjusting based on the evolutionary state.

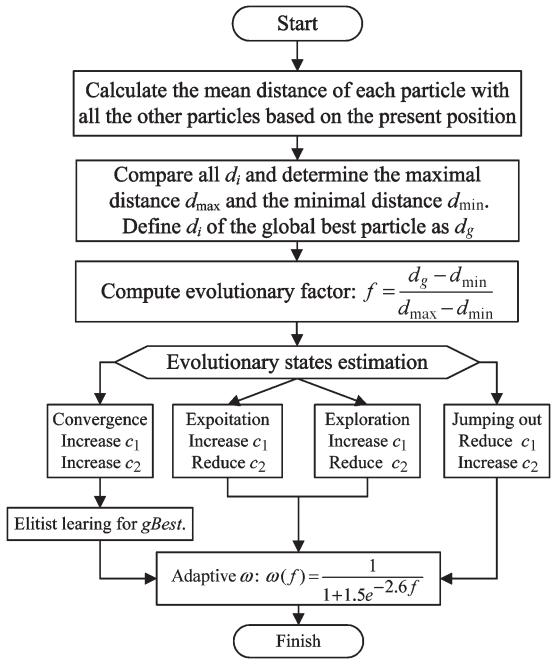


Fig. 6. Flowchart of ESE and the adaptive parameter control process.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE III
MEAN FEs IN OBTAINING ACCEPTABLE SOLUTIONS BY VARIOUS PSOs WITH AND WITHOUT PARAMETER ADAPTATION

PSOs		f_1	f_2	f_4	f_7	f_8	f_{10}
GPSO	Original (FEs)	106367	103077	98112	85973	93107	110844
	Adaptive (FEs)	**6578**	**8097**	**6683**	-	**3570**	**8039**
	Speedup (Original/Adaptive)	16.17	12.73	14.68	-	26.08	13.34
LPSO	Original (FEs)	117528	115441	99606	100150	100338	125543
	Adaptive (FEs)	**12756**	**11848**	**8895**	**51112**	**15473**	**21311**
	Speedup (Original/Adaptive)	9.21	9.74	11.20	1.96	6.48	5.89
VPSO	Original (FEs)	112620	109849	98742	91572	91699	118926
	Adaptive (FEs)	**8984**	**8730**	**6607**	-	**4429**	**12663**
	Speedup (Original/Adaptive)	12.54	12.58	14.95	-	20.70	9.39
CLPSO	Original (FEs)	70599	66525	73299	39495	53962	76646
	Adaptive (FEs)	**47452**	**45983**	**51523**	**18106**	**19355**	**62303**
	Speedup (Original/Adaptive)	1.49	1.45	1.42	2.18	2.79	1.23

found by a particle, which becomes the (possibly new) leader, others should follow it and fly to this new region as fast as possible. A large c_2 together with a relatively small c_1 helps to obtain this goal.

These four strategies are summarized in Table II, and the likely variations of the acceleration coefficients with the evolutionary state are illustrated in Fig. 5.

3) Bounds of the Acceleration Coefficients: As discussed earlier, the above adjustments on the acceleration coefficients should not be too irruptive. Hence, the maximum increment or decrement between two generations is bounded by

$$|c_i(g+1) - c_i(g)| \leq \delta, \qquad i = 1, 2 \tag{11}$$

where δ is termed the "*acceleration rate*" in this paper. Experiments reveal that a uniformly generated random value of δ in the interval [0.05, 0.1] performs best on most of the test functions (refer to Section VI-B). Note that we use 0.5 δ in strategies 2 and 3, where "slight" changes are recommended.

Further, the interval [1.5, 2.5] is chosen to clamp both c_1 and c_2 [52]. Similar to Clerc's constriction factor [29] and the polarized "competitive learning" paradigm in artificial neural networks, here, the interval [3.0, 4.0] suggested in [51] is used to bound the sum of the two parameters. If the sum is larger than 4.0, then both c_1 and c_2 are normalized to

$$c_i = \frac{c_i}{c_1 + c_2} 4.0, \qquad i = 1, 2. \tag{12}$$

The entire process of the ESE-enabled adaptive parameter control is illustrated in Fig. 6.

B. Effects of Parameter Adaptation

To test its effect, parameter adaptation is applied in this section to some well-known PSO algorithms, namely, GPSO [13], LPSO [45], VPSO [45], and CLPSO [12]. Note that CLPSO has only one acceleration parameter c, which is, hence, set to 2.0. These modified PSOs are then compared with their original versions using multidimensional unimodal and multimodal test functions.

Comparative tests have been performed using functions f_1, f_2, f_4, f_7, f_8, and f_{10} listed in Table I for a maximum of 2.0×10^5 FEs each. The results in terms of the minimum number of FEs that were required to reach the value of acceptance are listed in Table III. They are the mean values of all such successful runs over 30 independent trials. The reconfirmation that GPSO converges faster than LPSO while VPSO has a medium convergence speed [45] verifies that the tests are valid. The most interesting result is that parameter adaptation has, indeed, significantly speeded up the PSO, no matter for unimodal or multimodal functions. For example, the GPSO with adaptive parameters is about 16 times faster than GPSO without adaptive parameters on f_1, and the speedup ratio is about 26 on f_8. The speedup ratios in Table III have shown that efficiency is much more evident for GPSO, whereas VPSO, LPSO, and CLPSO rank second, third, and fourth, respectively. The mean values and the best values of all 30 trials are presented in Table IV. Boldface in the table indicates the best result obtained.

Note that, however, the results also reveal that none of the 30 runs of GPSO and VPSO with adaptive parameters successfully reached an acceptable accuracy on f_7 (Schwefel's function) by 2.0×10^5 FEs, as denoted by the symbol "$-$" in Table III. This is because the global optimum of f_7 is far away from any of the local optima [53], and there is no jumping-out mechanism implemented at the same time as parameter adaptation that accelerates convergence. In the original GPSO, for example, when the evolution is in a convergence state, the particles are refining the solutions around the globally best particle. Hence, according to (1), when $nBest$ becomes $gBest$, the self-cognition and the social influence learning components for the globally best particle are both nearly zero. Further, its velocity will become smaller and smaller as the inertia weight is smaller than 1. The standard learning mechanism does not help $gBest$ escape from this local optimum since its velocity approaches 0.

C. ELS

The failures of using parameter adaptation alone for GPSO and VPSO on Schwefel's function suggest that a jumping-out mechanism would be necessary for enhancing the globality of these search algorithms. Hence, an "ELS" is designed here and applied to the globally best particle so as to help jump out of local optimal regions when the search is identified to be in a convergence state.

Unlike the other particles, the global leader has no exemplars to follow. It needs fresh momentum to improve itself. Hence, a perturbation-based ELS is developed to help $gBest$ push itself

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE IV
MEAN SOLUTIONS AND BEST SOLUTIONS OF THE 30 TRIALS OBTAINED BY VARIOUS PSOs WITH AND WITHOUT PARAMETER ADAPTATION

PSOs			f_1	f_2	f_4	f_7	f_8	f_{10}
GPSO	Original	Mean	5.37×10^{-51}	2.51×10^{-34}	24.486	**-10032.5**	**29.6099**	$\mathbf{1.15\times10^{-14}}$
		Best	6.69×10^{-57}	2.02×10^{-37}	0.393448	**-11029.8**	**16.9143**	$\mathbf{7.69\times10^{-15}}$
	Adaptive	Mean	$\mathbf{2.66\times10^{-160}}$	$\mathbf{1.74\times10^{-90}}$	**10.2012**	-7254.36	52.9317	1.35991
		Best	$\mathbf{7.69\times10^{-174}}$	$\mathbf{1.12\times10^{-97}}$	**0.001884**	-8483.2	21.8891	$\mathbf{7.69\times10^{-15}}$
LPSO	Original	Mean	6.33×10^{-29}	2.03×10^{-20}	18.9472	**-9592.54**	**35.0819**	1.85×10^{-14}
		Best	4.50×10^{-32}	9.30×10^{-22}	0.58546	**-10417.8**	**24.874**	1.12×10^{-14}
	Adaptive	Mean	$\mathbf{3.88\times10^{-84}}$	$\mathbf{6.84\times10^{-57}}$	**16.444**	-8201.3	44.531	$\mathbf{6.98\times10^{-15}}$
		Best	$\mathbf{1.43\times10^{-89}}$	$\mathbf{5.45\times10^{-58}}$	**0.00162**	**-10417.8**	26.8639	$\mathbf{4.14\times10^{-15}}$
VPSO	Original	Mean	8.49×10^{-39}	6.29×10^{-27}	35.9005	**-9884**	**27.7394**	$\mathbf{1.46\times10^{-14}}$
		Best	2.21×10^{-42}	7.34×10^{-29}	8.88851	**-10832.4**	**10.9445**	$\mathbf{1.12\times10^{-14}}$
	Adaptive	Mean	$\mathbf{8.70\times10^{-117}}$	$\mathbf{2.18\times10^{-76}}$	**14.0733**	-7682.45	40.8264	1.62×10^{-14}
		Best	$\mathbf{2.51\times10^{-122}}$	$\mathbf{2.32\times10^{-79}}$	**0.09016**	-9055.77	25.8689	$\mathbf{1.12\times10^{-14}}$
CLPSO	Original	Mean	1.06×10^{-19}	1.01×10^{-13}	9.2047	**-12560**	$\mathbf{7.40\times10^{-11}}$	2.01×10^{-12}
		Best	1.14×10^{-20}	2.73×10^{-14}	1.12274	**-12569.5**	3.91×10^{-14}	7.11×10^{-13}
	Adaptive	Mean	$\mathbf{4.00\times10^{-25}}$	$\mathbf{7.96\times10^{-17}}$	**0.70586**	-12506.3	0.239	$\mathbf{2.63\times10^{-14}}$
		Best	$\mathbf{2.28\times10^{-26}}$	$\mathbf{1.52\times10^{-17}}$	**0.04634**	**-12569.5**	**0**	$\mathbf{1.48\times10^{-14}}$

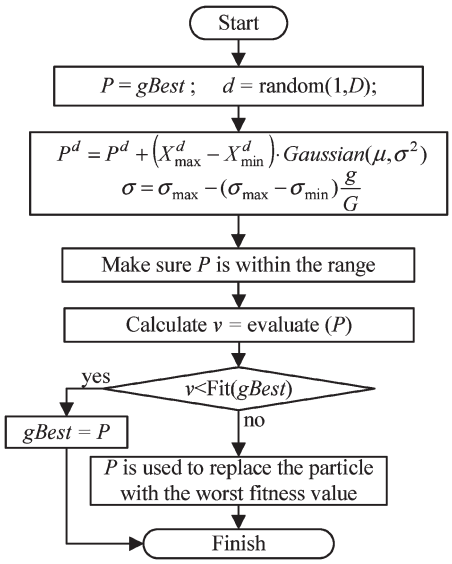


Fig. 7. Flowchart of ELS for $gBest$ upon convergence state emerging.

