Genetic Learning Particle Swarm Optimization

Yue-Jiao Gong, *Member, IEEE*, Jing-Jing Li, Yicong Zhou, *Senior Member, IEEE*, Yun Li, *Member, IEEE*, Henry Shu-Hung Chung, *Senior Member, IEEE*, Yu-Hui Shi, *Senior Member, IEEE*, and Jun Zhang, *Senior Member, IEEE*

***Abstract*—Social learning in particle swarm optimization (PSO) helps collective efficiency, whereas individual reproduction in genetic algorithm (GA) facilitates global effectiveness. This observation recently leads to hybridizing PSO with GA for performance enhancement. However, existing work uses a mechanistic parallel superposition and research has shown that construction of superior exemplars in PSO is more effective. Hence, this paper first develops a new framework so as to organically hybridize PSO with another optimization technique for "learning." This leads to a generalized "learning PSO" paradigm, the *L-PSO. The paradigm is composed of two cascading layers, the first for exemplar generation and the second for particle updates as per a normal PSO algorithm. Using genetic evolution to breed promising exemplars for PSO, a specific novel *L-PSO algorithm is proposed in the paper, termed genetic learning PSO (GL-PSO). In particular, genetic operators are used to generate exemplars from which particles learn and, in turn, historical search information of particles provides guidance to the evolution of the exemplars. By performing crossover, mutation, and selection on the historical information of particles, the constructed exemplars are not only well diversified, but also high qualified. Under such guidance, the global search ability and search efficiency of PSO are both enhanced. The proposed GL-PSO is tested on 42 benchmark functions widely adopted in the literature. Experimental results verify the effectiveness, efficiency, robustness, and scalability of the GL-PSO.**

***Index Terms*—Exemplar construction, genetic algorithm (GA), hybrid method, learning scheme, particle swarm optimization (PSO).**

Manuscript received December 12, 2014; revised June 23, 2015; accepted August 21, 2015. This work was supported in part by the National High-Technology Research and Development Program (863 Program) of China under Grant 2013AA01A212, in part by the National Science Fund for Distinguished Young Scholars under Grant 61125205, and in part by the National Natural Science Foundation of China under Grant 6120002 and Grant 61502542. This paper was recommended by Associate Editor L. Zhang. *(Corresponding author: Jing-Jing Li.)*

Y.-J. Gong and J. Zhang are with the Department of Computer Science, Sun Yat-sen University, Guangzhou 510275, China, also with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Guangzhou, China, and also with the Engineering Research Center of Supercomputing Engineering Software, Ministry of Education, Guangzhou 510006, China, (e-mail: junzhang@ieee.org).

J.-J. Li is with the School of Computer Science, South China Normal University, Guangzhou 510006, China.

Y. Zhou is with the Department of Computer and Information Science, University of Macau, Macau 999078, China.

Y. Li is with the School of Engineering, University of Glasgow, Glasgow G12 8QQ, U.K.

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

Y.-H. Shi is with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China.

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors.

Digital Object Identifier 10.1109/TCYB.2015.2475174

I. Introduction

PARTICLE swarm optimization (PSO), a nature-inspired optimization technique, has attracted significant attention since introduced by Kennedy and Eberhart [1], [2] in 1995. Simulating the social behavior of birds flocking or fish schooling, a population of particles in a PSO algorithm cooperates and interacts to search for solutions in the problem space. Owing to its conceptual simplicity and high efficiency, PSO has been successful in solving a variety of problems in many areas such as power systems [3], [4], industrial electronics [5], wireless sensor networks [6], and feature selection [7].

However, in canonical PSO, all particles keep learning from the personal best experience (pbests) and the global best-so-far solution (gbest) of the entire swarm, which may lead to premature convergence. To improve the performance, a number of PSO variants have been developed during the past two decades. These variants can be generally divided into four categories that focus on: 1) population topology and multiswarm techniques [8]–[10]; 2) parameter control [11]–[14]; 3) hybrid methods [15]–[25]; and 4) novel learning schemes [26]–[30], respectively. Although much effort has been made to enhance the performance of PSO, many of the variants cannot maintain their improvements over problems of different characteristics. For example, some PSO variants are able to increase population diversity and avoid premature convergence, but their search speed and solution accuracy are decreased as a result. On the other hand, some other algorithms still cannot locate the global optimum for difficult problems involving many local optima. So far, it has been a challenging task to improve the overall performance of PSO for wider applicability.

In nature, it is widely accepted that the behavior of organisms, such as bee foraging and scouting [31], and bird migration [32], is influenced by genetic information. A discipline named "behavioral genetics" is dedicated to discussing how such genetic control is executed [33], [34]. In particular, Raine *et al.* [35] discovered that the animals' foraging behavior depends primarily on its evolutionary history rather than the current foraging condition. Alcock [36] put forward a viewpoint that behavioral variation is partially derived from genotypic variation, and therefore, it is an evolutionary process. On the other hand, the skills acquired by animals in social activities affect the natural selection force on the animals and hereafter alter the genes in the population [37], [38]. As biological experiments reveal that the behavior and genetic information of organisms influence each other, it would be

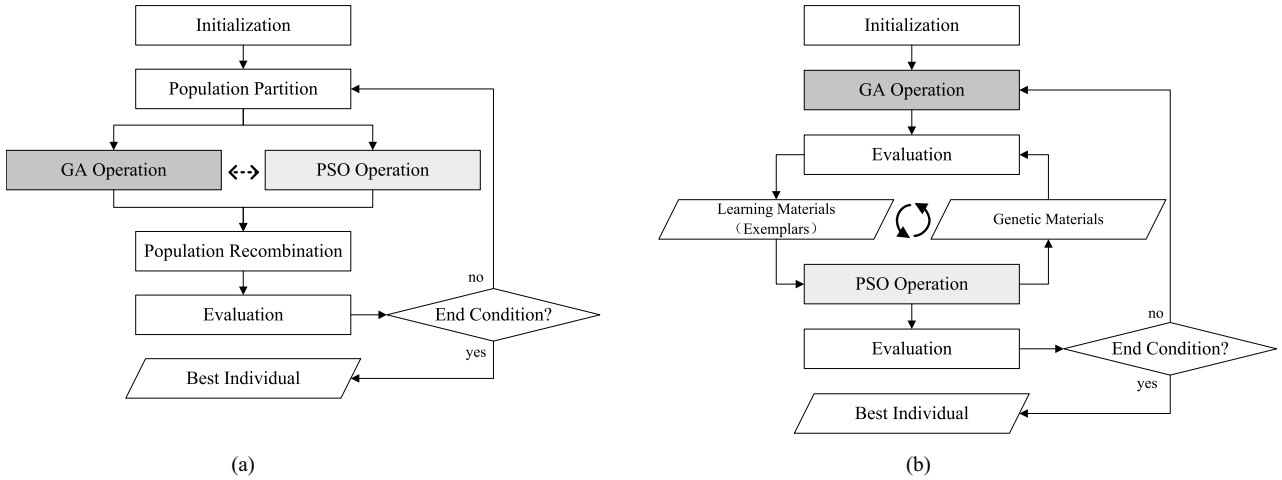


Fig. 1. (a) Parallel and (b) cascade frameworks of GA-PSO hybrid algorithms.

possible and interesting to utilize such interaction between the behavior and genes to enhance the performance of the biologically inspired PSO technique.

In the literature, a number of work have been reported in improving the performance of PSO by adopting genetic methods, which can be categorized into the following two types. First, some PSO algorithms embed a specific genetic operator of genetic algorithm (GA) such as mutation [39] and crossover [40]. Second, significant efforts are made in fully hybridization of PSO and GA. For example, Shi *et al.* [15] proposed a PSO-GA-based hybrid algorithm by executing the two algorithms simultaneously. Then, Kao and Zahara [16] hybridized PSO with GA based on a fitness ranking method. Valdez *et al.* [17] integrated GA and PSO by using a fuzzy logic method for decision-making. Three different hybrid mechanisms are proposed in [18], the first being to run GA and PSO in parallel and exchange their information via crossover between chromosomes and the gbest particle, the second being to apply the mutation of GA on stagnated particles, and the third being to divide the number of iterations to run GA and PSO in complement. Furthermore, GA-PSO hybrid algorithms have been developed to tackle some *ad hoc* applications, including recurrent network design [19], electromagnetic structure optimization [20], and conformal array pattern synthesis [21]. More recently, Jeong *et al.* [22] proposed an efficient GA-PSO hybrid algorithm for real-world multiobjective problems.

Most of these hybrid algorithms share conceptual and operational similarities in that they divide the population into two parallel subpopulations, each controlled by GA and PSO separately, and then recombine the subpopulations at set intervals. A general framework of those algorithms is illustrated in Fig. 1(a), which shows that GA and PSO are hybridized in a parallel manner. In this hybridization mechanism, GA and PSO are loosely coupled, and the effect coming from the interaction of GA and PSO is ambiguous to recognize. For example, it is hard to identify how the individuals exchanged from GA affect the PSO population and whether they will result in updating the pbests and gbest. Inversely, it is also difficult to characterize how the individuals transferred from PSO influence the GA population, especially whether the migrated individuals will survive in the selection. More critically, the benchmark-testing results of PSO-GA hybrid algorithms reported in the literature are inconclusive [15]–[17]. In particular, the algorithms are tested on low-dimensional problems (no more than 10-D in most cases), and their performance lies between that of PSO and GA in optimizing many benchmarks. It can be seen that hybridizing PSO with GA to achieve the perfect goal of "$1 + 1 > 2$" still remains a challenging task.

To further improve the performance of PSO, this paper develops a genetic learning scheme that applies GA for exemplar construction. As illustrated in Fig. 1(b), GA and PSO are hybridized in a cascade manner. The main loop of the algorithm is composed of two cascading layers, the first for exemplar generation by GA and the second for particle updates as per a normal PSO algorithm. In this way, particles in PSO are no longer simply guided by the gbest and pbests, but are guided by the exemplars constructed by GA. GA and PSO are hybridized in a highly cohesive way, which establishes a positive feedback loop to accelerate the population to locate the optimum.

1) By learning from the exemplars constructed from GA, the search of particles is more diversified, thus helping avoid the premature convergence of PSO. Moreover, owing to the effect of the selection operator in GA, the survived exemplars are also of high quality. They are capable of providing effective guidance for particles and hence improving the search efficiency of PSO.
2) In turn, the search experience (pbests and gbest) of particles propagates promising genetic materials back to GA and helps GA reproduce improved exemplars.

Therefore, in this cascade architecture, the effect of the interaction between PSO and GA is clear and the interaction enhances PSO and GA alternatively in the optimization process. In addition, as the cascade hybrid mechanism is divided into two layers, a different learning approach can be used in the upper layer to design a different hybrid PSO algorithm,

providing an open opportunity to improve the performance of PSO in the future.

The rest of this paper is organized as follows. Section II reviews the technical and biological backgrounds on hybridizing PSO with GA. Section III presents the GL-PSO algorithm with implementation details and analysis, followed by a generalized hybrid paradigm in Section IV. Experimental tests are carried out in Section V. Finally, conclusions are drawn in Section VI.

II. Backgrounds

A. Technical Background: GA Versus PSO

Currently, GA and PSO are two well-known branches existing in the field of bio-inspired optimization. A typical GA consists of three basic operators: 1) selection; 2) crossover; and 3) mutation. The selection operator duplicates higher-quality chromosomes to pass on for improving the average fitness values of the population, also known as evolution. Crossover and mutation are reproductive operators that provide the evolving population with alternative but probably higher-quality genetic materials. In contrast, a PSO algorithm does not employ the selection operator, but the evolution is represented by updating particles toward historically best positions. The flying trajectories of particles in PSO can correspond to the changing genetic materials in GA.

