Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 1957812, 17 pages
https://doi.org/10.1155/2020/1957812

Research Article

Sequential Hybrid Particle Swarm Optimization and Gravitational Search Algorithm with Dependent Random Coefficients

Shanhe Jiang, Chaolong Zhang, and Shijun Chen

Department of Physics and Power Engineering, Anqing Normal University, Anqing 246011, China

Correspondence should be addressed to Shanhe Jiang; jshxlxlw@163.com

Received 30 December 2019; Revised 1 March 2020; Accepted 30 March 2020; Published 21 April 2020

Guest Editor: Dr. Dilbag Singh

Particle swarm optimization (PSO) has been proven to show good performance for solving various optimization problems. However, it tends to suffer from premature stagnation and loses exploration ability in the later evolution period when solving complex problems. This paper presents a sequential hybrid particle swarm optimization and gravitational search algorithm with dependent random coefficients called HPSO-GSA, which first incorporates the gravitational search algorithm (GSA) with the PSO by means of a sequential operating mode and then adopts three learning strategies in the hybridization process to overcome the aforementioned problem. Specifically, the particles in the HPSO-GSA enter into the PSO stage and update their velocities by adopting the dependent random coefficients strategy to enhance the exploration ability. Then, the GSA is incorporated into the PSO by using fixed iteration interval cycle or adaptive evolution stagnation cycle strategies when the swarm drops into local optimum and fails to improve their fitness. To evaluate the effectiveness and feasibility of the proposed HPSO-GSA, the simulations were conducted on benchmark test functions. The results reveal that the HPSO-GSA exhibits superior performance in terms of accuracy, reliability, and efficiency compared to PSO, GSA, and other recently developed hybrid variants.

1. Introduction

As many real-world optimization problems become increasingly complex, traditional optimization algorithms cannot sufficiently satisfy the problem requirements and better effective optimization algorithms are needed. Hence, various kinds of metaheuristic algorithms that are inspired by natural phenomena have launched into a center stage in recent decades for solving complex optimization problems. Genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], artificial immune system (AIS) [3], differential evolution (DE) [4], ant colony optimization (ACO) [5], glowworm swarm optimization (GSO) [6], artificial bee colony (ABC) [7], gravitational search algorithm (GSA) [8], grey wolf optimization (GWO) [9], cat swarm optimization (CSO) [10], harmony search algorithm (HS) [11], and bacterial foraging optimization algorithm (BFOA) [12] have been developed in recent years by researchers and have shown superior performance for solving a wide range of optimization problems, such as function optimization [4–9, 11–15], fuzzy inference system [16, 17], image processing [18, 19], economic dispatch [20, 21], and neural networks training [22, 23].

Overall, the review of the presented literature states that there is no single superior method for solving optimization problems. That is, no one algorithm can solve all of the optimization problems, but each algorithm can solve a special class of problems. Although the aforementioned algorithms that have been proposed to solve optimization problems do achieve good performance, there are still undesirable shortcomings. For instance, PSO often suffers from premature convergence, whereas it tends to be trapped into local optima due to the rapid convergence speed [24]. GSA requires a long computation time to find the solution for some problems [21]. Hence, there is a lot of room for improvement in finding the better optimization algorithm.

Another issue is how to balance the exploration/exploitation search ability for a single metaheuristic algorithm including PSO or GSA. The key operation of metaheuristic optimization algorithms is how to keep a better trade-off

between exploitation and exploration abilities in the searching process. A good algorithm should have the capability of these two abilities to seek the global optimal solution. However, some algorithms present more outstanding advantage on one of these abilities. For instance, PSO has a tendency to rapid convergence in a multivariable optimization problem. By comparison, GSAs global exploration performance is particularly conspicuous. Hence, PSO and GSA approaches possess respective advantages and potentialities. It encourages us to develop an appropriate hybridization technique of different metaheuristic algorithms to mitigate the weakness of the original algorithm and obtain the outstanding optimization performance against a single algorithm, thereby acquiring rapid response and avoiding premature convergence.

Inspired by abovementioned ideas, to further improve the respective drawbacks of PSO and GSA, a novel combination strategy that integrates PSO with GSA, sequential hybrid particle swarm optimization and gravitational search algorithm with dependent random coefficients called HPSO-GSA, is proposed in this paper based on a sequential hybrid pattern. To be specific, we first propose a new velocity updating equation for PSO based on dependent random coefficients strategy to enhance the balance between exploitation and exploration search. Second, the existing PSO evolution framework is improved, and the GSA is incorporated into the PSO by using a sequential mode when the swarm drops into local optimum and fails to improve their fitness. That is, the HPSO-GSA first enters into the PSO phase to update its velocity and position. Then, the GSA operator is carried out on condition that the fixed iteration interval cycle or adaptive evolution stagnation cycle strategies are met in the process of evolution. Finally, the performance of the proposed algorithm is evaluated against PSO, GSA, and state-of-the-art hybrid variants by using a set of benchmark test functions. The results reveal that the proposed HPSO-GSA can achieve better optimization performance compared with the involved algorithms.

The remainder of this paper is organized as follows. A brief review of related works on PSO and GSA is given in Section 2. Section 3 introduces the proposed HPSO-GSA approach. In Section 4, the experiments, comparisons, and discussion for the used benchmark test problems are carried out to evaluate the performance of the proposed algorithm. Finally, the conclusion and future work are given in Section 5.