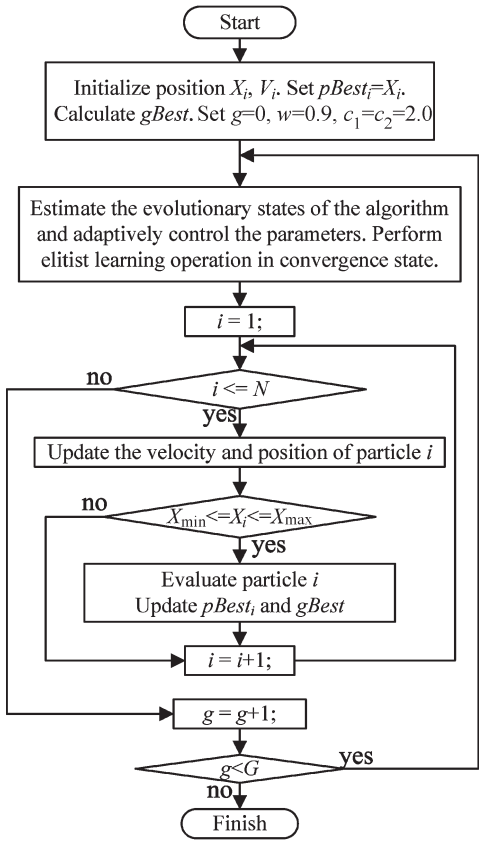


Fig. 8. Flowchart of the APSO algorithm.

out to a potentially better region. If another better region is found, then the rest of the swarm will follow the leader to jump out and converge to the new region.

The ELS randomly chooses one dimension of $gBest$'s historical best position, which is denoted by P^d for the dth dimension. Only one dimension is chosen because the local optimum is likely to have some good structure of the global optimum, and, hence, this should be protected. As every dimension has the same probability to be chosen, the ELS operation can be regarded to perform on every dimension in a statistical sense. Similar to simulated annealing, the mutation operation in evolutionary programming or in evolution strategies, elitist learning is performed through a Gaussian perturbation

$$P^d = P^d + \left(X^d_{\max} - X^d_{\min}\right)\cdot Gaussian(\mu,\sigma^2). \quad (13)$$

The search range $[X^d_{\min}, X^d_{\max}]$ is the same as the lower and upper bounds of the problem. The $Gaussian(\mu,\sigma^2)$ is a random number of a Gaussian distribution with a zero mean μ and a standard deviation (SD) σ, which is termed the "*elitist learning rate*." Similar to some time-varying neural network training schemes, it is suggested that σ be linearly decreased with the generation number, which is given by

$$\sigma = \sigma_{\max} - (\sigma_{\max} - \sigma_{\min})\frac{g}{G} \quad (14)$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the upper and lower bounds of σ, which represents the learning scale to reach a new region. Empirical study shows that $\sigma_{\max} = 1.0$ and $\sigma_{\min} = 0.1$ result in good performance on most of the test functions (refer to Section VI-C for an in-depth discussion). Alternatively, σ

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

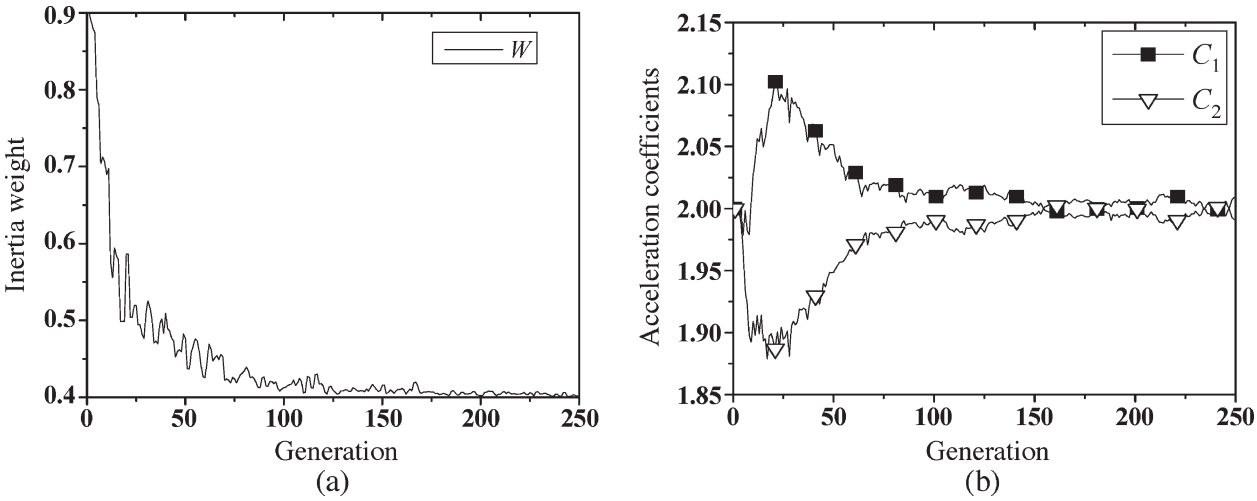


Fig. 9. Search behaviors of the APSO on Sphere function. (a) Mean value of the inertia weight during the run time showing an adaptive momentum. (b) Mean values of acceleration coefficients c_1 and c_2 adapting to the evolutionary states.

may geometrically be decreased, similar to the temperature-decreasing scheme in Boltzmann learning seen in simulated annealing. The ELS process is illustrated in Fig. 7.

In a statistical sense, the decreasing SD provides a higher learning rate in the early phase for $gBest$ to jump out of a possible local optimum, whereas a smaller learning rate in the latter phase guides the $gBest$ to refine the solution. In ELS, the new position will be accepted if and only if its fitness is better than the current $gBest$. Otherwise, the new position is used to replace the particle with the worst fitness in the swarm.

D. Search Behaviors of APSO

The complete flowchart of the APSO algorithm with adaptive parameters and ELS is shown in Fig. 8. Before applying the APSO to comprehensive tests on benchmark functions, we first investigate its search behaviors in unimodal and multimodal search spaces.

1) APSO in Unimodal Search Space: The search behavior of the APSO in a unimodal space has been investigated on the Sphere function (f_1 in Table I). In a unimodal space, it is important for an optimization or search algorithm to converge fast and to refine the solution for high accuracy. The inertia weight shown in Fig. 9(a) confirms that the APSO maintains a large ω in the exploration phase (for about 50 generations), and then a rapidly decreasing ω follows exploitation, leading to convergence, as the unique global optimum region is found by a leading particle, and the swarm follows it.

Fig. 9(b) shows how the ESE in APSO has influenced the acceleration coefficients. The curves for c_1 and c_2 somewhat show good agreement with the ones given in Fig. 5. It can be seen that c_1 increases while c_2 decreases in the exploration and exploitation phases. Then, c_1 and c_2 reverse their directions when the swarm converges, eventually returning to around 2.0. Then, trials in elitist learning perturb the particle that leads the swarm, which is reflected in the slight divergence between c_1 and c_2 that follows. The search behavior on the unimodal function indicates that the proposed APSO algorithm has indeed identified the evolutionary states and can adaptively control the parameters for improved performance.

2) APSO in Multimodal Search Space: Here, the APSO is tested again to see how it will adapt itself to a multimodal space. When solving multimodal functions, a search algorithm should maintain diversity of the population and search for as many optimal regions as possible. The search behavior of the APSO is investigated on Rastrigin's function (f_8 in Table I). To compare the diversity in the search by the APSO and the traditional PSO, a yardstick proposed in [56] is used here, called the "population standard deviation," which is denoted by psd as

$$psd = \sqrt{\left[\sum_{i=1}^{N}\sum_{j=1}^{D}\left(x_i^j - \overline{x}^j\right)^2\right] \Big/ (N-1)} \qquad (15)$$

where N, D, and $\overline{x}$ are the population size, the number of dimension, and the mean position of all the particles, respectively.

The variations in psd can indicate the diversity level of the swarm. If psd is small, then it indicates that the population has closely converged to a certain region, and the diversity of the population is low. A larger value of psd indicates that the population is of higher diversity. However, it does not necessarily mean that a larger psd is always better than a smaller one because an algorithm that cannot converge may also present a large psd. Hence, the psd needs to be considered together with the solution that the algorithm arrives at.