The differences in the mechanisms of GA and PSO often result in the differences in their performance. In canonical GA, two chromosomes are randomly picked to exchange their component genes through crossover, and some genes in the chromosomes are randomly varied through mutation. Therefore, the reproduction process of GA is, to some extent, omnidirectional. On the other hand, in canonical PSO, as particles are guided by their previous best positions (pbests) and the global best position (gbest) found by the swarm, the search is more directional than that of GA. Hence, it is expected that GA possesses a better exploration ability than PSO whereas the latter facilitates faster convergence.

In Section S-I of the supplementary file of this paper, experiments have been carried out on unimodal and multimodal benchmark functions to compare GA and PSO, the results of which verify the above analysis. For the unimodal function, owing to its efficient search mechanism, PSO is seen to quickly converge to the optimum and to obtain high solution accuracy. However, for the multimodal function, when the pbests fall into a local optimum far from the global optimum in the landscape, particles are seen difficult to jump out of the local optimum. This is why PSO performs poorly in optimizing the multimodal function in the experiment. On the contrary, GA owns a much better global search ability that it is capable of locating the global optimum of the multimodal function. However, as the crossover and mutation are relatively directionless, the GA population approaches an optimum more slowly, which leads to the insufficient performance of GA on both the two kinds of problems.

To summarize, GA and PSO have their respective merits and demerits. The issue remaining is how to utilize the merits of both to enhance the overall performance, i.e., how to improve the global search ability of PSO by incorporating GA mechanisms without slowing down the search?

B. Biological Background: Gene–Behavior Interaction

From a biological perspective, GA changes the phenotypes of individuals via changing their genotypes, which is a long-term process. Unlike GA, PSO adjusts the phenotypes of particles based on interactive learning activities, which is short-term. This can also be interpreted as the reason why PSO converges faster than GA.

However, in nature, the behavioral traits of organisms are under genetic control. This perspective was firstly proposed by Galton [41] in 1869. After a century of verification and development, the branch of science to explore and formulate the relationships between genes and behavior is recognized as a research discipline termed "behavioral genetics" by Fuller and Thompson [42] in 1960. Since then, Galton has been considered as the first behavioral geneticists. Nowadays, it has become a widely accepted concept that most behavior of organisms are covary with both genotypes and environment [33], [34]. For example, Robinson and Page [31] proposed that genotype distinguishes the nectar foraging, pollen foraging, and nest-site scouting behavior of honey bee colonies. The work in [32] shows that the blackcap's migratory motivation and direction are under genetic control. Raine *et al.* [35] put forward that the animals' foraging behavior could be better explained by its evolutionary history, rather than their present foraging conditions. The book entitled *Animal Behaviour—An Evolutionary Approach* presents many examples showing that a proportion of phenotypic variation in behavior is derived from genotypic variation.

Inversely, Sterelny [37] developed a perspective that learning has evolutionary consequences. Although the skills acquired by animals cannot be directly encoded into their genes, the variations in animals' phenotype affect the natural selection force acting on the animals and hereafter alter the genes in the animal population generation by generation. For instance, Jones *et al.* [38] found that some woodpeckers learn to use cactus spines to harvest food in tree trunk, and natural selection is more likely to preserve those individuals in the population whose beak shape facilitates manipulating spines. Afterward, more and more birds not only acquire the feeding skill but also inherit the specific beak shape.

To conclude, in nature, the behavior of organisms, such as foraging and migration, is under genetic control, and, inversely, their acquired skills via social activities (such as learning) would finally play roles in genotypic variations. In brief, the genes and behavior of organisms interact with each other. As PSO is a kind of nature-inspired technique, this motivates us to incorporate some genetic mechanism into the learning and search process of the algorithm.

III. Genetic Learning PSO

A. Canonical PSO Learning Scheme and its Variants

In canonical PSO, each particle learns from its own pbest and the gbest found by the entire swarm in order to update velocity and position. Let $V_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,D}]$ and

$X_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}]$ represent the velocity and position of the ith particle ($i = 1, 2, \ldots, M$, where M is the population size), respectively. Let $P_i = [p_{i,1}, p_{i,2}, \ldots, p_{i,D}]$ denotes the pbest of particle i and $G = [g_1, g_2, \ldots, g_D]$ denotes the gbest of the whole swarm. The update equations for the dth dimension ($d = 1, 2, \ldots, D$) of particle i are defined as

$$v_{i,d} \leftarrow \omega \cdot v_{i,d} + c_1 \cdot r_{1,d} \cdot \left(p_{i,d} - x_{i,d}\right) + c_2 \cdot r_{2,d} \cdot \left(g_d - x_{i,d}\right) \quad (1)$$

$$x_{i,d} \leftarrow x_{i,d} + v_{i,d} \quad (2)$$

where ω denotes the inertia weight, c_1 and c_2 are accelerate coefficients determining the relative importance of P_i and G, and $r_{1,d}$ and $r_{2,d}$ are random numbers uniformly selected within [0, 1].

With this canonical learning scheme, a particle learns from both P_i and G and may hence oscillate if these two exemplars locate on opposite sides of X_i. On the other hand, if P_i and G are located in a same local optimum, the particle may be trapped in this optimum, which causes premature convergence.

In order to overcome the above shortcomings, researchers have developed variant PSO learning schemes, among which the following velocity update is widely adopted [26]–[28]:

$$v_{i,d} \leftarrow \omega \cdot v_{i,d} + c \cdot r_d \cdot \left(e_{i,d} - x_{i,d}\right). \quad (3)$$

Here, replacing two exemplars P_i and G, a single composite exemplar $E_i = [e_{i,1}, e_{i,2}, \ldots, e_{i,D}]$ is constructed in phenotype combination to attract particle i. In fact, the canonical velocity update scheme in (1) can be transformed into this condensed form by regarding $e_{i,d}$ in (3) as a linear combination of $p_{i,d}$ and g_d, as according to [26]

$$e_{i,d} = \frac{c_1 \cdot r_{1,d} \cdot p_{i,d} + c_2 \cdot r_{2,d} \cdot g_d}{c_1 \cdot r_{1,d} + c_2 \cdot r_{2,d}}. \quad (4)$$

For examples, a fully informed particle swarm (FIPS) algorithm [26] guilds a particle to learn from all its neighbors with the exemplar vector being the linear combination of all the pbests in the neighborhood. In comprehensive learning PSO (CLPSO) [27], $e_{i,d}$ is set to $p_{i,d}$ within a predefined probability range, or set to the dth dimension of another particle's pbest via a tournament selection. In orthogonal learning PSO (OLPSO) [28], it conducts orthogonal experimental design on P_i and G in order to construct E_i with orthogonal combination.

B. Genetic Learning Scheme

Because the exemplar vector E_i determines the search trajectories of particles, it plays a significant role in PSO. In this paper, aiming for a more promising vector than using linear or orthogonal combination, a genetic learning scheme that applies GA to breed exemplars is proposed and developed. The motivation of this paper is described as follows. First, canonical PSO algorithm simply uses P_i and G to guide search, which induces premature phenomenon easily. As described in Section II-A, compared to PSO, GA exhibits a better global search ability. So that it is expected that the genetic operators can bring diversity to the exemplar vectors to discourage the premature convergence of PSO. Moreover, the exemplar built by GA is not only well diversified but also with high quality, which is capable of providing a good guidance for particles and hence improving the efficiency of PSO. Furthermore, from a biological point of view, the foraging behavior of birds in nature is under genetic control. It is reasonable to incorporate such a genetic learning scheme into a PSO algorithm. In the following, the detailed implementation of the genetic learning scheme is presented.

1) Crossover: For each particle i, crossover operation is first conducted on P_i and G to generate an offspring $O_i = [o_{i,1}, o_{i,2}, \ldots, o_{i,D}]$

$$o_{i,d} = \begin{cases} r_d \cdot p_{i,d} + (1 - r_d) \cdot g_d, & \text{if } f(P_i) < f\left(P_{k_d}\right) \\ p_{k_d,d}, & \text{otherwise} \end{cases} \quad (5)$$

where r_d is a random number uniformly distributed in [0, 1], $k_d \in \{1, 2, \ldots, M\}$ is the index of a random particle, $i = 1, 2, \ldots, M$, and $d = 1, 2, \ldots, D$. Here, without loss of generality, the considered objective f is for minimization.

This way, a good particle is more likely to perform the arithmetic crossover between its pbest position P_i and the gbest position G. By integrating the information of the global best particle, the performance of O_i has potential to further improve. On the contrary, if particle i is an inferior one in the swarm, the offspring will have more dimensions coming from that of another particle with better fitness. Instead of randomly picking two individuals in the population to undergo crossover as in traditional GA, the above crossover utilizes the historical search experience of particles in PSO to improve the gene quality. An experimental discussion of the crossover operation is provided in Section S-II of the supplementary file, which shows that the operator helps improving search efficiency.

2) Mutation: The bred offspring O_i then undergoes the mutation operation with a probability bounded by probability of mutation (pm). This is the same as the classical mutation operation of GA, which is very simple. For each dimension d, a random number $r_d \in [0, 1]$ is generated, and then, if r_d is smaller than pm, this dimension of O_i is reinitialized in the search space

$$o_{i,d} = \text{rand}(lb_d, ub_d), \text{ if } r_d < pm \quad (6)$$

where lb_d and ub_d stand for the lower and upper bounds of the dth dimension. The mutation brings in diversity of the exemplar to reach more exploratory coordinates (theoretically, any coordinates) in the search space.

3) Selection: After applying crossover and mutation to create the offspring, selection is performed to determine whether the offspring or the current exemplar survives in this generation. The following operation is executed:

$$E_i \leftarrow \begin{cases} O_i, & \text{if } f(O_i) < f(E_i) \\ E_i, & \text{otherwise.} \end{cases} \quad (7)$$

Note that, the calculation of $f(O_i)$ consumes the number of function evaluations (FEs) in the new algorithm.

As shown in (7), the exemplar remains unchanged if it is better than the new offspring. This elitism ensures that the exemplar evolves in every generation and never deteriorates. Moreover, if the exemplar of a particle ceases improving for

a certain number, a stopping gap *sg*, of generations, the exemplar may be considered trapped in a deep local optimum. In this case, we employ the 20%M-tournament selection to update the particle's exemplar. Here the tournament size is empirically set to a value proportional to the population size M. That is, 20%M exemplars are chosen at random to join the tournament, where the winner with the best fitness replaces the particle's current exemplar. By learning from another exemplar, the particle is able to change the search direction abruptly so as to fly out of the local optimum.

For each particle, in summary, it conducts the crossover, mutation, and selection described above once in every generation to construct a promising exemplar and hereafter learns from the exemplar as per the traditional PSO. With the genetic breeding scheme, a novel genetic learning PSO (GL-PSO) algorithm is developed. The pseudo code of GL-PSO is shown in **Algorithm 1** and its framework is sketched in Fig. 1(b). It can be observed that GL-PSO is relatively easy to implement. For public use, we provide the source code of GL-PSO online, which can be downloaded at [43].

Note that the proposed GL-PSO algorithm maintains the canonical framework of PSO, in which particles update their velocities and positions in the search space by utilizing historical search experience. The novelty of GL-PSO lies in that it applies GA to process the pbests and gbest of particles so as to breed the exemplars, which captures the historical search experience with a more global prospective. Particles can hence learn with a good diversity and a high quality, for potentially improved exploration and exploitation abilities. It is to be pointed out that the fundamental component of the proposed algorithm is PSO but rather than GA, and GA is used as an auxiliary technique.