The results of psd comparisons are plotted in Fig. 10(a) and those of the evolutionary processes in Fig. 10(b). It can be seen that the APSO has an ability to jump out of the local optima, which is reflected by the regained diversity of the population, as revealed in Fig. 10(a), with a steady improvement in the solution, as shown in Fig. 10(b). Fig. 10(c) and (d) shows the inertia weight and the acceleration coefficient behaviors of the APSO, respectively. These plots confirm that, in a multimodal space, the APSO can also find a potential optimal region (maybe a local optimum) fast in an early phase and converge fast with a rapid decreasing diversity due to the

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

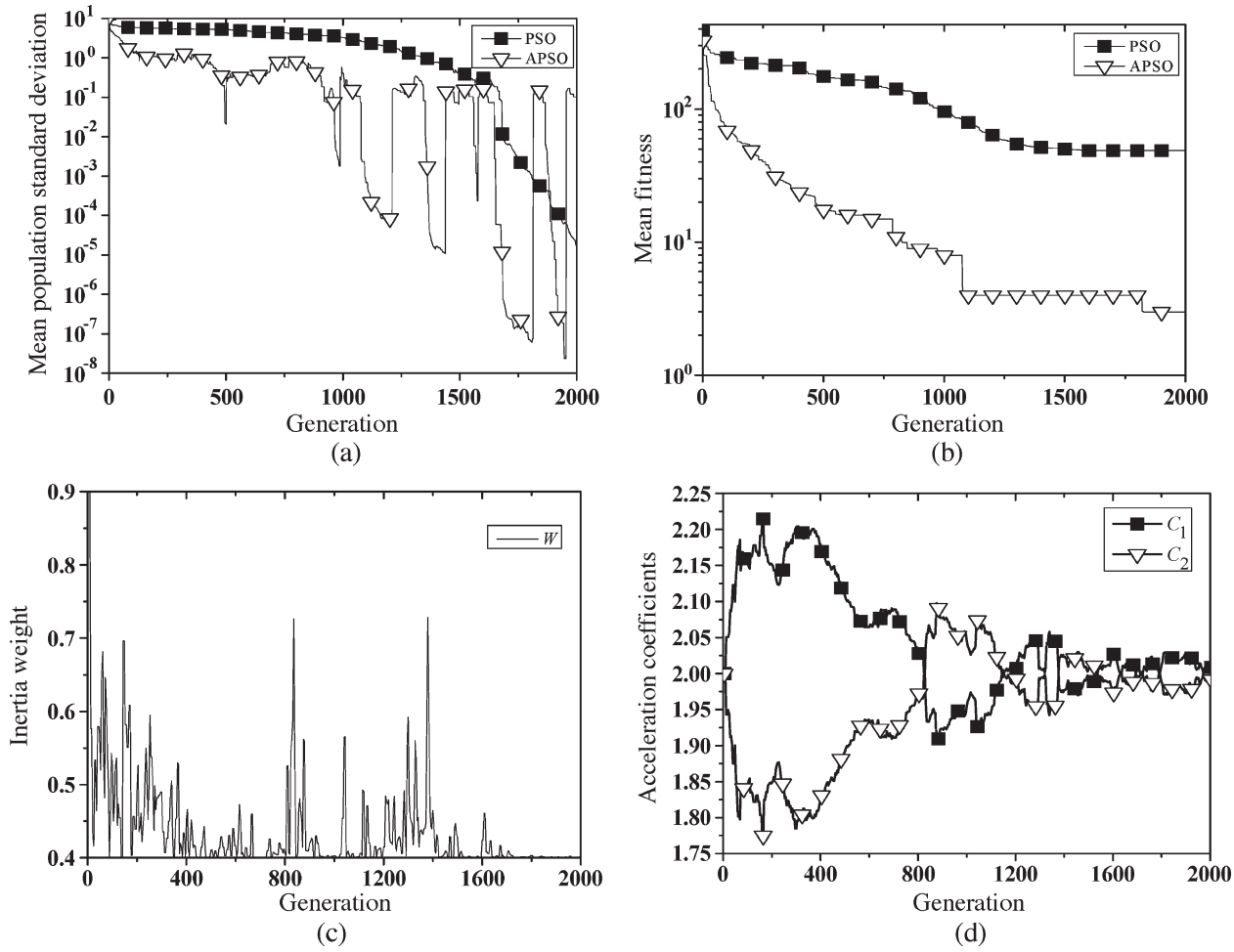


Fig. 10. Search behaviors of PSOs on Rastrigin's function. (a) Mean *psd* during run time. (b) Plots of convergence during the minimization run. (c) Mean value of the inertia weight during the run time showing an adaptive momentum. (d) Mean values of acceleration coefficients c_1 and c_2 adapting to the evolutionary states.

adaptive parameter strategies. However, if the current optimal region is local, then the swarm can separate and jump out. Hence, the APSO can appropriately increase the diversity of the population so as to explore for a better region owing to the ELS in the convergence state. This behavior with adaptive population diversity is valuable for a global search algorithm to prevent from being trapped in the local optima and to find the global optimum in a multimodal space.

V. Benchmark Tests and Comparisons

Further experimental tests with benchmark functions are carried out in this section to validate the proposed APSO techniques and to compare the APSO with existing PSOs.

A. Benchmark Functions and Algorithm Configuration

The twelve benchmark functions listed in Table I are used for the experimental tests here. Seven existing PSO algorithms, as detailed in Table V, are compared with the APSO. The first three PSOs (GPSO [13], LPSO with ring neighborhood [45] and VPSO with von Neumann neighborhood [45]) are regarded as standard PSOs and have widely been used in PSO applications. The FIPS [49] is a "fully informed" PSO that uses all the neighbors to influence the flying velocity. In FIPS, the URing topology structure is implemented with a weighted FIPS based on the goodness (wFIPS) algorithm for higher successful ratio, as recommended in [49]. HPSO-TVAC [18] is a "performance-improvement" PSO by improving the acceleration parameters and incorporating a self-organizing technique. Dynamic multi-swarm PSO (DMS-PSO) [48] is devoted to improve the topological structure in a dynamic way. Finally, in Table V, CLPSO offers a comprehensive-learning strategy, which aims at yielding better performance for multimodal functions [12]. The parameter configurations for these PSO variants are also given in Table V, according to their corresponding references.

TABLE V
PSO Algorithms Used in the Comparison

Algorithm	Year	Topology	Parameters Settings	Reference
GPSO	1998	Global Star	ω: 0.9-0.4, c_1= c_2=2.0	[13]
LPSO	2002	Local Ring	ω: 0.9-0.4, c_1= c_2=2.0	[45]
VPSO	2002	Local von Neumann	ω: 0.9-0.4, c_1= c_2=2.0	[45]
FIPS	2004	Local URing	χ=0.729, $\sum c_i$ = 4.1	[49]
HPSO-TVAC	2004	Global Star	ω: 0.9-0.4, c_1: 2.5-0.5, c_2: 0.5-2.5	[18]
DMS-PSO	2005	Dynamic Multi-swarm	ω: 0.9-0.2, c_1= c_2=2.0, m=3, R=5	[48]
CLPSO	2006	Comprehensive Learning	ω: 0.9-0.4, c =1.49445, m=7	[12]

In the tests, the algorithm configuration of the APSO is as follows. The inertia weight ω is initialized to 0.9, and c_1 and c_2 to 2.0, same as the common configuration in a standard PSO.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE VI
SEARCH RESULT COMPARISONS AMONG EIGHT PSOs ON 12 TEST FUNCTIONS

Functions		GPSO	LPSO	VPSO	FIPS	HPSO-TVAC	DMS-PSO	CLPSO	APSO
f_1	Mean	1.98×10^{-53}	4.77×10^{-29}	5.11×10^{-38}	3.21×10^{-30}	3.38×10^{-41}	3.85×10^{-54}	1.89×10^{-19}	**1.45×10^{-150}**
	Std. Dev	7.08×10^{-53}	1.13×10^{-28}	1.91×10^{-37}	3.60×10^{-30}	8.50×10^{-41}	1.75×10^{-53}	1.49×10^{-19}	**5.73×10^{-150}**
f_2	Mean	2.51×10^{-34}	2.03×10^{-20}	6.29×10^{-27}	1.32×10^{-17}	6.9×10^{-23}	2.61×10^{-29}	1.01×10^{-13}	**5.15×10^{-84}**
	Std. Dev	5.84×10^{-34}	2.89×10^{-20}	8.68×10^{-27}	7.86×10^{-18}	6.89×10^{-23}	6.6×10^{-29}	6.51×10^{-14}	**1.44×10^{-83}**
f_3	Mean	6.45×10^{-2}	18.60	1.44	0.77	2.89×10^{-7}	47.5	395	**1.0×10^{-10}**
	Std. Dev	9.46×10^{-2}	30.71	1.55	0.86	2.97×10^{-7}	56.4	142	**2.13×10^{-10}**
f_4	Mean	28.1	21.8627	37.6469	22.5387	13	32.3	11	**2.84**
	Std. Dev	24.6	11.1593	24.9378	0.310182	16.5	24.1	14.5	**3.27**
f_5	Mean	0	0	0	0	0	0	0	0
	Std. Dev	0	0	0	0	0	0	0	0
f_6	Mean	7.77×10^{-3}	1.49×10^{-2}	1.08×10^{-2}	**2.55×10^{-3}**	5.54×10^{-2}	1.1×10^{-2}	3.92×10^{-3}	4.66×10^{-3}
	Std. Dev	2.42×10^{-3}	5.66×10^{-3}	3.24×10^{-3}	**6.25×10^{-4}**	2.08×10^{-2}	3.94×10^{-3}	1.14×10^{-3}	1.7×10^{-3}
f_7	Mean	-10090.16	-9628.35	-9845.27	-10113.8	-10868.57	-9593.33	-12557.65	**-12569.5**
	Std. Dev	495	456.54	588.87	889.58	289	441	36.2	**5.22×10^{-11}**
f_8	Mean	30.7	34.90	34.09	29.98	2.39	28.1	2.57×10^{-11}	**5.8×10^{-15}**
	Std. Dev	8.68	7.25	8.07	10.92	3.71	6.42	6.64×10^{-11}	**1.01×10^{-14}**
f_9	Mean	15.5	30.40	21.33	35.91	1.83	32.8	0.167	**4.14×10^{-16}**
	Std. Dev	7.4	9.23	9.46	9.49	2.65	6.49	0.379	**1.45×10^{-15}**
f_{10}	Mean	1.15×10^{-14}	1.85×10^{-14}	1.4×10^{-14}	**7.69×10^{-15}**	2.06×10^{-10}	8.52×10^{-15}	2.01×10^{-12}	1.11×10^{-14}
	Std. Dev	2.27×10^{-15}	4.80×10^{-15}	3.48×10^{-15}	**9.33×10^{-16}**	9.45×10^{-10}	1.79×10^{-15}	9.22×10^{-13}	3.55×10^{-15}
f_{11}	Mean	2.37×10^{-2}	1.10×10^{-2}	1.31×10^{-2}	9.04×10^{-4}	1.07×10^{-2}	1.31×10^{-2}	**6.45×10^{-13}**	1.67×10^{-2}
	Std. Dev	2.57×10^{-2}	1.60×10^{-2}	1.35×10^{-2}	2.78×10^{-3}	1.14×10^{-2}	1.73×10^{-2}	**2.07×10^{-12}**	2.41×10^{-2}
f_{12}	Mean	1.04×10^{-2}	2.18×10^{-30}	3.46×10^{-3}	1.22×10^{-31}	7.07×10^{-30}	**2.05×10^{-32}**	1.59×10^{-21}	3.76×10^{-31}
	Std. Dev	3.16×10^{-2}	5.14×10^{-30}	1.89×10^{-2}	4.85×10^{-32}	4.05×10^{-30}	**8.12×10^{-33}**	1.93×10^{-21}	1.2×10^{-30}

These parameters are then adaptively controlled during the run. Parameter δ in (11) is a random value uniformly generated in the interval [0.05, 0.1], whereas the parameter σ in (13) linearly decreases from $\sigma_{\max} = 1.0$ to $\sigma_{\min} = 0.1$.

For a fair comparison among all the PSO algorithms, they are tested using the same population size of 20, a value of which is commonly adopted in PSO [4]. Further, all the algorithms use the same number of 2.0×10^5 FEs for each test function [54]. All the experiments are carried out on the same machine with a Celeron 2.26-GHz CPU, 256-MB memory, and Windows XP2 operating system. For the purpose of reducing statistical errors, each function is independently simulated 30 times, and their mean results are used in the comparison.

B. Comparisons on the Solution Accuracy

The performance on the solution accuracy of every PSO listed in Table V is compared with the APSO. The results are shown in Table VI in terms of the mean and SD of the solutions obtained in the 30 independent runs by each algorithm. Boldface in the table indicates the best result among those obtained by all eight contenders. Fig. 11 graphically presents the comparison in terms of convergence characteristics of the evolutionary processes in solving the 12 different problems.

An interesting result is that all the PSO algorithms have most reliably found the minimum of f_5. It is a region rather than a point in f_5 that is the optimum. Hence, this problem may relatively be easy to solve with a 100% success rate. The comparisons in both Table VI and Fig. 11 show that, when solving unimodal problems, the APSO offers the best performance on most test functions. In particular, the APSO offers the highest accuracy on functions f_1, f_2, f_3, f_4, and f_5, and ranks third on f_6.

The APSO also achieves the global optimum on the optimization of complex multimodal functions f_7, f_8, f_9, f_{10}, and f_{12}. Although CLPSO outperforms APSO and others on f_{11} (Griewank's function), its mean solutions on other functions are worse than those of the APSO. Further, the APSO can successfully jump out of the local optima on most of the multimodal functions and surpasses all the other algorithms on functions f_7, f_8, and f_9, where the global optimum of f_7 (Schwefel's function) is far away from any of the local optima [53], and the globally best solutions of f_8 and f_9 (continuous/noncontiguous Rastrigin's functions) are surrounded by a large number of local optima [12], [44]. The ability of avoiding being trapped into the local optima and achieving global optimal solutions to multimodal functions suggests that the APSO can indeed benefit from the ELS (cf. Table IV with respect to the much improved performance over the original GPSO and VPSO on f_7).

C. Comparisons on the Convergence Speed

The speed in obtaining the global optimum is also a salient yardstick for measuring the algorithm performance. Table VII reveals that the APSO generally offers a much higher speed, which is measured by either the mean number of FEs or by the mean CPU time needed to reach an acceptable solution. The CPU time is important to measure the computational load, as many existing PSO variants have added extra operations that cost computational time. Although the APSO needs to calculate

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

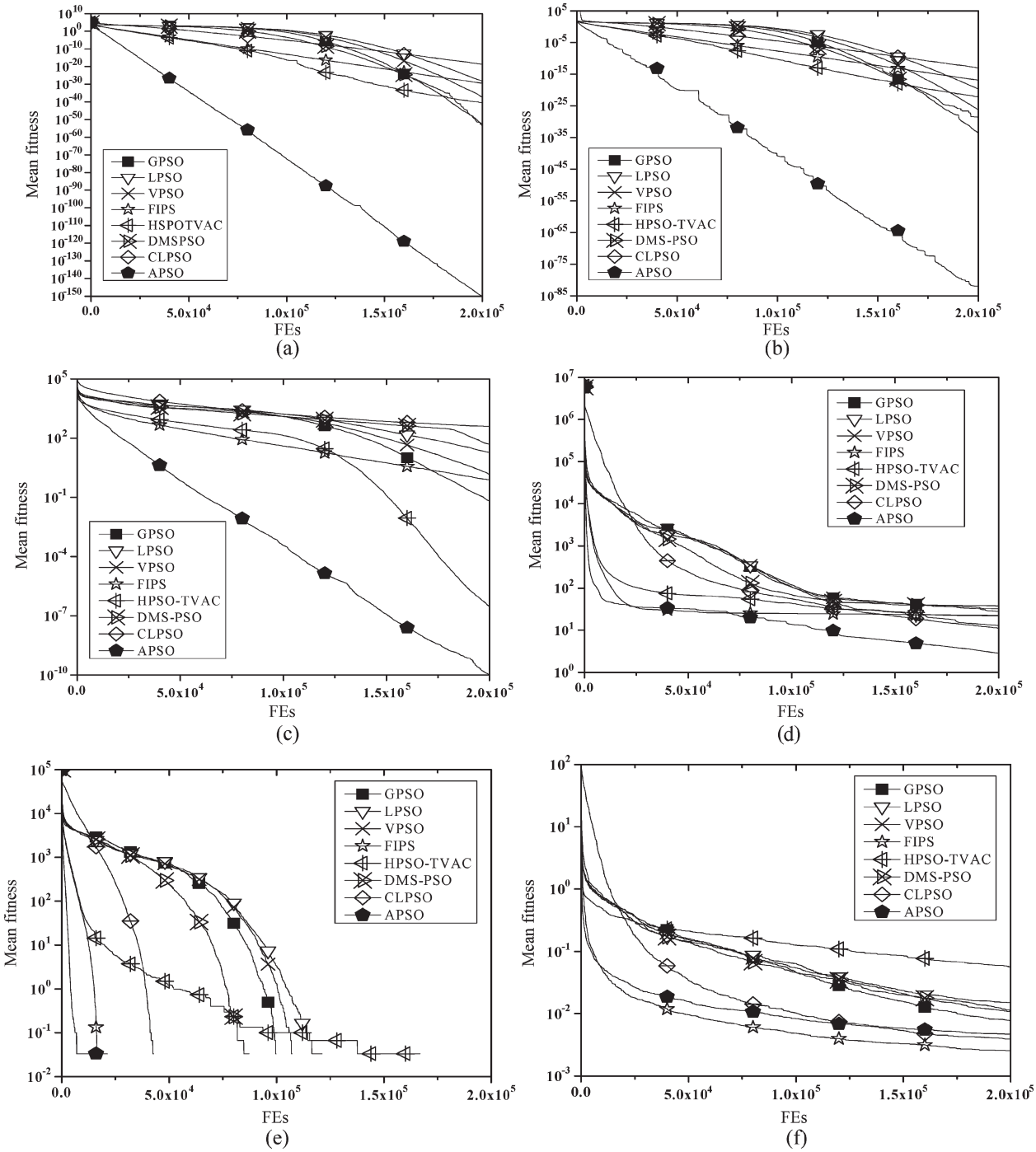


Fig. 11. Convergence performance of the eight different PSOs on the 12 test functions. (a) f_1. (b) f_2. (c) f_3. (d) f_4. (e) f_5. (f) f_6.

the mean distance between every pair of particles in the swarm, the calculation costs negligible CPU time.

In solving real-world problems, the "FE" time overwhelms the algorithm overhead. Hence, the mean number of FEs needed to reach acceptable accuracy would be much more interesting than the CPU time. Thus, the mean FEs are also explicitly presented and compared in Table VII. For example, tests on f_1 show that the average numbers of FEs of 105695, 118197, 112408, 32561, 30011, 91496, and 72081 are needed by the GPSO, LPSO, VPSO, FIPS, HPSO-TVAC, DMS-PSO, and CLPSO algorithms, respectively, to reach an acceptable solution. However, the APSO only uses 7074 FEs, whereas its CPU time of 0.11 s is also the shortest among the eight algorithms. In summary, the APSO uses the least CPU time and the smallest number of FEs to reach acceptable solutions on 9 out of 12 test functions (f_1, f_2, f_3, f_4, f_5, f_7, f_8, f_9, and f_{11}).