Algorithm 1 Genetic Learning PSO Algorithm (GL-PSO)

```
/* Initialization */
for i = 1 to M do
    Randomly initialize V_i and X_i;
    Evaluate f(X_i);
    P_i = X_i;
end for
Set G to the current best position of particles;

/* Main Loop*/
repeat
    for i = 1 to M do
        /* Exemplar Update: Crossover */
        for d = 1 to D do
            Randomly select a particle k ∈ {1, 2, ..., M};
            if f(P_i) < f(P_k) then
                o_{i,d} = r_d · p_{i,d} + (1 − r_d) · g_d;
            else
                o_{i,d} = p_{k,d};
            end if
        end for
        /* Exemplar Update: Mutation */
        for d = 1 to D do
            if rand(0, 1) < pm then
                o_{i,d} = rand(lb_d, ub_d);
            end if
        end for
        /* Exemplar Update: Selection */
        Evaluate f(O_i);
        if f(O_i) < f(E_i) then
            E_i = O_i
        end if
        if f(E_i) ceases improving for sg generations then
            Select E_j by 20%M tournament;
            E_i = E_j;
        end if

        /* Particle Update */
        for d = 1 to D do
            v_{i,d} = ω · v_{i,d} + c · r_d · (e_{i,d} − x_{i,d});
            x_{i,d} = x_{i,d} + v_{i,d}
        end for
        Evaluate f(X_i);
        Update P_i and G;
    end for
until Terminal Condition
```

C. Complexity Analysis of GL-PSO

The computational costs of the canonical PSO algorithm involve the initialization (T_{ini}), evaluation (T_{eva}), and velocity and position update (T_{upd}) for each particle. Assume D is the dimensionality of the search space and *MaxFEs* is the maximum number of FEs allowed for the algorithm. The time complexity of PSO can be estimated as $T(D) = T_{\text{ini}} + (T_{\text{eva}} + T_{\text{upd}}) \cdot MaxFEs = D + (D+2 \cdot D) \cdot MaxFEs = D \cdot (1+3 \cdot MaxFEs)$. Therefore, O ($D \cdot MaxFEs$) is the time complexity of the canonical PSO. As the parameter *MaxFEs* is commonly set to $10\,000 \cdot D$ in the literature, the canonical PSO algorithm has a time complexity quadratic to the problem size D.

In GL-PSO, the time complexity is determined by the computational costs of the PSO operation (T_{PSO}) and the GA operation (T_{GA}). Here T_{GA} consists of the computational costs of crossover (T_{cro}), mutation (T_{mut}), and selection (T_{sel}). In the worst cases, we have $T_{\text{cro}} = D, T_{\text{mut}} = D$, and $T_{\text{sel}} = 1 + 0.2 \cdot M$. Besides, as PSO and GA both consume evaluation times, the maximum number of iterations for the update of a particle and an exemplar is both (*MaxFEs*/2). Therefore, the worst-case time complexity of GL-PSO can be calculated as $T(D) = T_{\text{ini}} + (T_{\text{PSO}} + T_{\text{GA}}) \cdot (MaxFEs/2) = T_{\text{ini}} + [(T_{\text{eva}} + T_{\text{upd}}) + (T_{\text{eva}} + T_{\text{cro}} + T_{\text{mut}} + T_{\text{sel}})] \cdot (MaxFEs/2) = D + (3 \cdot D + 0.1 \cdot M + 0.5)\ MaxFEs$. Therefore, the GL-PSO algorithm has an $O[(D+M) \cdot MaxFEs]$ time complexity, also linear to *MaxFEs*. Still, the time complexity is quadratic to the problem size D.

D. Search Behavior of GL-PSO

The unimodal Sphere function and multimodal Schwefel function, which have been introduced in Section S-I of the supplementary file, are used in the investigation of GL-PSO performance. Fig. 2 plots the convergence curves of a typical run,

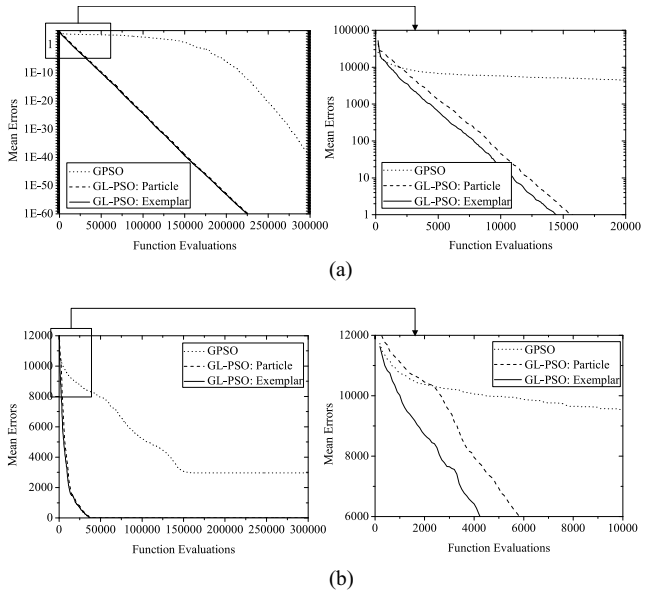


Fig. 2. Search behavior of GL-PSO and its comparison with that of canonical PSO. (a) Relationship between exemplar and particle on Sphere problem. (b) Relationship between exemplar and particle on Schwefel's problem 2.26.

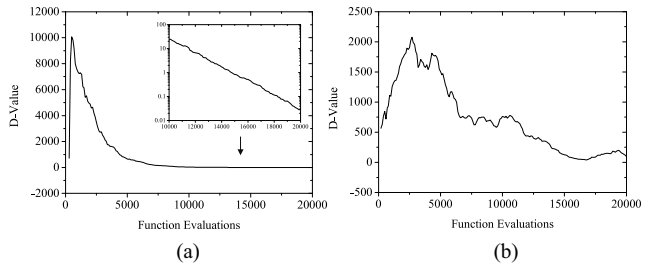


Fig. 3. Mean difference of the fitness of exemplars and particles in GL-PSO. (a) *D*-value on Sphere problem. (b) *D*-value on Schwefel's problem 2.26.

where the solid and dashed lines refer to the average fitness values of exemplars and particles, respectively. Meanwhile, the convergence curve of the average fitness value obtained by the canonical PSO with global topology (GPSO) is represented by dotted line in Fig. 2 for the purpose of comparison. Moreover, we define a *D*-value as the mean difference between the fitness of exemplars and particles in GL-PSO, which is depicted in Fig. 3.

Generally, the exemplars generated in the genetic learning scheme are capable of providing good guidance for the particles, as observed in Figs. 2 and 3. No matter the function is unimodal or multimodal, the fitness values of exemplars are better than those of particles during the optimization process.

At an early stage of the swarm search, the randomly initialized population covers a relatively broad range in the solution space. At this time, the crossover performed on pbests and gbest has an exploration effect in that it constructs an exemplar far away from the particle's current pbest. Fig. 3 shows that the fitness difference between exemplars and particles increases to a large value at this stage. Afterward, with the convergence of the population, pbests and the gbest become geographically close. The crossover operation of GL-PSO tends to have an exploitation effect because the constructed exemplars locate in the vicinity of particles. As seen in Fig. 3, the fitness difference between exemplars and particles begins to decrease after the initial time period.

The effect of mutation is more prominent in optimizing multimodal functions. As shown in Fig. 3, the curve of *D*-value on Schwefel function is not as smooth as that on Sphere function, which contains many extreme points. As particles gradually fly toward their exemplars, the local minimum points on the *D*-value curve imply the moment that exemplars are jumping out of a local optimum. This owes to the mutation operation embedded in the genetic learning scheme, which injects diversified information into the exemplars even when the population has converged. In Fig. 2(b), the curve of GPSO slows down after 2000 FEs and then flattens out to a horizontal course after 150 000 FEs. On contrast, owing to the effective and efficient guidance of exemplars, GL-PSO exhibits promising exploration and exploitation abilities from beginning to end, as shown in Fig. 2(b).

The enhanced performance of the genetic learning scheme also benefits from the selection operation. First let us look at a shortcoming of the canonical learning scheme. As described in Section III-A, the velocity update rule in canonical PSO can be transformed into a composite form (3) by defining the exemplar as a linear combination of pbest and gbest in (4). Consider a 2-D Sphere function whose optimum is [0, 0]. If particle i's pbest is $P_i = [6, 5]$ and the gbest is $G = [2, 2]$, then the constructed exemplar could be $E_i = [3, 3]$. In the next generation, suppose that pbest and gbest are updated to $P_i = [5, 4]$ and $G = [2, 1]$, respectively, and the newly calculated exemplar is $E_i = [4, 3]$. In such a case, although both pbest and gbest evolve, the exemplar deteriorates. Hence, particle i learns from the deteriorated exemplar, which is inefficient. In the genetic learning scheme, however, selection is performed to overcome this problem. It ensures that the exemplar evolves generation by generation directionally until gets trapped in a local optimum. Moreover, once the exemplar gets trapped, the particle will select another particle's exemplar to learn from, hence flying out of the local optimum. It can be seen in Fig. 2 that GL-PSO is more efficient than GPSO. To summarize, by adopting the proposed genetic learning scheme consisting of crossover, mutation, and selection, the global search ability and search speed of the PSO algorithm are both improved.

IV. The *L-PSO Hybrid Paradigm

In addition to the GL-PSO algorithm, a generalized paradigm of hybridizing PSO with other optimization techniques for "learning" is proposed in this paper, which we termed the "*L-PSO" family. The PSO algorithm can be divided into two cascading layers, in which the upper layer is used for generating superior exemplars and the lower layer is applied to update particles as per ordinary PSO. Information in the upper layer propagates to the lower layer through (3) and (2), i.e., each particle learns from the superior exemplar to update velocity and position. On the other hand, the historical search information of particles in the lower layer is delivered as materials to the upper layer to breed even better exemplar. In this way, the two layers cooperate with each other and enhance each other.

Besides the GL-PSO, the *L-PSO family can embrace some other members such as DL-PSO that uses a differential learning scheme as in differential evolution (DE), EL-PSO with an estimation-of-distribution learning scheme adopting an estimation of distribution algorithm (EDA), etc. All these could lead to interesting future work. This paper uses the genetic learning method because of the technical and biological backgrounds described in Section II. Moreover, the proposed GL-PSO obtains promising numerical results which will be presented in Section V.

Besides, embedding some traditional mathematical methods could be alternative choices of learning. For example, the existing FIPS, CLPSO, and OLPSO algorithms apply a weighted-sum method, a simple composition approach, and an orthogonal experiment design, respectively. It is to be noticed that FIPS and CLPSO are not commonly identified as hybrid algorithms since their upper layer do not involve any evaluation procedure to calculate the fitness of the constructed exemplars.

V. Experimental Verification and Comparisons

A. Experimental Setup

To thoroughly evaluate the performance of the proposed GL-PSO, two popular test suites consisting of 42 benchmark functions in total [44], [45] are tested in the numerical experiments. A list and description of these functions are presented in Section S-III of the supplementary file. The first test suite (f_1–f_{14}) includes all the scalable functions recommended by Yao *et al.* [44] and has been widely used in [46]–[48]. In the test suite, f_1–f_4 are unimodal functions, f_5 is unimodal in 2-D and 3-D space but has multiple optima when $D > 3$, f_6 is a step function, and f_7 is a noisy quartic function. These functions are used to investigate the convergence feature of the algorithm, since many PSO variants improve the global search ability at the cost of slowing down their convergence rate. Then, f_8–f_{13} are multimodal functions with different landscapes, and f_{14} is a discontinuous version of the Rastrigin function f_9. These functions are used to show whether the proposed algorithm improves the exploration ability of previous PSO algorithms in order to avoid premature convergence.