D. Comparisons on the Algorithm Reliability

Table VII also reveals that APSO offers a generally highest percentage of trials (reaching acceptable solutions) and the highest reliability averaged over all the test functions. The APSO reaches the acceptable solutions with a successful ratio of 100% on all the test functions except function f_{11}. Note that

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

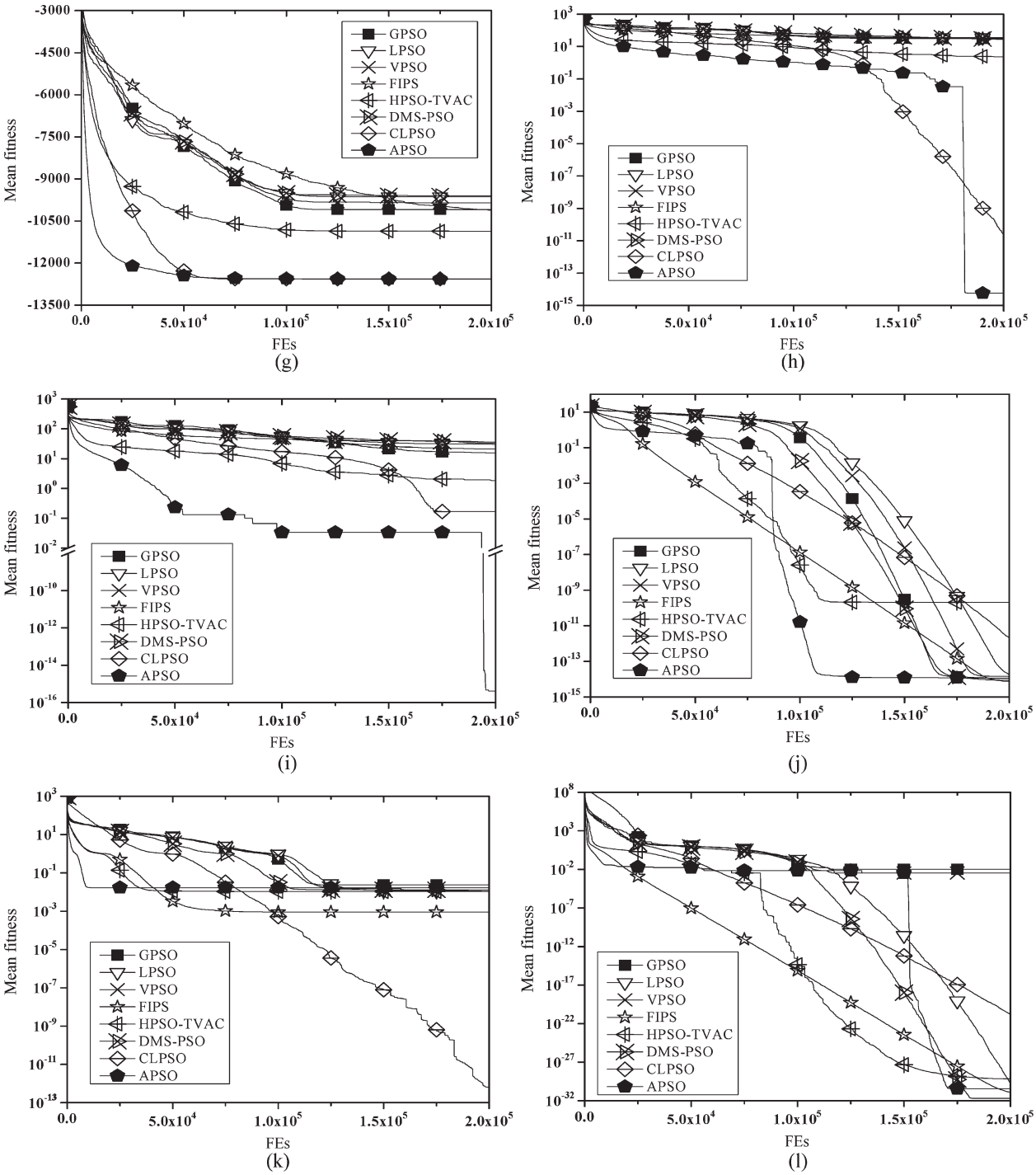


Fig. 11. (*Continued.*) Convergence performance of the eight different PSOs on the 12 test functions. (g) f_7. (h) f_8. (i) f_9. (j) f_{10}. (k) f_{11}. (l) f_{12}.

HPSO-TVAC and CLPSO did not converge on functions f_6 and f_3, respectively. For the mean reliability of all the test functions, APSO offers the highest reliability of 97.23%, followed by FIPS, CLPSO, GPSO, HPSO-TVAC, VPSO, DMS-PSO, and LPSO.

According to the theorem of "no free lunch" [57], one algorithm cannot offer better performance than all the others on every aspect or on every kind of problem. This is also observed in our experimental results. The GPSO outperforms LPSOs, including LPSO, VPSO, and FIPS with the U-Ring structure, on simple unimodal functions f_1, f_2, and f_3. However, on difficult unimodal functions (e.g., Rosenbrock's function f_4) and the multimodal functions, LPSO and FIPS offer better performance than GPSO. FIPS achieves the highest accuracy on function f_{10}, whereas CLPSO and DMS-PSO perform best on f_{11} and f_{12}, respectively, but these global algorithms sacrifice performance on unimodal functions. However, APSO outperforms most on both unimodal and multimodal functions, owing to its adaptive parameters that deliver faster convergence and to its adaptive ELS that avoids local optima. Further, such outperformance has been achieved with the highest success rate on all but Griewank's function (f_{11}).

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE VII
CONVERGENCE SPEED AND ALGORITHM RELIABILITY COMPARISONS (SPEED BEING MEASURED ON BOTH THE MEAN NUMBER OF FEs AND THE MEAN CPU TIME NEEDED TO REACH AN ACCEPTABLE SOLUTION; RELIABILITY "RATIO" BEING THE PERCENTAGE OF TRIAL RUNS REACHING ACCEPTABLE SOLUTIONS; "−" INDICATING NO TRIALS REACHED AN ACCEPTABLE SOLUTION)

	Function	GPSO	LPSO	VPSO	FIPS	HPSO-TVAC	DMS-PSO	CLPSO	APSO
	Mean FEs	105695	118197	112408	32561	30011	91496	72081	**7074**
f_1	Time (sec)	0.96	1.12	1.04	0.36	0.29	0.85	0.48	**0.11**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	103077	115441	109849	36322	31371	91354	66525	**7900**
f_2	Time (sec)	1.02	1.19	1.10	0.44	0.32	0.91	0.62	**0.17**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	137985	162196	147133	73790	102499	185588	-	**21166**
f_3	Time (sec)	2.16	2.69	2.33	1.35	1.67	2.91	-	**0.98**
	Ratio (%)	**100.0**	96.7	**100.0**	**100.0**	**100.0**	86.7	0.0	**100.0**
	Mean FEs	101579	102259	103643	13301	33689	87518	74815	**5334**
f_4	Time (sec)	0.99	1.05	1.05	0.16	0.35	0.86	0.55	**0.09**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	93147	107315	100389	15056	64555	76975	39296	**4902**
f_5	Time (sec)	1.18	1.41	1.41	0.23	0.85	0.98	0.41	**0.09**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	165599	161784	170675	**47637**	-	180352	99795	78117
f_6	Time (sec)	1.57	1.65	1.73	**0.56**	-	1.74	0.72	1.30
	Ratio (%)	80.0	26.7	43.3	**100.0**	0.0	40.0	**100.0**	**100.0**
	Mean FEs	90633	89067	91811	122210	44697	101829	23861	**5159**
f_7	Time (sec)	2.22	1.92	2.02	2.38	0.72	2.21	0.43	**0.12**
	Ratio (%)	56.7	20.0	40.0	66.7	**100.0**	20.0	**100.0**	**100.0**
	Mean FEs	94379	99074	98742	87760	7829	127423	53416	**3531**
f_8	Time (sec)	1.24	1.38	1.31	1.34	0.10	1.67	0.95	**0.08**
	Ratio (%)	96.7	96.7	**100.0**	93.3	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	104987	110115	99480	80260	8293	115247	47440	**2905**
f_9	Time (sec)	1.74	1.99	1.67	1.52	0.14	1.91	0.69	**0.07**
	Ratio (%)	**100.0**	**100.0**	**100.0**	**90.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	110844	125543	118926	**38356**	52516	100000	76646	40736
f_{10}	Time (sec)	1.40	1.81	1.65	**0.62**	0.70	1.27	0.79	0.93
	Ratio (%)	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Mean FEs	111733	125777	117946	42604	34154	97213	81422	**7568**
f_{11}	Time (sec)	1.46	1.86	1.68	0.72	0.48	1.29	0.95	**0.16**
	Ratio (%)	40.0	60.0	46.7	**100.0**	56.7	56.7	**100.0**	66.7
	Mean FEs	99541	107452	102779	**19404**	44491	95830	59160	21538
f_{12}	Time (sec)	2.17	2.57	2.38	**0.50**	0.98	2.10	1.38	0.68
	Ratio (%)	90.0	100.0	96.7	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Mean Reliability		88.62%	83.34%	85.56%	95.83%	88.06%	83.62%	91.67%	**97.23%**

To depict how fast the algorithms reach acceptable solutions, accumulative percentages of the acceptable solutions obtained in each FE are shown in Fig. 12. The figure includes the representative unimodal functions (f_1 and f_4) and the complex multimodal functions (f_7 and f_8). For example, Fig. 12(c) shows that while optimizing the function f_7, we have the following: 1) the APSO, the CLPSO, and the HPSO-TVAC manage to obtain acceptable solutions in all the trials, but the APSO is faster than the CLPSO and the HPSO-TVAC; 2) only about two-thirds of the trails in the GPSO and the FIPS obtain acceptable solutions (with a medium convergence speed); 3) the VPSO succeeds in about 40% of the trials; and 4) the DMS-PSO and the LPSO converge slowest and only succeed in about one-sixth of the trails.