The second test suite (f_{101}–f_{128}) consists of 28 shifted and rotated functions from the CEC 2013 test suit for real-parameter optimization, where f_1–f_5 are unimodal functions, f_6–f_{20} are multimodal functions, and f_{21}–f_{28} are hybrid composition functions [45], [49]. These shifted and rotated functions are used to test the performance of different algorithms in more complex and difficult cases.

The effectiveness and efficiency of the proposed GL-PSO are then compared with those of seven peer algorithms. GPSO [11] is the global PSO algorithm with inertia weight ω linearly decreasing from 0.9 to 0.4. Hierarchical PSO with Time-Varying Accelerating Coefficients (HPSO-TVAC) [13] makes improvements by adjusting accelerating coefficients and integrating auxiliary procedure. FIPS [26], CLPSO [27], and OLPSO [28], as described in Section III-A, focus on developing novel learning schemes and constructing effective exemplars. Besides, GAPSO [18] is a GA and PSO hybrid algorithm in which the GA and PSO run in parallel and exchange information by crossover. DEPSO [23] adopts a DE update strategy, a PSO update strategy, and a random update strategy alternately to improve particles.

TABLE I
Parameter Configurations

Algorithm	Parameter Configurations
GPSO	$\omega = 0.9 \sim 0.4,\ c_1 = c_2 = 2.0$
HPSO-TVAC	$c_1 = 0.5 \sim 0.5,\ c_2 = 0.5 \sim 2.5$
FIPS	$\omega = 0.7298,\ \sum c_i = 4.1$
CLPSO	$\omega = 0.7298,\ c = 1.49618,\ m = 7,\ Pc = 0.05 \sim 0.5$
OLPSO	$\omega = 0.9 \sim 0.4,\ c = 2.0,\ G = 5$
GAPSO	$\omega = 0.9 \sim 0.4,\ c_1 = c_2 = 2.0,\ pc = 0.5,\ pm = 0.05,\ ds = 7$
DEPSO	$\omega = 0.7298,\ c_1 = c_2 = 1.49618,\ C_R = 0.5,\ F_i \in [-1,-0.4] \cup [0.4,1],\ \delta = 0.2,\ I = 4$
GL-PSO	$\omega = 0.7298,\ c = 1.49618,\ pm = 0.01,\ sg = 7$

The parameter configurations of these algorithms are according to the corresponding references, which are shown in Table I. An exception is that, for GAPSO, we have conducted experiment to find out a promising parameter configuration because the parameter settings of *pc* (probability of crossover), *pm*, and *ds* (designated step) are not provided in the reference. For the proposed GL-PSO, parameters ω and c are empirically set to 0.7298 and 1.49618, respectively; the mutation probability is set as $pm = 0.01$; and the stopping gap is set as $sg = 7$. The population size is fixed at 50 for all the algorithms.

These algorithms are tested on 30-D functions with the same $MaxFEs = 10\,000 \cdot D$. For each benchmark, each algorithm is repeated 30 times independently to obtain statistical results. All the algorithms are coded in C, executed on a PC with the Intel Core2 Quad CPU Q6600 at 2.40 GHz with 2 GB of RAM (note that only a single processor is used).

In addition, because PSO belongs to nondeterministic algorithms, the differences of results are influenced by both intrinsic errors (determined by the algorithms' optimization performance) and random errors (caused by random number generation in the algorithm). To determine whether the differences of results are caused by intrinsic errors, a statistical hypothesis test is applied to evaluate whether the differences between the results are significant. In statistics, the Wilcoxon rank-sum test is one of the most well-known nonparametric statistical hypothesis tests [50]. In this paper, the one-tail Wilcoxon test at a significance level $\alpha = 0.05$ is applied to compare the results of algorithm pairs.

B. Experimental Results and Comparisons on First Test Suite (f_1–f_{14})

1) Results on Unimodal Functions: In Table II, the mean, best, and standard deviations of the error values obtained by the eight algorithms on functions f_1–f_7 are presented, where the best results are marked in bold. For these unimodal functions without local optimum, solution accuracy is the paramount criterion to compare the performance of different algorithms. However, as shown in Table II, although FIPS, CLPSO, GAPSO, and DEPSO reported improved results in the literature, their solution accuracy on unimodal functions is lower than the canonical PSO. On contrast, HPSO-TVAC,

TABLE II
STATISTICAL RESULTS ON UNIMODAL FUNCTIONS OF THE FIRST TEST SUITE

Function		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
f_1	Mean	1.30E-40	**4.87E-87**	2.45E-17	3.67E-16	2.02E-67	2.15E-11	1.23E-26	1.32E-81
	Best	2.76E-45	**1.04E-98**	8.91E-18	1.10E-16	8.10E-70	2.19E-13	5.38E-29	1.11E-83
	Std	4.41E-40	**2.41E-86**	1.40E-17	1.87E-16	5.84E-67	3.27E-11	2.98E-26	1.64E-81
	p-Val	*0.000000-	*0.000000+	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	
f_2	Mean	5.73E-28	1.10E-35	2.15E-10	3.50E-10	4.75E-37	1.12E-04	3.38E-15	**1.81E-46**
	Best	5.40E-30	1.48E-37	1.24E-10	1.90E-10	2.83E-38	2.83E-10	3.31E-16	**2.97E-47**
	Std	9.20E-28	1.98E-35	5.70E-11	8.96E-11	7.90E-37	6.13E-04	6.66E-15	**1.14E-46**
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	
f_3	Mean	0.426633	5.63E-12	16.789	551.297	0.209764	147.837	279.589	**9.75E-16**
	Best	6.96E-02	5.31E-14	4.76534	347.83	1.12E-02	10.9732	96.9292	**1.12E-17**
	Std	0.336919	9.71E-12	6.36673	134.548	0.269207	223.346	134.534	**1.45E-15**
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	
f_4	Mean	0.204476	1.56E-04	3.93E-03	7.14742	4.28E-02	0.936264	4.60E-02	**3.52E-06**
	Best	5.66E-02	9.41E-06	2.53E-03	5.6353	3.72E-03	0.513297	6.60E-03	**1.59E-08**
	Std	9.73E-02	1.79E-04	8.91E-04	0.853612	4.28E-02	0.383178	3.68E-02	**7.27E-06**
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.000000-	
f_5	Mean	46.0364	**1.16958**	24.2327	2.3734	14.3195	37.7172	29.4285	3.67107
	Best	6.38E-02	3.49E-05	22.0714	0.248147	**1.09E-05**	10.6493	0.144504	1.63E-03
	Std	31.7937	1.9459	**0.581468**	1.59176	30.8265	28.3629	24.4037	3.34417
	p-Val	*0.000000-	*0.000452+	*0.000000-	0.223210+	0.072660+	*0.000000-	*0.000000-	
f_6	Mean	**0**	1.06667	**0**	**0**	**0**	**0**	**0**	**0**
	Best	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Std	**0**	1.25762	**0**	**0**	**0**	**0**	**0**	**0**
	p-Val	#	*0.000000-	#	#	#	#	#	
f_7	Mean	5.90E-03	4.43E-02	**3.05E-03**	5.67E-03	9.35E-03	1.15E-02	1.08E-02	1.61E-02
	Best	2.37E-03	1.86E-02	**1.40E-03**	3.59E-03	5.11E-03	5.98E-03	5.56E-03	3.69E-03
	Std	2.08E-03	1.69E-02	**9.15E-04**	1.02E-03	2.78E-03	3.32E-03	3.10E-03	8.71E-03
	p-Val	*0.000000+	*0.000000-	*0.000000+	*0.000000+	*0.000095+	*0.026843+	*0.006106+	
Algorithm		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
Best-Mean		1	2	2	1	1	1	1	4
Sig-Better		5	5	5	4	4	5	5	
Sig-Worse		1	2	1	1	1	1	1	

* The difference between the two samples (*p*-value) is significant at level α=0.05 by the Wilcoxon rank-sum test.
\+ The compared algorithm is significantly better than GL-PSO.
− The compared algorithm is significantly worse than GL-PSO.
\# The two algorithms obtain same results.

TABLE III
SEARCH SPEED AND RELIABILITY ON UNIMODAL FUNCTIONS OF THE FIRST TEST SUITE

Function		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
f_1	FEs	166380	69765	79104	98920	98363	228871	37942	**21047**
	CPU	0.7599	0.3416	0.3568	0.3408	0.2338	0.8240	0.1438	**0.0806**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_2	FEs	161974	68333	88448	104019	101052	220491	42737	**22975**
	CPU	0.7719	0.3476	0.4266	0.3683	0.2652	0.8429	0.1695	**0.0938**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_3	FEs	217100	155998	178585	N/A	167023	281142	298178	**25402**
	CPU	1.4838	1.1068	1.2438	N/A	0.7721	1.6449	1.7927	**0.1518**
	SR%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**	43.33%	3.33%	**100.00%**
f_4	FEs	130531	69161	20171	249128	86772	57190	16040	**6196**
	CPU	0.6308	0.3510	0.0952	0.8442	0.2037	0.2192	0.0620	**0.0254**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_5	FEs	175072	90776	49178	112366	109029	220849	42150	**18799**
	CPU	0.8624	0.4681	0.2355	0.4145	0.2985	0.8642	0.1721	**0.0806**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	96.67%	**100.00%**	**100.00%**
f_6	FEs	162133	254137	46886	66836	86289	205498	35148	**24224**
	CPU	0.9755	1.5313	0.2744	0.3141	0.3064	1.0319	0.1788	**0.1302**
	SR%	**100.00%**	40.00%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_7	FEs	107209	53150	10775	41851	82991	34917	12798	**4601**
	CPU	0.5250	0.2578	0.0500	0.1558	0.2187	0.1342	0.0513	**0.0186**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
Avg-SR%		**100.00%**	91.43%	**100.00%**	85.71%	**100.00%**	91.43%	86.19%	**100.00%**

OLPSO, and GL-PSO successfully overcome this deficiency and achieve high solution accuracy. Moreover, the proposed GL-PSO algorithm performs the best in optimizing four out of the seven functions.

The *p*-value obtained in the hypothesis test is also reported in Table II. If the *p*-value is less than the significance level $\alpha = 0.05$, the difference of results is statistically significant. It can be seen that GL-PSO significantly outperforms seven other algorithms on most benchmarks except f_7. This is because f_7 is a noise function with random walk that distracts the selection of good exemplars and lowers the performance of GL-PSO.

The search speed of different algorithms is compared in Table III, where the average number of FEs and computing time (in s) used to reach the error bound are presented. Besides, Table III also reports the percentage of the runs successfully reaching the error bound in all the 30 runs, which reflects the reliability of the corresponding algorithms. The best value in each row is marked in bold. It can be observed from Table III that GL-PSO generally possesses the highest search speed as well as the highest reliability.

2) Results on Multimodal Functions: Comparisons of the algorithms on multimodal functions f_8–f_{14} are reported in Table IV. These functions contain a number of local optima, which may lead to premature convergence of PSO algorithms. Traditionally, PSO has seen difficulties in locating the global optimum of the Schwefel function f_8, because this problem has many deep local optima being far away from the global optimum. If a particle enters into a deep local optimum, it can hardly fly out of it. As shown in Table IV, five peer algorithms generate relatively poor results in optimizing f_8.

TABLE IV
STATISTICAL RESULTS ON MULTIMODAL FUNCTIONS OF THE FIRST TEST SUITE

Function		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
f_8	Mean	2377.99	1625.24	1915.13	**3.82E-04**	221.085	1.84816	4595.88	**3.82E-04**
	Best	1638.4	1065.95	829.813	**3.82E-04**	**3.82E-04**	0.482243	3355.77	**3.82E-04**
	Std	459.323	300.279	473.667	8.72E-13	134.622	0.916338	993.777	**0**
	p-Val	*0.000000-	*0.000000-	*0.000000-	#	*0.000000-	*0.000000-	*0.000000-	
f_9	Mean	23.4211	0.132661	28.6814	1.28248	0.49748	0.500083	114.354	**9.47E-15**
	Best	11.9395	1.78E-15	12.6597	2.71E-03	**0**	0.167859	20.8941	**0**
	Std	8.10328	0.344003	7.44754	1.78238	0.626549	0.186445	31.6682	**3.94E-14**
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	*0.013721-	*0.000000-	*0.000000-	
f_{10}	Mean	1.07E-14	2.56E-11	2.88E-09	9.23E-09	**7.34E-15**	5.38E-07	2.56E-14	7.46E-15
	Best	7.69E-15	8.07E-13	1.31E-09	5.86E-09	**4.14E-15**	6.60E-08	1.12E-14	**4.14E-15**
	Std	2.10E-15	6.79E-11	8.07E-10	1.85E-09	**1.08E-15**	4.15E-07	1.55E-14	1.85E-15
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	0.404530+	*0.000000-	*0.000000-	
f_{11}	Mean	2.26E-02	1.25E-02	**5.98E-14**	1.21E-08	1.40E-03	1.25E-02	1.45E-02	5.91E-03
	Best	**0**	4.44E-16	**0**	1.12E-11	**0**	2.12E-12	**0**	**0**
	Std	2.60E-02	1.19E-02	**2.72E-13**	2.12E-08	3.81E-03	1.92E-02	1.32E-02	7.33E-03
	p-Val	*0.000586-	*0.000436-	*0.019899+	0.5	*0.001246+	*0.007314-	*0.002028-	
f_{12}	Mean	6.91E-03	1.27E-30	1.52E-19	2.25E-17	**1.57E-32**	3.35E-13	1.19E-26	**1.57E-32**
	Best	**1.57E-32**	4.43E-31	3.47E-20	5.38E-18	**1.57E-32**	2.05E-15	1.35E-29	**1.57E-32**
	Std	2.63E-02	7.14E-31	9.79E-20	9.33E-18	**2.78E-48**	7.61E-13	6.16E-26	**2.78E-48**
	p-Val	*0.020948-	*0.000000-	*0.000000-	*0.000000-	#	*0.000000-	*0.000000-	
f_{13}	Mean	1.46E-03	1.38E-29	2.40E-18	3.05E-16	**1.35E-32**	1.46E-11	4.59E-27	**1.35E-32**
	Best	**1.35E-32**	4.54E-30	5.61E-19	9.12E-17	**1.35E-32**	3.27E-13	3.99E-29	**1.35E-32**
	Std	3.80E-03	6.78E-30	1.54E-18	1.45E-16	**5.57E-48**	2.70E-11	7.69E-27	**5.57E-48**
	p-Val	*0.000000-	*0.000000-	*0.000000-	*0.000000-	#	*0.000000-	*0.000000-	
f_{14}	Mean	9.96667	0.966667	34.0038	**0**	**0**	**0**	96.7243	**0**
	Best	3	**0**	25.4857	**0**	**0**	**0**	59.3867	**0**
	Std	4.86708	1.37674	5.22933	**0**	**0**	**0**	17.1089	**0**
	p-Val	*0.000000-	*0.000015-	*0.000000-	#	#	#	*0.000000-	
Algorithm		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
Best-Mean		0	0	1	2	4	1	0	5
Sig-Better		7	7	6	4	2	6	7	
Sig-Worse		0	0	1	0	1	0	0	

* The difference between the two samples (*p*-value) is significant at level α=0.05 by the Wilcoxon rank-sum test.
\+ The compared algorithm is significantly better than GL-PSO.
− The compared algorithm is significantly worse than GL-PSO.
\# The two algorithms obtain same results.

TABLE V
SEARCH SPEED AND RELIABILITY ON MULTIMODAL FUNCTIONS OF THE FIRST TEST SUITE

Function		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
f_8	FEs	148831	100636	212604	27692	18257	**5941**	N/A	13501
	CPU	2.1758	0.8312	1.7380	0.2020	0.1952	**0.0394**	N/A	0.0859
	SR%	26.67%	86.67%	56.67%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**
f_9	FEs	188189	45618	235135	139373	76936	27777	N/A	**17901**
	CPU	1.2799	0.3114	1.5520	0.7176	0.3170	0.1584	N/A	**0.1001**
	SR%	40.00%	**100.00%**	10.00%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**
f_{10}	FEs	174882	102445	93016	119923	107736	232235	47945	**23409**
	CPU	1.0996	0.6667	0.6401	0.6078	0.4003	1.2110	0.2665	**0.1239**
	SR%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_{11}	FEs	176605	75380	95143	112831	107881	239101	70717	**23000**
	CPU	1.2789	0.5655	0.6756	0.7311	0.4530	1.3569	0.4067	**0.1344**
	SR%	43.33%	50.00%	**100.00%**	**100.00%**	93.33%	63.33%	53.33%	76.67%
f_{12}	FEs	155678	81822	47219	81975	81870	176862	42349	**18815**
	CPU	1.6909	0.8344	0.4817	0.7828	0.6530	1.6686	0.3913	**0.1732**
	SR%	93.33%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_{13}	FEs	173298	99485	64304	98924	95594	226591	44959	**18911**
	CPU	1.9939	1.0239	0.6339	0.8839	0.7562	2.1282	0.4030	**0.1745**
	SR%	86.67%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
f_{14}	FEs	224121	31421	N/A	73199	74690	**7821**	N/A	13202
	CPU	1.9109	0.2629	N/A	0.4776	0.4109	**0.0558**	N/A	0.0956
	SR%	96.67%	**100.00%**	0.00%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**
Avg-SR%		70%	91%	67%	**100%**	99%	95%	50%	97%

On the contrary, the other three algorithms, GAPSO, CLPSO, and GL-PSO, are not trapped into poor local optima. The proposed GL-PSO is the only algorithm that can reach a low error of 3.82×10^{-4} in all the runs with standard deviation 0 on this function.

Consider the Rastrigin function f_9. It is a complex multimodal problem with a significant number of local optima. For this problem, an algorithm maintaining larger diversity is more likely to yield good results. It can be observed in Table IV that GL-PSO performs the best on this function, which means that the proposed genetic learning effectively maintains the population diversity. This success owes much to the mutation operation, which diversifies the exemplars and hence diversifies the search of particles most. In general, in terms of solution accuracy, the GL-PSO algorithm performs the best in five out of the seven multimodal functions. Moreover, according to the Wilcoxon test results (WTRs), GL-PSO significantly outperforms the other PSO variants on more benchmarks.

Furthermore, from the comparison of search speed and reliability shown in Table V, it can be observed that GL-PSO is also efficient in solving problems with a number of local optima. Generally, the proposed GL-PSO algorithm possesses both a strong global search ability and a high convergence speed, which are very promising for tackling multimodal problems.

It is to be noticed that, compared with the experimental results of GA and PSO reported in Section S-I of the supplementary file, the proposed GL-PSO algorithm improves performance over both GA and PSO on the unimodal Sphere function f_1 and the multimodal Schwefel function f_8. GL-PSO is not a passive combination of GA and PSO algorithms, but

TABLE VI
STATISTICAL RESULTS ON THE SECOND TEST SUITE (MEAN ERROR ± STANDARD DEVIATION)