E. Comparisons Using t-Tests

For a thorough comparison, the t-test [53], [58] has also been carried out. Table VIII presents the t values and the P values on every function of this two-tailed test with a significance level of 0.05 between the APSO and another PSO algorithm. Rows "1 (Better)," "0 (Same)," and "−1 (Worse)" give the number of functions that the APSO performs significantly better than, almost the same as, and significantly worse than the compared algorithm, respectively. Row "General Merit" shows the difference between the number of 1's and the number of −1's, which is used to give an overall comparison between the two algorithms. For example, comparing APSO and GPSO, the former significantly outperformed the latter on seven functions (f_2, f_3, f_4, f_6, f_7, f_8, and f_9), does as better as the latter on five functions (f_1, f_5, f_{10}, f_{11}, and f_{12}), and does worse on 0 function, yielding a "General Merit" figure of merit of $7 - 0 = 7$, indicating that the APSO generally outperforms the GPSO. Although it performed slightly weaker on some functions, the APSO in general offered much improved performance than all the PSOs compared, as confirmed in Table VIII.

VI. ANALYSIS OF PARAMETER ADAPTATION AND ELITIST LEARNING

The APSO operations involve an acceleration rate δ in (11) and an elitist learning rate σ in (13). Hence, are these new parameters sensitive in the operations? What impacts do the two operations of parameter adaptation and elitist learning have on the performance of the APSO? This section aims to

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

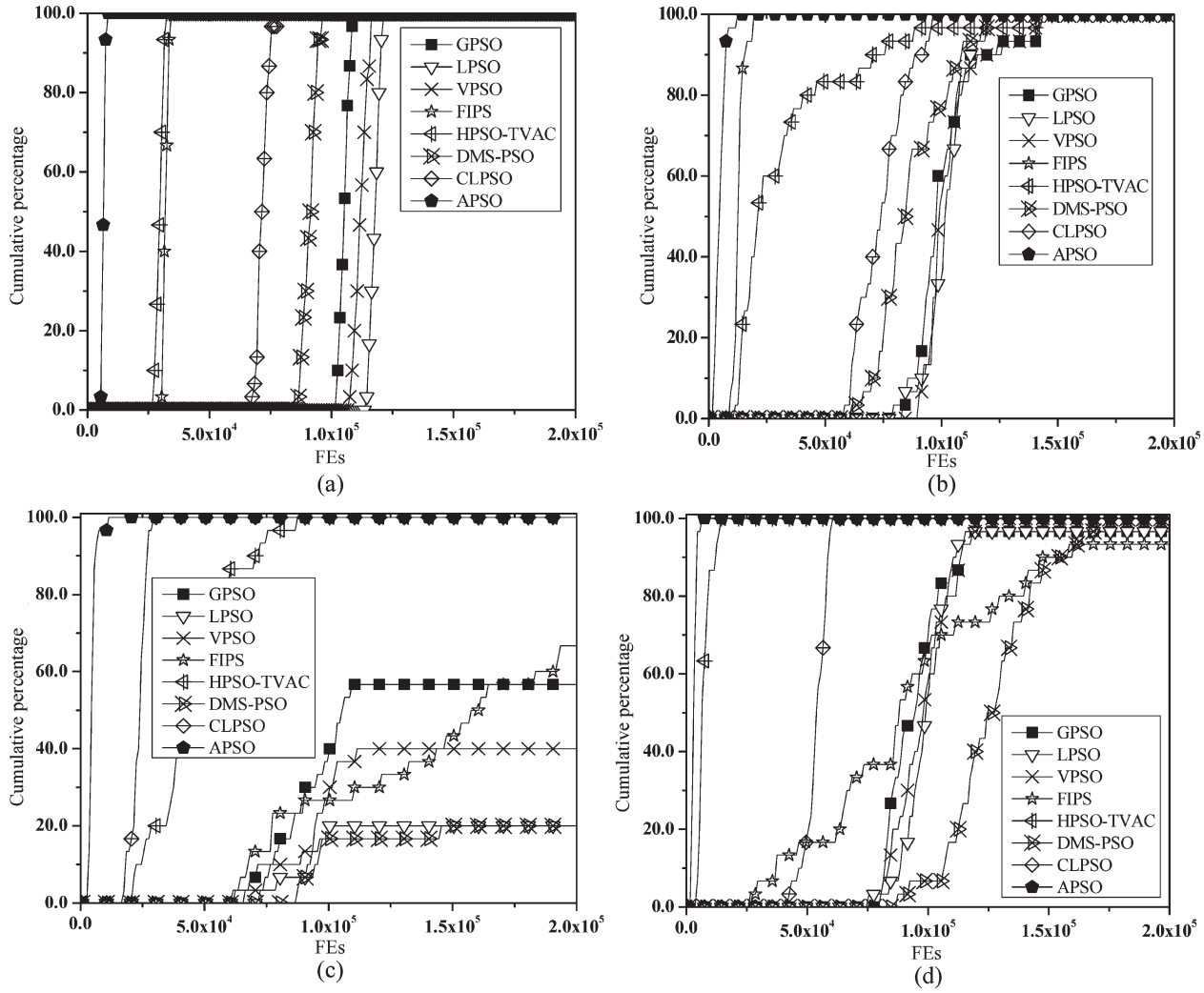


Fig. 12. Cumulative percentages of the acceptable solutions obtained in each FE by the eight PSOs on four test functions. (a) f_1. (b) f_4. (c) f_7. (d) f_8.

answer these questions by further testing the APSO on three unimodal (f_1, f_2, and f_4) and three multimodal (f_7, f_8, and f_{10}) functions.

A. Merits of Parameter Adaptation and Elitist Learning

To quantify the significance of these two operations, the performance of APSO without parameter adaptation or elitist learning is tested under the same running conditions as in Section V. Results of the mean values on 30 independent trials are presented in Table IX.

It is clear from the results that with elitist learning alone and without adaptive control of parameters, the APSO can still deliver good solutions to multimodal functions. However, the APSO suffers from lower accuracy in solutions to unimodal functions. As algorithms can easily locate the global optimal region of a unimodal function and then refine the solution, the lower accuracy may be caused by the slower convergence speed to reach the global optimal region. On the other hand, the APSO with parameter adaptation alone but without ELS can hardly jump out of the local optima and, hence, results in poor performance on multimodal functions. However, it can still solve unimodal problems well.

Note that both of the reduced APSO algorithms generally outperform a standard PSO that involves neither adaptation parameters nor elitist learning. However, the full APSO is the most powerful and robust for any tested problem. This is most evident in the test results on f_4. These results together with the results in Section IV-B confirm the hypothesis that parameter adaptation speeds up the convergence of the algorithm and elitist learning helps the swarm jump out of the local optima and find better solutions.

B. Sensitivity of the Acceleration Rate

The effect of the acceleration rate, which is reflected by its bound δ, on the performance of the APSO is investigated here. For this, the learning rate σ is, hence, fixed (e.g., $\sigma_{\max} = \sigma_{\min} = 0.5$), and the other parameters of the APSO remain the same as in Section V-A. The investigation consists of six test strategies for δ, the first three being to fix its value to 0.01, 0.05, and 0.1, respectively, and the remaining three being randomly to generate its value using a uniform distribution within [0.01, 0.05], [0.05, 0.1], and [0.01, 0.1], respectively. The results are presented in Table X in terms of the mean values of the solutions found in 30 independent trials.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE VIII
COMPARISONS BETWEEN APSO AND OTHER PSOS ON t-TESTS

PSOs / Functions		GPSO	LPSO	VPSO	FIPS	HPSO-TVAC	DMS-PSO	CLPSO
f_1	t-value	1.52851	**2.31098**†	1.4671	**4.88501**†	**2.17917**†	1.20579	**6.93676**†
	P-value	0.13182	**0.02441**	0.14775	**0.00001**	**0.03339**	0.23279	**0.00000**
f_2	t-value	**2.35366**†	**3.85389**†	**3.96641**†	**9.17296**†	**5.48133**†	**2.16682**†	**8.47224**†
	P-value	**0.02200**	**0.00029**	**0.00020**	**0.00000**	**0.00000**	**0.03437**	**0.00000**
f_3	t-value	**3.73355**†	**3.3183**†	**5.08843**†	**4.8526**†	**5.32372**†	**4.60526**†	**15.24494**†
	P-value	**0.00043**	**0.00157**	**0.00000**	**0.00001**	**0.00000**	**0.00002**	**0.00000**
f_4	t-value	**5.57538**†	**8.96094**†	**7.58036**†	**32.85535**†	**3.29589**†	**6.62263**†	**3.00317**†
	P-value	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00168**	**0.00000**	**0.00394**
f_5	t-value	0	0	0	0	0	0	0
	P-value	0	0	0	0	0	0	0
f_6	t-value	**5.76807**†	**9.46838**†	**9.26301**†	**-6.35398**†	**13.34007**†	**8.08689**†	-1.98019
	P-value	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	0.05243
f_7	t-value	**27.42668**†	**35.28576**†	**25.33892**†	**15.12005**†	**32.28794**†	**36.92784**†	1.79505
	P-value	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	0.07786
f_8	t-value	**19.3625**†	**26.38238**†	**23.13992**†	**15.03442**†	**3.52542**†	**24.02031**†	**2.12311**
	P-value	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00083**	**0.00000**	**0.03802**
f_9	t-value	**11.47452**†	**18.03982**†	**12.35315**†	**20.72115**†	**3.7845**†	**27.68977**†	**2.40832**†
	P-value	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00037**	**0.00000**	**0.01923**
f_{10}	t-value	0.46159	**6.73442**†	**3.78212**†	**-5.12379**†	1.19682	**-3.58847**†	**11.89982**†
	P-value	0.64610	**0.00000**	**0.00037**	**0.00000**	0.23624	**0.00068**	**0.00000**
f_{11}	t-value	1.08486	-1.07192	-0.70658	**-3.56237**†	-1.23569	-0.6606	**-3.79146**†
	P-value	0.28247	0.28820	0.48265	**0.00074**	0.22155	0.51148	**0.00036**
f_{12}	t-value	1.79505	1.87465	1	-1.15597	**8.68004**†	-1.61897	**4.51261**†
	P-value	0.07786	0.06588	0.32146	0.25243	**0.00000**	0.11088	**0.00003**
1 (Better)		7	9	8	7	9	7	8
0 (Same)		5	3	4	2	3	4	3
-1 (Worse)		0	0	0	3	0	1	1
General merit over contender		7	9	8	4	9	6	7

†The value of t with 29 degrees of freedom is significant at α=0.05 by a two-tailed test.