	f_{101}	f_{102}	f_{103}	f_{104}	f_{105}	f_{106}	f_{107}
GPSO	2.27E-13 ± 5.97E-14	2.89E+06 ± 1.17E+06	3.67E+07 ± 4.23E+07	**1.24E+03 ± 3.21E+02**	2.27E-13 ± 5.97E-14	4.43E+01 ± 2.98E+01	2.36E+01 ± 8.41E+00
HPSO-TVAC	2.87E-12 ± 2.57E-12	2.75E+05 ± 1.23E+05	4.59E+08 ± 6.57E+08	6.51E+03 ± 2.52E+03	4.37E-11 ± 1.30E-10	**1.46E+01 ± 2.06E+01**	1.30E+02 ± 2.97E+01
FIPS	1.00E+00 ± 3.46E+00	1.09E+07 ± 2.89E+06	**1.88E+06 ± 2.08E+06**	9.34E+03 ± 2.28E+03	4.10E-02 ± 1.16E-01	3.43E+01 ± 2.34E+01	**2.15E+01 ± 9.71E+00**
CLPSO	2.27E-13 ± 0.00E+00	2.00E+07 ± 3.91E+06	4.15E+08 ± 2.26E+08	2.90E+04 ± 5.72E+03	5.64E-11 ± 1.95E-11	3.32E+01 ± 8.86E+00	7.95E+01 ± 1.02E+01
OLPSO	2.58E-13 ± 7.86E-14	1.60E+07 ± 1.96E+07	2.37E+07 ± 5.35E+07	5.83E+04 ± 1.52E+04	3.33E-13 ± 9.87E-14	2.70E+01 ± 2.15E+01	2.26E+01 ± 9.35E+00
GAPSO	6.46E-11 ± 1.20E-10	1.16E+07 ± 6.64E+06	2.72E+09 ± 2.51E+09	5.08E+03 ± 2.06E+03	3.17E-06 ± 2.87E-06	5.48E+01 ± 2.73E+01	7.10E+01 ± 3.23E+01
DEPSO	2.27E-13 ± 0.00E+00	1.54E+07 ± 7.83E+06	5.60E+08 ± 9.01E+08	4.14E+03 ± 1.04E+03	1.52E-13 ± 5.45E-14	4.12E+01 ± 2.75E+01	4.56E+01 ± 2.78E+01
GL-PSO	**1.97E-13 ± 7.86E-14**	**1.97E+05 ± 1.10E+05**	1.95E+07 ± 2.61E+07	1.56E+04 ± 5.21E+03	**1.48E-13 ± 5.30E-14**	1.61E+01 ± 1.19E+01	4.05E+01 ± 1.48E+01
RankOfGLPSO	1	1	2	6	1	2	4
	f_{108}	f_{109}	f_{110}	f_{111}	f_{112}	f_{113}	f_{114}
GPSO	**2.09E+01 ± 6.91E-02**	1.94E+01 ± 3.69E+00	1.61E-01 ± 1.02E-01	2.05E+01 ± 6.11E+00	7.37E+01 ± 3.20E+01	1.38E+02 ± 3.80E+01	1.11E+03 ± 2.77E+02
HPSO-TVAC	**2.09E+01 ± 6.04E-02**	3.15E+01 ± 2.49E+00	2.42E-01 ± 2.10E-01	2.26E+00 ± 2.26E+00	3.44E+02 ± 5.77E+01	4.01E+02 ± 6.95E+01	6.43E+02 ± 2.86E+02
FIPS	2.10E+01 ± 5.04E-02	2.30E+01 ± 2.68E+00	4.34E+00 ± 4.62E+00	6.34E+01 ± 1.35E+01	1.83E+02 ± 8.38E+00	1.59E+02 ± 1.40E+01	3.81E+03 ± 6.14E+02
CLPSO	**2.09E+01 ± 4.34E-02**	2.84E+01 ± 1.80E+00	3.42E+00 ± 1.03E+00	1.24E+00 ± 1.42E+00	1.61E+02 ± 2.02E+01	1.66E+02 ± 1.62E+01	5.35E+02 ± 1.23E+02
OLPSO	2.10E+01 ± 7.79E-02	2.26E+01 ± 7.70E+00	1.44E-01 ± 7.91E-02	6.30E-01 ± 8.05E-01	1.01E+02 ± 5.76E+01	1.35E+02 ± 4.27E+01	1.79E+02 ± 1.23E+02
GAPSO	2.10E+01 ± 5.38E-02	1.66E+01 ± 3.09E+00	1.72E+01 ± 7.75E+00	1.17E+00 ± 5.36E-01	1.06E+02 ± 3.17E+01	1.77E+02 ± 2.69E+01	7.81E+00 ± 2.48E+00
DEPSO	**2.09E+01 ± 5.26E-02**	**1.53E+01 ± 4.49E+00**	**4.43E-02 ± 2.54E-02**	1.70E+02 ± 1.90E+01	1.98E+02 ± 1.24E+01	2.02E+02 ± 1.79E+01	6.04E+03 ± 5.03E+02
GL-PSO	**2.09E+01 ± 7.28E-02**	1.87E+01 ± 4.14E+00	1.39E-01 ± 7.75E-02	**1.19E-12 ± 5.80E-12**	**5.07E+01 ± 1.41E+01**	**1.16E+02 ± 3.54E+01**	**9.25E-01 ± 7.49E-01**
RankOfGLPSO	2	3	2	1	1	1	1
	f_{115}	f_{116}	f_{117}	f_{118}	f_{119}	f_{120}	f_{121}
GPSO	4.81E+03 ± 1.44E+03	1.98E+00 ± 3.98E-01	6.09E+01 ± 9.30E+00	2.27E+02 ± 2.43E+01	2.96E+00 ± 6.31E-01	1.20E+01 ± 1.01E+00	3.35E+02 ± 8.91E+01
HPSO-TVAC	3.86E+03 ± 5.65E+02	1.34E+00 ± 4.15E-01	4.02E+01 ± 1.23E+01	3.65E+02 ± 7.95E+01	4.10E+00 ± 1.37E+00	1.37E+01 ± 9.83E-01	3.35E+02 ± 8.91E+01
FIPS	6.57E+03 ± 3.80E+02	2.51E+00 ± 2.32E-01	1.71E+02 ± 1.08E+01	2.08E+02 ± 6.87E+00	1.25E+01 ± 8.02E-01	1.21E+01 ± 6.28E-01	**2.77E+02 ± 3.90E+01**
CLPSO	6.09E+03 ± 3.11E+02	2.44E+00 ± 2.43E-01	7.23E+01 ± 4.92E+00	2.22E+02 ± 1.21E+01	3.23E+00 ± 8.03E-01	1.38E+01 ± 5.71E-01	3.00E+02 ± 3.17E+01
OLPSO	6.12E+03 ± 1.20E+03	1.90E+00 ± 3.78E-01	3.16E+01 ± 6.74E-01	1.81E+02 ± 3.19E+01	1.74E+00 ± 4.14E-01	1.34E+01 ± 1.55E+00	3.12E+02 ± 7.64E+01
GAPSO	3.99E+03 ± 8.32E+02	1.88E+00 ± 4.03E-01	3.61E+01 ± 1.11E+00	2.23E+02 ± 2.01E+01	2.20E+00 ± 4.56E-01	1.21E+01 ± 4.67E-01	3.14E+02 ± 7.63E+01
DEPSO	6.99E+03 ± 3.52E+02	2.57E+00 ± 3.12E-01	2.36E+02 ± 1.32E+01	2.42E+02 ± 1.00E+01	1.79E+01 ± 1.77E+00	1.24E+01 ± 5.05E-01	3.05E+02 ± 1.17E+02
GL-PSO	**3.55E+03 ± 6.99E+02**	**5.16E-01 ± 2.65E-01**	**3.14E+01 ± 5.67E-01**	**7.54E+01 ± 1.30E+01**	**1.60E+00 ± 2.76E-01**	**1.19E+01 ± 8.60E-01**	2.99E+02 ± 7.01E+01
RankOfGLPSO	1	1	1	1	1	1	2
	f_{122}	f_{123}	f_{124}	f_{125}	f_{126}	f_{127}	f_{128}
GPSO	1.09E+03 ± 3.03E+02	4.31E+03 ± 9.56E+02	2.46E+02 ± 1.51E+01	2.87E+02 ± 1.61E+01	3.22E+02 ± 5.56E+01	8.05E+02 ± 1.17E+02	4.09E+02 ± 3.58E+02
HPSO-TVAC	3.73E+02 ± 1.59E+02	5.07E+03 ± 8.59E+02	3.00E+02 ± 1.80E+01	3.48E+02 ± 1.75E+01	2.42E+02 ± 7.79E+01	1.22E+03 ± 1.16E+02	3.24E+03 ± 5.23E+02
FIPS	3.07E+03 ± 5.95E+02	7.22E+03 ± 2.69E+02	**2.11E+02 ± 6.17E+00**	**2.49E+02 ± 2.98E+01**	2.09E+02 ± 3.27E+01	**5.83E+02 ± 1.26E+02**	**2.71E+02 ± 6.32E+01**
CLPSO	9.95E+02 ± 2.11E+02	6.62E+03 ± 4.00E+02	2.73E+02 ± 6.00E+00	2.91E+02 ± 5.69E+00	**2.02E+02 ± 5.76E-01**	8.10E+02 ± 2.81E+02	3.00E+02 ± 1.11E-03
OLPSO	1.95E+02 ± 8.00E+01	6.19E+03 ± 1.12E+03	2.28E+02 ± 1.44E+01	2.65E+02 ± 8.18E+00	2.67E+02 ± 7.35E+01	6.57E+02 ± 9.66E+01	3.00E+02 ± 3.54E-13
GAPSO	1.12E+02 ± 4.88E+01	**4.20E+03 ± 9.85E+02**	2.65E+02 ± 9.03E+00	2.88E+02 ± 8.44E+00	2.75E+02 ± 7.52E+01	8.62E+02 ± 7.63E+01	3.85E+02 ± 2.67E+02
DEPSO	5.88E+03 ± 5.37E+02	7.22E+03 ± 4.40E+02	2.36E+02 ± 2.12E+01	2.85E+02 ± 3.53E+01	2.72E+02 ± 7.27E+01	7.90E+02 ± 1.69E+02	4.75E+02 ± 4.18E+02
GL-PSO	**1.04E+02 ± 3.18E+01**	4.33E+03 ± 1.01E+03	2.39E+02 ± 1.07E+01	2.67E+02 ± 8.12E+00	2.50E+02 ± 6.69E+01	6.65E+02 ± 1.18E+02	3.00E+02 ± 2.38E-13
RankOfGLPSO	1	3	4	3	4	3	2

organically embeds GA to benefit the swarm search of PSO, resulting in a "$1 + 1 > 2$" effect for both the GA and PSO.

C. Experimental Results and Comparisons on Second Test Suite (f_{101}–f_{128})

To further validate the performance of GL-PSO, comparisons are made on shifted and rotated functions f_{101}–f_{128}, which are very complex and difficult to solve. The mean and standard deviations of the error values obtained by the eight algorithms are listed in Table VI, where the performance rank of GL-PSO among the eight algorithms is also presented. It can be observed that the proposed algorithm is a competitive PSO variant for the second test suite. In optimizing the 28 functions, GL-PSO is ranked the first for 14 times, the second for six times, and ranked the third and fourth for four and three times respectively. Compared with the peer algorithms, the proposed algorithm can produce competitive results on most functions except f_{104}. Considering the fitness landscape of function f_{104}, it can be seen in [45] that the local regularities are smoothed. The proposed algorithm is not the best in dealing with this kind of problems.

TABLE VII
WTRS BETWEEN GL-PSO AND OTHER PSO ALGORITHMS ON THE SECOND TEST SUITE

Algorithm	GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO
Sig-Better	19	24	19	23	18	22	20
Sig-Worse	2	1	6	0	2	2	3

The WTRs on f_{101}–f_{128} are tabulated in Table VII, which shows that the proposed GL-PSO algorithm produces significantly better results than the other algorithms on more functions.

Moreover, the convergence curves of the eight algorithms are plotted in Section S-IV of the supplementary file. In most

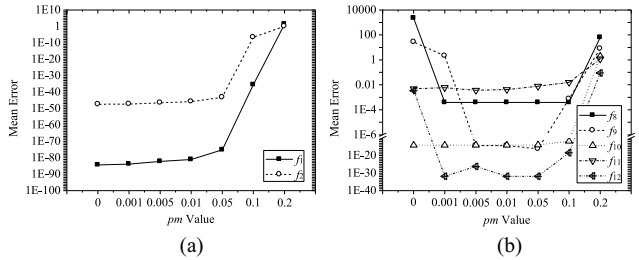


Fig. 4. Effect of the mutation probability *pm* on the performance of GL-PSO. (a) Unimodal functions. (b) Multimodal functions.

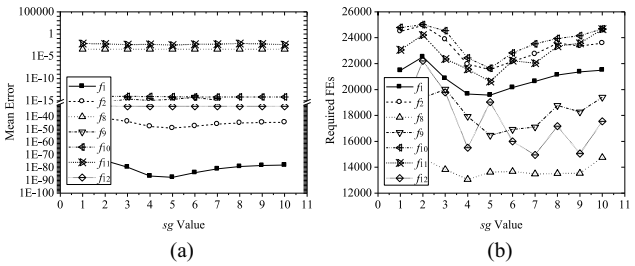


Fig. 5. Effect of the stopping gap *sg* on the performance of GL-PSO. (a) Solution accuracy. (b) Search efficiency.

cases, GL-PSO exhibits the fastest convergence among the compared algorithms, and meanwhile, it is not easy to be trapped in local optima. To conclude, the above results have verified the effectiveness and powerfulness of using the proposed genetic learning scheme for performance enhancement of PSO.

D. Investigation of Parameters

This section investigates the sensitivity of GL-PSO to the mutation probability *pm* and the stopping gap *sg*. First, we conduct GL-PSO with $pm = 0$, 0.001, 0.005, 0.01, 0.05, 0.1, and 0.2, respectively, and fix the other parameters as those presented in Section V-A. Unimodal functions f_1 and f_2 and multimodal functions f_8–f_{12} are tested in the experiments. The effect of different settings of *pm* on the performance of GL-PSO is plotted in Fig. 4, where the horizontal axis is the *pm* value and the vertical axis is the mean error for each function. It can be observed that the proper range of *pm* is [0.005, 0.05]. A larger *pm* will perturb a smooth search of particles and result in poor convergence, whereas a smaller *pm* cannot prevent premature convergence in solving multimodal problems. With a $pm \in [0.005, 0.05]$, the algorithm can exhibit favorable performance on both unimodal and multimodal problems.

In addition, GL-PSO is tested with $sg = 1, 2, \ldots, 10$, respectively, with the other parameters being fixed. Fig. 5(a) shows that the solution accuracy obtained by GL-PSO is insensitive to the stopping gap, and all the ten *sg* values can provide good performance for the algorithm. Besides, in Fig. 5(b), the search speed of using different *sg* values is depicted and compared, where the vertical axis is the average number of FEs required to reach the predefined error bound for each function. It can be observed that setting *sg* in the range of [4, 7] can help improving the search efficiency of GL-PSO. This parameter determines the condition of the jumping-like behavior of exemplars in the algorithm. A small *sg* value will lead to a particle sensitively changing its exemplar (search guidance), which would result in population oscillation and hence reduce the efficiency of PSO. On the other hand, when using a large *sg*, a particle can be trapped in a local optimum for a long time, which is also not efficient. To summarize, $pm \in [0.005, 0.05]$ and $sg \in [4, 7]$ are recommended in this paper.