TABLE IX
MERITS OF PARAMETER ADAPTATION AND ELITIST LEARNING ON SEARCH QUALITY

Algorithms	APSO With Both Adaptation & Learning		APSO With Only Adaptive Parameters		APSO With Only Elitist Learning Strategy		GPSO (PSO Without Either)	
Functions	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
f_1	1.45×10^{-150}	5.73×10^{-150}	**7.67×10^{-160}**	**3.42×10^{-159}**	3.6×10^{-50}	1.43×10^{-49}	1.98×10^{-53}	7.08×10^{-53}
f_2	5.15×10^{-84}	1.44×10^{-83}	**6.58×10^{-88}**	**2.34×10^{-87}**	2.41×10^{-32}	9.98×10^{-32}	2.51×10^{-34}	5.84×10^{-34}
f_4	**2.84**	**3.27**	13.8879	14.6335	12.7464	18.1979	28.0972	24.5981
f_7	**-12569.5**	5.22×10^{-11}	-7367.77	681.983	**-12569.5**	**3.34×10^{-12}**	-10090.16	495.135
f_8	5.8×10^{-15}	1.01×10^{-14}	52.7327	15.0326	**1.78×10^{-16}**	**5.42×10^{-16}**	30.6779	8.6781
f_{10}	**1.11×10^{-14}**	3.55×10^{-15}	1.0885	1.00363	1.12×10^{-14}	2.64×10^{-15}	1.15×10^{-14}	**2.27×10^{-15}**

It can be seen that APSO is not very sensitive to the acceleration rate δ, and the six acceleration rates all offer good performance. This may be owing to the use of bounds for the acceleration coefficients and the saturation to restrict their sum by (12). Therefore, given the bounded values of c_1 and c_2 and their sum restricted by (12), an arbitrary value within the range [0.05, 0.1] for δ should be acceptable to the APSO algorithm.

C. Sensitivity of the Elitist Learning Rate

To assess the sensitivity of σ in elitist learning, six strategies for setting the value of σ are tested here using three fixed values (0.1, 0.5, and 1.0) and three time-varying values (from 1.0 to 0.5, from 0.5 to 0.1, and from 1.0 to 0.1). All the other parameters of the APSO remain as those in Section V-A. The mean results of 30 independent trials are presented in Table XI.

The results show that if σ is small (e.g., 0.1), then the learning rate is not enough for a long jump out of the local optima, which is evident in the performance on f_7. However, all other settings, which permit a larger σ, have delivered almost the same excellent performance, particularly the strategy with a time-varying σ decreasing from 1.0 to 0.1. It is seen that a smaller σ contributes more to helping the leading particle refine, whereas a larger σ contributes more to helping the leader move away from its existing position so as to jump out of the local optima. This confirms the intuition that long jumps should be accommodated at an early phase to avoid local optima and premature convergence, whereas small perturbations at a latter phase should help refine global solutions, as recommended in this paper.

VII. CONCLUSION

In this paper, PSO has been extended to APSO. This progress in PSO has been made possible by ESE, which utilizes the information of population distribution and relative particle fitness,

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

TABLE X
EFFECTS OF THE ACCELERATION RATE ON GLOBAL SEARCH QUALITY

Value of δ	f_1	f_2	f_4	f_7	f_8	f_{10}
Fixed at 0.01	**6.79×10^{-151}**	**1.23×10^{-83}**	3.4159	-12474.3	2.25×10^{-2}	**1.08×10^{-14}**
Fixed at 0.05	8.3×10^{-149}	2.81×10^{-83}	4.06522	-12317	6.16×10^{-15}	1.14×10^{-14}
Fixed at 0.1	8.03×10^{-149}	6.05×10^{-84}	3.72976	-12153.8	6.63×10^{-2}	1.14×10^{-14}
Random (0.01,0.05)	2×10^{-148}	2.15×10^{-80}	2.6106	-12420.7	0.132661	1.14×10^{-14}
Random (0.05,0.1)	2.62×10^{-150}	6.95×10^{-82}	**1.79069**	**-12475.2**	**8.11×10^{-15}**	1.17×10^{-14}
Random (0.01,0.1)	2.64×10^{-149}	3.12×10^{-83}	3.00886	-12133.6	9.95×10^{-2}	1.11×10^{-14}

TABLE XI
EFFECTS OF THE ELITIST LEARNING RATE ON GLOBAL SEARCH QUALITY

Value of σ	f_1	f_2	f_4	f_7	f_8	f_{10}
Fixed at 0.1	5.16×10^{-152}	1.62×10^{-82}	1.94812	-11622	6.87×10^{-15}	**1.10×10^{-14}**
Fixed at 0.5	6.83×10^{-148}	7.02×10^{-77}	**1.73717**	-12045.9	3.32×10^{-2}	1.05×10^{-7}
Fixed at 1.0	2.07×10^{-149}	**3.39×10^{-83}**	2.7744	-12277.3	9.95×10^{-2}	1.12×10^{-14}
From 1.0 to 0.5	2.85×10^{-148}	5.21×10^{-82}	2.34211	-12263.9	0.132661	1.34×10^{-14}
From 0.5 to 0.1	1.90×10^{-148}	8.83×10^{-82}	2.0236	-12565.5	6.63×10^{-2}	1.21×10^{-14}
From 1.0 to 0.1	**1.24×10^{-152}**	8.71×10^{-83}	3.23075	**-12569.5**	**4.03×10^{-15}**	1.12×10^{-14}

sharing a similar spirit to the internal modeling in evolution strategies. Based on such information, an evolutionary factor is defined and computed with a fuzzy classification method, which facilitates an effective and efficient ESE approach and, hence, an adaptive algorithm.

As shown in the benchmark tests, the adaptive control of the inertia weight and the acceleration coefficients makes the algorithm extremely efficient, offering a substantially improved convergence speed in terms of both number of FEs and CPU time needed to reach acceptable solutions for both unimodal and multimodal functions. Together with an acceleration bound control similar to competitive learning in artificial neural networks, an acceleration rate control is also developed to assist a gradual parameter change in APSO.

Further, a Gaussian perturbation-based ELS is developed to lead the swarm to jump out of any possible local optima and also to refine converging solutions. A time-varying learning rate that shares a similar spirit to neural network training or Boltzmann learning in simulated annealing is developed to further assist the delivery of the two-folded learning goal. The substantially improved global solution accuracy as a result of the ELS is evident in the benchmark tests.

The ESE-based parameter adaptation technique departs from the existing parameter variation schemes, which are passively based on the generation number alone. This technique and the elitist learning technique also make the improved PSO algorithm very reliable in solving both unimodal and multimodal problems, as evident in the t-test results detailed in Table VIII and in the comparisons detailed in Table IX. While the APSO as a whole introduces two new parameters to the PSO paradigm, i.e., the acceleration rate and the learning rate, they are easy to set and add no burden to program design or implementation. Hence, the APSO is still simple and almost as easy to use as the standard PSO, whereas it brings in substantially improved performance in terms of convergence speed, global optimality, solution accuracy, and algorithm reliability.

It is expected that APSO will make an impact on the applications of PSO to real-world optimization and search problems. Further work includes research into adaptive control of topological structures based on ESE and applications of the ESE technique to other evolutionary computation algorithms. Results will be reported in due course.

ACKNOWLEDGMENT

The authors would like to thank the associate editor and reviewers for their valuable comments and suggestions that improved the paper's quality.

REFERENCES

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, Australia, 1995, vol. 4, pp. 1942–1948.
[2] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micromachine Human Sci.*, Nagoya, Japan, 1995, pp. 39–43.
[3] J. Kennedy, R. C. Eberhart, and Y. H. Shi, *Swarm Intelligence*. San Mateo, CA: Morgan Kaufmann, 2001.
[4] R. C. Eberhart and Y. H. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, Korea, 2001, pp. 81–86.
[5] X. D. Li and A. P. Engelbrecht, "Particle swarm optimization: An introduction and its recent developments," in *Proc. Genetic Evol. Comput. Conf.*, 2007, pp. 3391–3414.
[6] R. A. Krohling and L. dos Santos Coelho, "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 6, pp. 1407–1416, Dec. 2006.
[7] N. Franken and A. P. Engelbrecht, "Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma," *IEEE Trans. Evol. Comput.*, vol. 9, no. 6, pp. 562–579, Dec. 2005.
[8] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, "OPSO: Orthogonal particle swarm optimization and its application to task assignment problems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 2, pp. 288–298, Mar. 2008.
[9] B. Liu, L. Wang, and Y. H. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 18–27, Feb. 2007.
[10] R. C. Eberhart and Y. Shi, "Guest editorial," *IEEE Trans. Evol. Comput.—Special Issue Particle Swarm Optimization*, vol. 8, no. 3, pp. 201–203, Jun. 2004.
[11] G. Ciuprina, D. Ioan, and I. Munteanu, "Use of intelligent-particle swarm optimization in electromagnetics," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 1037–1040, Mar. 2002.
[12] 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, Jun. 2006.
[13] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, 1998, pp. 69–73.
[14] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 1999, pp. 1945–1950.
[15] A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Comput. Oper. Res.*, vol. 33, no. 3, pp. 859–871, Mar. 2004.
[16] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2001, vol. 1, pp. 101–106.
[17] A. Ratnaweera, S. Halgamuge, and H. Watson, "Particle swarm optimization with self-adaptive acceleration coefficients," in *Proc. 1st Int. Conf. Fuzzy Syst. Knowl. Discovery*, 2003, pp. 264–268.
[18] 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, Jun. 2004.
[19] T. Yamaguchi and K. Yasuda, "Adaptive particle swarm optimization: Self-coordinating mechanism with updating information," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Taipei, Taiwan, Oct. 2006, pp. 2303–2308.
[20] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, "Adaptive multi-objective particle swarm optimization algorithm," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 2281–2288.
[21] P. J. Angeline, "Using selection to improve particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Anchorage, AK, 1998, pp. 84–89.
[22] Y. P. Chen, W. C. Peng, and M. C. Jian, "Particle swarm optimization with recombination and dynamic linkage discovery," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1460–1470, Dec. 2007.
[23] P. S. Andrews, "An investigation into mutation operators for particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1044–1051.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

[24] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer with local search," in *Proc. IEEE Congr. Evol. Comput.*, 2005, pp. 522–528.
[25] A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in *Proc. Int. Conf. Artif. Intell.*, Las Vegas, NV, 2000, pp. 429–434.
[26] X. Hu and R. C. Eberhart, "Adaptive particle swarm optimization: Detection and response to dynamic systems," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1666–1670.
[27] X. Xie, W. Zhang, and Z. Yang, "Adaptive particle swarm optimization on individual level," in *Proc. Int. Conf. Signal Process.*, 2002, pp. 1215–1218.
[28] M. Clerc, "The swarm and the queen: Toward a deterministic and adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 1999, pp. 1951–1957.
[29] 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, Feb. 2002.
[30] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, Mar. 2003.
[31] K. Yasuda, A. Ide, and N. Iwasaki, "Stability analysis of particle swarm optimization," in *Proc. 5th Metaheuristics Int. Conf.*, 2003, pp. 341–346.
[32] V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 245–255, Jun. 2006.
[33] F. van den Bergh and A. P. Engelbrecht, "A study of particle optimization particle trajectories," *Inf. Sci.*, vol. 176, no. 8, pp. 937–971, Apr. 2006.
[34] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, Korea, 2001, pp. 94–97.
[35] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2000, pp. 84–88.
[36] J. Kennedy, "The particle swarm social adaptation of knowledge," in *Proc. IEEE Int. Conf. Evol. Comput.*, Indianapolis, IN, Apr. 1997, pp. 303–308.
[37] P. N. Suganthan, "Particle swarm optimizer with neighborhood operator," in *Proc. IEEE Congr. Evol. Comput.*, Washington DC, 1999, pp. 1958–1962.
[38] C. F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 997–1006, Apr. 2004.
[39] W. J. Zhang and X. F. Xie, "DEPSO: Hybrid particle swarm with differential evolution operator," in *Proc. IEEE Conf. Syst., Man, Cybern.*, Oct. 2003, pp. 3816–3821.
[40] 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, Jun. 2004.
[41] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 211–224, Jun. 2004.
[42] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "A niching particle swarm optimizer," in *Proc. 4th Asia-Pacific Conf. Simul. Evol. Learn.*, 2002, pp. 692–696.
[43] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "Locating multiple optima using particle swarm optimization," *Appl. Math. Comput.*, vol. 189, no. 2, pp. 1859–1883, Jun. 2007.
[44] D. Parrott and X. D. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 440–458, Aug. 2006.
[45] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1671–1676.
[46] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 36, no. 4, pp. 515–519, Jul. 2006.
[47] X. Hu and R. C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, 2002, pp. 1677–1681.
[48] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proc. Swarm Intell. Symp.*, Jun. 2005, pp. 124–129.
[49] 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, Jun. 2004.
[50] J. Zhang, H. S.-H. Chung, and W.-L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 11, no. 3, pp. 326–335, Jun. 2007.
[51] Z. H. Zhan, J. Xiao, J. Zhang, and W. N. Chen, "Adaptive control of acceleration coefficients for particle swarm optimization based on clustering analysis," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, Sep. 2007, pp. 3276–3282.
[52] A. Carlisle and G. Dozier, "An off-the-shelf PSO," in *Proc. Workshop Particle Swarm Optimization*. Indianapolis, IN: Purdue School Eng. Technol., Apr. 2001, pp. 1–6.
[53] X. Yao, Y. Liu, and G. M. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.
[54] 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," in *Proc. IEEE Congr. Evol. Comput.*, 2005, pp. 1–50.
[55] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing*. Englewood Cliffs, NJ: Prentice-Hall, 1997.
[56] K. Deb and H. G. Beyer, "Self-adaptive genetic algorithms with simulated binary crossover," *Evol. Comput.*, vol. 9, no. 2, pp. 197–221, Jun. 2001.
[57] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[58] G. E.-P. Box, J. S. Hunter, and W. G. Hunter, *Statistics for Experiments: Design, Innovation, and Discovery*, 2nd ed. New York: Wiley, 2005.

Zhi-Hui Zhan (S'09) received the Bachelor's degree in computer science and technology in 2007 from Sun Yat-Sen University, Guangzhou, China, where he is currently working toward the Ph.D. degree.

His current research interests include particle swarm optimization, genetic algorithms, ant colony optimization, and other evolutionary computation techniques.

Jun Zhang (M'02–SM'08) received the Ph.D. degree in electrical engineering from the City University of Hong Kong, Kowloon, Hong Kong, in 2002.

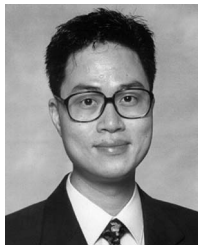
From 2003 to 2004, he was a Brain Korean 21 Postdoctoral Fellow with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea. Since 2004, he has been with the Sun Yat-Sen University, Guangzhou, China, where he is currently a Professor in the Department of Computer Science. He is the author of four research book chapters and over 60 technical papers. His research interests include genetic algorithms, ant colony system, PSO, fuzzy logic, neural network, nonlinear time series analysis and prediction, and design and optimization of power electronic circuits.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.

Yun Li (S'87–M'89) received the B.Sc. degree in radio electronics science from Sichuan University, Chengdu, China, in 1984, the M.Sc. degree in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, in 1987, and the Ph.D. degree in computing and control engineering from the University of Strathclyde, Glasgow, U.K., in 1990.

From 1989 to 1990, he was with the U.K. National Engineering Laboratory and the Industrial Systems and Control Limited, Glasgow. In 1991, he was a Lecturer with the University of Glasgow, Glasgow. In 2002, he was a Visiting Professor with Kumamoto University, Kumamoto, Japan. He is currently a Senior Lecturer with the University of Glasgow, and a Visiting Professor with UESTC. In 1996, he independently invented the "indefinite scattering matrix" theory, which opened up a groundbreaking way for microwave feedback circuit design. From 1987 to 1991, he carried out a leading work in parallel processing for recursive filtering and feedback control. In 1992, he achieved first symbolic computing for power electronic circuit design without needing to invert any matrix, complex-numbered or not. Since 1992, he has pioneered into the design automation of control systems and the discovery of novel systems using evolutionary learning and intelligent search techniques. He established the IEEE CACSD Evolutionary Computation Working Group and the European Network of Excellence in Evolutionary Computing (EvoNet) Workgroup on Systems, Control, and Drives in 1998. He has produced 12 Ph.D. degrees in this area and has over 140 publications.

Dr. Li is a Chartered Engineer and a member of the Institution of Engineering and Technology.



Henry Shu-Hung Chung (S'92–M'95–SM'03) received the B.Eng. degree in electrical engineering and the Ph.D. degree from the Hong Kong Polytechnic University, Kowloon, Hong Kong, in 1991 and 1994, respectively.

Since 1995, he has been with the City University of Hong Kong (CityU), Kowloon, where he is currently a Professor in the Department of Electronic Engineering and the Chief Technical Officer of e.Energy Technology Limited—an associated company of CityU. He is the holder of ten patents. He has authored four research book chapters and over 250 technical papers including 110 refereed journal papers in his research areas. His research interests include time- and frequency-domain analysis of power electronic circuits, switched-capacitor-based converters, random-switching techniques, control methods, digital audio amplifiers, soft-switching converters, and electronic ballast design.

Dr. Chung was awarded the Grand Applied Research Excellence Award in 2001 by the City University of Hong Kong. He was IEEE Student Branch Counselor and was the Track Chair of the technical committee on power electronics circuits and power systems of the IEEE Circuits and Systems Society from 1997 to 1998. He was an Associate Editor and a Guest Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART I: FUNDAMENTAL THEORY AND APPLICATIONS from 1999 to 2003. He is currently an Associate Editor of the IEEE TRANSACTIONS ON POWER ELECTRONICS and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART I: FUNDAMENTAL THEORY AND APPLICATIONS.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on September 27, 2009 at 06:57 from IEEE Xplore. Restrictions apply.