In addition, in the crossover of the exemplar construction in GL-PSO, gbest of the entire swarm is utilized to generate new exemplars. If the local best information (lbest) of a neighborhood structure is used instead, GL-PSO can be implemented on different PSO topologies. Comparisons between global and local versions of GL-PSO are made in Section S-V of the supplementary file. Results show that GL-PSO with the global topology performs the best.

E. Scalability Analysis

In the literature, the performance of many PSO algorithms decreases drastically with the increase in the problem scale. In this section, we conduct scalability analysis for the above eight PSO algorithms to test their performance on 50-D and 100-D functions. Experimental settings are the same as those described in Section V-A, among which the *MaxFEs* is set to $10\,000 \cdot D$.

Table VIII reports the mean results obtained by the eight PSO variants, where the best results are marked in bold. It can be seen that the proposed algorithm maintains good performance when the problem dimension increases. Compared with the other algorithms, GL-PSO obtains the best results on most of the 50-D and 100-D problems being tested and it also achieves competitive results on the remaining a few problems.

Particularly, for the multimodal functions, the number of local optima increases drastically with the problem dimension, which makes PSO vulnerable to premature convergence. As shown in Table VIII, GL-PSO is the only algorithm that can maintain a good global search ability in such cases, whereas the performance of the other algorithms deteriorates severely. For example, in optimizing f_8, f_9, and f_{14}, GL-PSO obtains highly accurate results when the dimensionality increases to 50 and 100, whereas all the other seven algorithms get trapped in poor local optima.

F. Comparisons With Other Evolutionary and Swarm Intelligence Algorithms

In this section, we further compare GL-PSO with other evolutionary and swarm intelligence algorithms, including a continuous ant colony optimization (ACO_R) [51], self-adaptive DE (SaDE) [52], EDA [53], artificial bee colony (ABC) [54], and evolution strategy with covariance matrix adaptation (CMA-ES) [55]. 30-D problems are tested in the experiments, with parameter configurations according to the corresponding references. As the procedures and time complexity of these algorithms differ substantially, for a fair comparison, we use a maximum running time (*MaxRT*) instead of the *MaxFEs* as the stopping criterion. All algorithms are coded and executed in the same environment, with *MaxRT* being 1 s for all functions.

Table IX reports the rankings of the results obtained by the six algorithms as well as the pairwise WTRs (where "+" and

TABLE VIII
RESULTS OF SCALABILITY TEST ON 50-D AND 100-D PROBLEMS

Function		GPSO	HPSO-TVAC	FIPS	CLPSO	OLPSO	GAPSO	DEPSO	GL-PSO
f_1	50D	1.52E-30	3.96E-78	5.24E-13	5.86E-18	8.13E-27	1.42E-06	3.01E-15	**1.16E-87**
	100D	9.33E-21	2.62E-72	6.43E-06	6.28E-22	2.56E-19	0.091801	8.18502	**6.24E-82**
f_2	50D	4.49E-23	3.77E-34	1.60E-08	2.38E-11	5.67E-15	3.13E-06	4.23E-10	**1.65E-52**
	100D	3.94E-08	5.88E-33	5.43E-05	1.30E-13	1.48E-16	0.654	0.556	**2.05E-43**
f_3	50D	166.606	**8.13E-07**	1135.72	5972.15	3047.4	4234.9	4708.31	9.60E-07
	100D	5.19E+04	**0.020**	1.93E+04	5.83E+04	4.36E+04	5.09E+04	4.00E+04	17.600
f_4	50D	6.284	**0.046**	0.238	8.037	1.819	6.539	6.904	0.248
	100D	18.910	**0.645**	6.936	7.966	19.260	17.593	20.780	2.013
f_5	50D	80.656	**6.046**	44.574	32.878	81.250	132.304	65.568	34.200
	100D	184.341	**23.686**	95.964	174.392	161.281	1.48E+04	19903.3	98.0994
f_6	50D	**0**	7.1	**0**	**0**	0.033	0.033	**0**	**0**
	100D	0.433	90.033	**0**	**0**	0.033	2.933	60.733	**0**
f_7	50D	0.020	0.114	**7.48E-03**	7.69E-03	0.055	0.038	0.048	0.037
	100D	0.090	0.370	0.041	**0.012**	0.132	0.680	0.498	0.097
f_8	50D	6695.4	2609.59	5698.79	7.897	538.238	89.823	8424.64	**6.36E-04**
	100D	32533.2	5231.69	23818	71.0643	1081.74	3909.89	23070.6	**1.27E-03**
f_9	50D	57.144	1.360	118.525	35.392	7.562	10.827	289.103	**1.13E-15**
	100D	172.447	7.032	455.179	220.515	12.864	165.255	743.692	**1.45E-14**
f_{10}	50D	2.00E-14	1.80E-11	2.04E-07	6.26E-10	2.77E-14	1.20E-04	3.41E-08	**1.53E-14**
	100D	6.91E-10	4.50E-11	3.30E-04	3.50E-12	4.61E-12	0.029	2.493	**7.05E-14**
f_{11}	50D	0.011	0.011	**2.47E-12**	2.78E-11	2.38E-03	4.27E-03	6.24E-03	3.77E-03
	100D	2.96E-03	6.87E-03	5.65E-06	**1.11E-17**	0.011	0.060	0.895	0.016
f_{12}	50D	0.021	1.14E-29	3.23E-15	2.21E-19	0.035	2.07E-03	5.37E-08	**9.63E-33**
	100D	0.091	7.72E-28	7.70E-08	1.28E-23	0.061	0.313	23.749	**1.09E-32**
f_{13}	50D	5.12E-03	3.08E-28	6.94E-14	3.61E-18	5.86E-03	7.91E-04	3.68E-04	**6.13E-30**
	100D	0.011	**4.28E-26**	4.08E-06	3.49E-22	0.063	24.13	9405.47	1.10E-03
f_{14}	50D	40.136	2.733	112.127	24.414	8.942	0.090	237.05	**0**
	100D	192.215	9.568	418.832	184.44	12.763	49.194	660.981	**0**

TABLE IX
COMPARISONS BETWEEN GL-PSO AND OTHER ALGORITHMS FOR FUNCTION OPTIMIZATION (RUN TIME = 1 S)

Function	ACO_R Rank (WTR)	SaDE Rank (WTR)	EDA Rank (WTR)	ABC Rank (WTR)	CMA-ES Rank (WTR)	GL-PSO Rank
f_1	6 (−)	3 (−)	4 (−)	1 (+)	5 (−)	2
f_2	6 (−)	3 (−)	4 (−)	1 (+)	5 (−)	2
f_3	5 (−)	3 (−)	4 (−)	6 (−)	2 (−)	1
f_4	6 (−)	3 (−)	4 (−)	2 (−)	5 (−)	1
f_5	6 (−)	3 (−)	5 (−)	2 (−)	4 (−)	1
f_6	6 (−)	1 (≈/=)	1 (≈/=)	1 (≈/=)	1 (≈/=)	1
f_7	6 (−)	2 (+)	1 (+)	4 (−)	5 (−)	3
f_8	5 (−)	3 (−)	6 (−)	1 (≈/=)	4 (−)	1
f_9	5 (−)	1 (+)	6 (−)	1 (+)	4 (−)	3
f_{10}	6 (−)	5 (−)	1 (+)	3 (−)	4 (−)	2
f_{11}	6 (−)	4 (≈/=)	1 (+)	1 (+)	3 (+)	5
f_{12}	6 (−)	1 (≈/=)	4 (−)	1 (≈/=)	5 (−)	1
f_{13}	6 (−)	3 (≈/=)	2 (−)	1 (≈/=)	4 (−)	4
f_{14}	5 (−)	1 (≈/=)	6 (−)	1 (≈/=)	4 (−)	1
f_{101}	6 (−)	1 (+)	1 (+)	5 (−)	4 (−)	3
f_{102}	4 (−)	3 (≈/=)	6 (−)	5 (−)	1 (+)	2
f_{103}	6 (−)	2 (+)	4 (−)	5 (−)	1 (+)	3
f_{104}	5 (−)	1 (+)	3 (−)	6 (−)	4 (−)	2
f_{105}	6 (−)	1 (+)	5 (−)	3 (−)	4 (−)	2
f_{106}	5 (−)	4 (−)	6 (−)	2 (≈/=)	1 (+)	3
f_{107}	6 (−)	2 (+)	3 (≈/=)	5 (−)	1 (+)	4
f_{108}	6 (−)	3 (−)	4 (−)	2 (≈/=)	5 (−)	1
f_{109}	2 (+)	5 (−)	6 (−)	4 (−)	1 (+)	3
f_{110}	4 (−)	3 (−)	6 (−)	5 (−)	1 (+)	2
f_{111}	6 (−)	2 (+)	5 (−)	1 (+)	4 (−)	3
f_{112}	5 (−)	3 (−)	4 (−)	6 (−)	1 (≈/=)	2
f_{113}	6 (−)	2 (≈/=)	4 (−)	5 (−)	1 (≈/=)	3
f_{114}	5 (−)	3 (−)	6 (−)	1 (+)	4 (−)	2
Rank_Sum	152	71	112	81	88	63
Rank_Final	6	2	5	3	4	1
Sig_Better	27	13	22	15	18	
Sig_Worse	1	8	4	6	7	

"−" denote that the compared algorithm is significantly better and worse than GL-PSO, respectively, and "≈/=" stands for that the differences between the results are not significant). Consider the rankings in Table IX. GL-PSO performs the best among the six algorithms, followed by SaDE, ABC, and CMA-ES. The proposed algorithm exhibits consistent performance and ranks the first or second in optimizing most of the benchmark functions. Moreover, the WTRs show that GL-PSO produces significantly better results than the other algorithms on more functions.

VI. CONCLUSION

In this paper, a genetic learning scheme for PSO algorithm has been proposed, which adopts genetic operators, specifically, crossover, mutation, and selection, to construct exemplars. The crossover utilizes the particles' historical information pbests and gbest to generate high-quality offspring, whereas the mutation injects diverse information into the offspring to enhance global exploration. Moreover, the selection operation ensures that each exemplar evolves directionally generation by generation. This way, the bred exemplars are well diversified and highly qualified, which are capable of providing improved guidance for the evolving particles. By adopting such a genetic learning scheme, a GL-PSO has been developed. In the experiments to verify the performance of GL-PSO, a group of numerical benchmarks with different characteristics are used, and the proposed algorithm is compared with several representative PSO algorithms. Experimental results show that the proposed GL-PSO outperforms the other algorithms on a majority of benchmarks in terms of the global search ability, solution accuracy, search speed, reliability, and scalability. Furthermore, this paper also proposes a generalized hybrid paradigm of PSO with generic learning techniques, the *L-PSO. Differing from the mechanistic parallel combination, *L-PSO presents a cascade hybrid paradigm that auxiliary learning techniques such as any evolutionary algorithm are used to generate improved exemplars to guide the search of particles. As a specific example belonging to this hybrid paradigm, the success of GL-PSO can encourage

future research into the cascade hybridization of PSO with other and emerging techniques. Moreover, this would also lead to interesting future development of PSO algorithms for dynamic and multiobjective optimization, as well as real-world applications [56], [57].

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Perth, WA, Australia, 1995, pp. 1942–1948.

[2] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Human Sci.*, Nagoya, Japan, 1995, pp. 39–43.

[3] Y. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. G. Harley, "Particle swarm optimization: Basic concepts, variants and applications in power systems," *IEEE Trans. Evol. Comput.*, vol. 12, no. 2, pp. 171–195, Apr. 2008.

[4] M. R. AlRashidi and M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power systems," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 913–918, Aug. 2009.

[5] R. Ruiz-Cruz, E. N. Sanchez, F. Ornelas-Tellez, A. G. Loukianov, and R. G. Harley, "Particle swarm optimization for discrete-time inverse optimal control of a doubly fed induction generator," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1698–1709, Dec. 2013.

[6] R. V. Kulkarni and G. K. Venayagamoorthy, "Particle swarm optimization in wireless-sensor networks: A brief survey," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 41, no. 2, pp. 262–267, Mar. 2011.

[7] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1656–1671, Dec. 2013.

[8] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1272–1282, Dec. 2005.

[9] 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.

[10] J. J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proc. IEEE Swarm Intell. Symp.*, Pasadena, CA, USA, 2005, pp. 124–129.

[11] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evol. Comput.*, Anchorage, AK, USA, 1998, pp. 69–73.

[12] 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.

[13] A. Ratnaweera, S. K. Halgamuge, and H. C. 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.

[14] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1362–1381, Dec. 2009.

[15] X.-H. Shi, Y.-C. Liang, H.-P. Lee, C. Liu, and L. M. Wang, "An improved GA and a novel PSO-GA-based hybrid algorithm," *Inf. Process. Lett.*, vol. 93, no. 5, pp. 255–261, Mar. 2005.

[16] Y.-T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Appl. Soft Comput.*, vol. 8, no. 2, pp. 849–857, Mar. 2008.

[17] F. Valdez, P. Melin, O. Castillo, and O. Montiel, "A new evolutionary method with a hybrid approach combining particle swarm optimization and genetic algorithms using fuzzy logic for decision making," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 1333–1339.

[18] K. Premalatha and A. M. Natarajan, "Hybrid PSO and GA for global maximization," *Int. J. Open Prob. Compt. Math.*, vol. 2, no. 4, pp. 597–608, Dec. 2009.

[19] 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.

[20] F. Grimaccia, M. Mussetta, and R. E. Zich, "Genetical swarm optimization: Self-adaptive hybrid evolutionary algorithm for electromagnetics," *IEEE Trans. Antennas Propag.*, vol. 55, no. 3, pp. 781–785, Mar. 2007.

[21] W.-T. Li, X.-W. Shi, Y.-Q. Hei, S.-F. Liu, and J. Zhu, "A hybrid optimization algorithm and its application for conformal array pattern synthesis," *IEEE Trans. Antennas Propag.*, vol. 58, no. 10, pp. 3401–3406, Oct. 2010.

[22] S. Jeong, S. Hasegawa, K. Shimoyama, and S. Obayashi, "Development and investigation of efficient GA/PSO-HYBRID algorithm applicable to real-world design optimization," *IEEE Comput. Intell. Mag.*, vol. 4, no. 3, pp. 36–44, Aug. 2009.

[23] C.-S. Zhang, J.-X. Ning, S. Lu, D.-T. Ouyang, and T.-N. Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization," *Oper. Res. Lett.*, vol. 37, no. 2, pp. 117–122, Mar. 2009.

[24] S. Li, M. Tan, I. W. Tsang, and J. T.-Y. Kwok, "A hybrid PSO-BFGS strategy for global optimization of multimodal functions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 4, pp. 1003–1014, Aug. 2011.

[25] Z.-H. Liu, J. Zhang, S.-W. Zhou, X.-H. Li, and K. Liu, "Coevolutionary particle swarm optimization using AIS and its application in multiparameter estimation of PMSM," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1921–1935, Dec. 2013.

[26] 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.

[27] 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.

[28] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 832–847, Dec. 2011.

[29] C.-H. Li, S.-X. Yang, and T. T. Nguyen, "A self-learning particle swarm optimizer for global optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 3, pp. 627–646, Jun. 2012.

[30] Z.-H. Ren, A.-M. Zhang, C.-Y. Wen, and Z.-R. Feng, "A scatter learning particle swarm optimization algorithm for multimodal problems," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1127–1140, Jul. 2014.

[31] G. E. Robinson and R. E. Page, "Genetic determination of nectar foraging, pollen foraging, and nest-site scouting in honey bee colonies," *Behav. Ecol. Sociobiol.*, vol. 24, no. 5, pp. 317–323, May 1989.

[32] P. Berthold and F. Pulido, "Heritability of migratory activity in a natural bird population," *Proc. Biol. Sci.*, vol. 257, no. 1350, pp. 311–315, Sep. 1994.

[33] S. M. Stigler, "Darwin, Galton and the statistical enlightenment," *J. Roy. Stat. Soc. A.*, vol. 173, no. 3, pp. 469–482, Jul. 2010.

[34] T. J. Bazzett, *An Introduction to Behavior Genetics.* Sunderland, MA, USA: Sinauer Assoc., 2008.

[35] N. E. Raine, T. C. Ings, A. Dornhaus, N. Saleh, and L. Chittka, "Adaptation, genetic drift, pleiotropy, and history in the evolution of bee foraging behavior," in *Advances in the Study of Behavior*. New York, NY, USA: Academic Press, 2006, pp. 305–354.

[36] J. Alcock, *Animal Behavior: An Evolutionary Approach*, 5th ed. Sunderland, MA, USA: Sinauer Assoc., 1993.

[37] K. Sterelny, "Made by each other: Organisms and their environment," *Biol. Philos.*, vol. 20, no. 1, pp. 21–36, Jan. 2005.

[38] C. G. Jones, J. H. Lawton, and M. Shachak, "Positive and negative effects of organisms as physical ecosystem engineers," *Ecology*, vol. 78, no. 7, pp. 1946–1957, 1997.

[39] S. H. Ling *et al.*, "Hybrid particle swarm optimization with wavelet mutation and its industrial applications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 3, pp. 743–763, Jun. 2008.

[40] M. S. Arumugam and M. V. C. Rao, "On the improved performances of the particle swarm optimization algorithms with adaptive parameters, crossover operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 324–336, 2008.

[41] F. Galton, *Hereditary Genius.* London, U.K.: Macmillan, 1869.

[42] J. L. Fuller and W. R. Thompson, *Behavior Genetics*. New York, NY, USA: Wiley, 1960.

[43] (Sep. 10, 2015). *GLPSO Source Code*. [Online]. Available: http://www.ai.sysu.edu.cn/GYJ/glpso/c_code/

[44] 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.

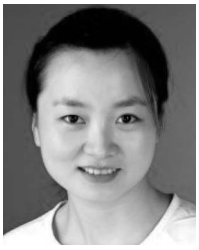
[45] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou, China, Tech. Rep. 201212, 2013.

[46] H. Gao and W.-B. Xu, "A new particle swarm algorithm and its globally convergent modifications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 5, pp. 1334–1350, Oct. 2011.

[47] T. Blackwell, "A study of collapse in bare bones particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 3, pp. 354–372, Jun. 2012.
[48] W.-J. Yu *et al.*, "Differential evolution with two-level parameter adaptation," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1080–1099, Jul. 2014.
[49] M. Ergezer and D. Simon, "Mathematical and experimental analyses of oppositional algorithms," *IEEE Trans. Cybern.*, vol. 44, no. 11, pp. 2178–2189, Nov. 2014.
[50] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.
[51] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *Eur. J. Oper. Res.*, vol. 185, no. 3, pp. 1155–1173, Mar. 2008.
[52] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.
[53] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña, "Optimization in continuous domains by learning and simulation of Gaussian networks," in *Proc. Genet. Evol. Comput. Conf.*, Las Vegas, NV, USA, 2000, pp. 201–204.
[54] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Glob. Optim.*, vol. 39, no. 3, pp. 459–471, Nov. 2007.
[55] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, 2003.
[56] X.-L. Hu and J. Wang, "An improved dual neural network for solving a class of quadratic programming problems and its k-winners-take-all application," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2022–2031, Dec. 2008.
[57] L. Shao, R.-M. Yan, X.-L. Li, and Y. Liu, "From heuristic optimization to dictionary learning: A review and comprehensive comparison of image denoising algorithms," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1001–1013, Jul. 2014.

Yue-Jiao Gong (S'10–M'15) received the Ph.D. degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2014.

She is currently a Post-Doctoral Research Fellow with the Department of Computer and Information Science, University of Macau, Macau, China. Her current research interests include evolutionary computation and swarm intelligence. She has published over 30 papers, including nine IEEE Transaction papers, in her research area.



Jing-Jing Li received the Ph.D. degree in computer science from Hong Kong Polytechnic University, Hong Kong, in 2012.

She is currently a Lecturer with the School of Computer Science, South China Normal University, Guangzhou, China. Her current research interests include evolutionary algorithm, energy efficient routing, and object tracking for wireless sensor networks.

Yicong Zhou (M'07–SM'14) received the Ph.D. degree in electronic engineering from Tufts University, Medford, MA, USA, in 2010.

He is currently an Assistant Professor with the Department of Computer and Information Science, University of Macau, Macau, China. He has authored/co-authored over 80 papers, including 13 IEEE Transaction papers, six most downloaded/popular papers in corresponded journals, and one "highly cited paper" within the top 1% of published papers in the ISI database up to 2015. His current research interests include chaotic system design, multimedia security, image processing and understanding, and machine learning.

Dr. Zhou was a recipient of the third price of Macau Natural Science Award in 2014. He is a member of the International Society for Optical Engineers and the Association for Computing Machinery.

Yun Li (S'87–M'90) received the Ph.D. degree in computing and control from the University of Strathclyde, Glasgow, U.K., in 1990.

He was a Visiting Professor with Kumamoto University, Kumamoto, Japan, in 2002. He has served as the Founding Director of the University of Glasgow Singapore, Singapore, from 2011 to 2013, and acted as the Founding Director of the University's international joint program with the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2013. He is currently a Professor with the Department of Systems Engineering, University of Glasgow, Glasgow, U.K., and a Visiting Professor with the UESTC and Sun Yat-sen University, Guangzhou, China, researching into smart design with market informatics via the cloud to complete the value chain for Industry 4.0. He has 200 publications and he is a Chartered Engineer.

Dr. Li is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the *SM Journal of Engineering Sciences*.

Henry Shu-Hung Chung (M'95–SM'03) received the Ph.D. degree in electrical engineering from Hong Kong Polytechnic University, Hong Kong, in 1994.

He is currently a Professor with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. He has authored six research book chapters and over 320 technical papers including 150 refereed journal papers in his research areas and holds 26 patents. His current research interests include time- and frequency-domain analysis of power electronic circuits, switched-capacitor-based converters, random-switching techniques, and control methods.

Dr. Chung is currently the Editor-in-Chief of the IEEE POWER ELECTRONICS LETTERS, and an Associate Editor of the IEEE TRANSACTIONS ON POWER ELECTRONICS, the IEEE Journal of Emerging and Selected Topics in Power Electronics, and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, PART I.

Yu-Hui Shi (SM'98) received the Ph.D. degree in electronic engineering from Southeast University, Nanjing, China, in 1992.

He is currently a Professor with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong–Liverpool University, Suzhou, China. His current research interests include computational intelligence techniques (including swarm intelligence) and their applications.

Dr. Shi is the Editor-in-Chief of the *International Journal of Swarm Intelligence Research* and an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. He is the Chair of the IEEE Chief Information Officer Task Force on Swarm Intelligence.

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

He is currently a Changjiang Chair Professor with the Department of Computer Science, Sun Yat-sen University, Guangzhou, China. His current research interests include computational intelligence, cloud computing, data mining, and power electronic circuits. He has published over 200 technical papers in his research area.

Dr. Zhang was a recipient of the China National Funds for Distinguished Young Scientists Award from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, and the IEEE TRANSACTIONS ON CYBERNETICS.