2. Related Works

In this section, we first introduce the relevant backgrounds including the PSO and GSA algorithms. Next, the state-of-the-art PSO and GSA hybrid variants are reviewed.

2.1. Particle Swarm Optimization (PSO). PSO is a population-based metaheuristic optimization method, which was originally introduced by Kennedy and Eberhart [2]. Since its development, PSO has become one of the most promising optimizing techniques for solving global optimization problems. The algorithm is motivated by intelligent collective behavior like the movement of a flock of birds, a school of fish, and a group of ants to seek for foods. Compared to other optimization techniques, PSO is easy to implement with few parameters to adjust and is computationally inexpensive. PSO does not require any gradient information from the objective functions, and it uses only primitive mathematical operators through the exchange of information among candidate individuals (particles), in order to attain desirable optimization performance. In the past decades, the PSO has been shown to successfully solve a wide range of optimization fields. It achieves better results more speedily and more cheaply than other methods, such as GA, DE, and ACO [25].

Suppose that each particle which represents a potential solution of the problem in the population size N flies through a D-dimensional search space. It is associated with two vectors, namely, a position vector $x_i(t) = [x_i^1(t), x_i^2(t), ..., x_i^D(t)]$ and a velocity vector $v_i(t) = [v_i^1(t), v_i^2(t), ..., v_i^D(t)]$ for the current iteration t. The personal best experience of the ith particle is represented by $p_i(t) = [p_i^1(t), p_i^2(t), ..., p_i^D(t)]$, and the best global position of the swarm found so far is stored in $p_g(t) = [p_g^1(t), p_g^2(t), ..., p_g^D(t)]$. The initial position for each particle $x_i(0) = [x_i^1(0), x_i^2(0), ..., x_i^D(0)]$ in the population is randomly generated from in the range of the decision space of the problem. The initial velocity $v_i(0) = [v_i^1(0), v_i^2(0), ..., v_i^D(0)]$ for each dimension is set as a zero vector. Then, the particle's velocity and its new position are updated by the following equations:

$$v_i^d(t+1) = w(t)v_i^d(t) + c_1(t)r_1(t)\left(p_i^d(t) - x_i^d(t)\right) + c_2(t)r_2(t)\left(p_g^d(t) - x_i^d(t)\right), \quad (1)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \quad (2)$$

where $d \in \{1, 2, ..., D\}, i \in \{1, 2, ..., N\}$. $r_1(t)$ and $r_2(t)$ are two mutually independent random coefficients drawn from uniform distributed within the range $[0, 1]$. $w(t)$ is the linear time-varying inertia weight (TVIW) factor within the interval $[0.4, 0.9]$ suggested by Shi and Eberhart [26]. Its goal is to control the impact of the previous velocity on the current velocity, thereby influencing the trade-off between global exploration and local search of the particles. $c_1(t)$ and $c_2(t)$ are two linear time-varying acceleration coefficients (TVAC) suggested by Ratnaweera et al. [27]. Their objectives are to control the influence of personal and swarm best experiences, respectively.

2.2. Gravitational Search Algorithm (GSA). GSA inspired by the Newton law of gravity and mass interactions is a newly developed swarm-based metaheuristic optimization method [8]. In GSA, all agents are regarded as objects including different masses, and their performances are evaluated by their masses using a fitness function. Each agent attracts each other agent through the gravity force which is directly proportional to the product of their masses and inversely

proportional to the square of the distance between them. This force leads to global movement of all agents towards heavier masses with the aid of the Newtonian law of motion. The heavier agent that corresponds to better solution to a problem moves more slowly than the lighter one. Then, it is concluded that masses should be attracted by the heaviest mass which represents an optimum solution in the search space by lapse of time.

Let us consider a population including N agents (masses) in a D-dimensional decision space, and the position vector of the ith agent is described as $x_i(t) = [x_i^1(t), x_i^2(t), ..., x_i^D(t)]$ at iteration t, where $x_i^d(t)$ represents the position of the ith agent in the dth dimension at iteration t. The algorithm initializes the N agents randomly in the given decision space. During the evolution process, the gravitational force acting on agent i from agent j at iteration t is defined as (3) and the whole force that acts on agent i in the dth dimension with randomly weighted sum of the dth component of the forces coming from other agents are given as (4):

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right), \tag{3}$$

$$F_i^d(t) = \sum_{j \in \text{kbest}, j \neq i} \text{rand}_j \times F_{ij}^d(t), \tag{4}$$

where $M_{aj}(t)$ represents the active gravitational mass related to agent j and $M_{pi}(t)$ represents the passive gravitational mass related to agent i. $G(t)$ is the gravitational constant defined by (5). ε is a small constant value. $R_{ij}(t)$ is the Euclidian distance between agents i and j defined by (6). rand_j is uniform random number in the interval [0, 1]. kbest represents the set of first K agents with the best fitness value and the biggest mass, which is defined as a function of time with the original value K_0 at the beginning, and it is linearly decreased to 1 by lapse of iteration. Finally, there will be just one agent applying force to the others:

$$G(t) = G_0 \exp\left(-\alpha \frac{t}{t_{\max}}\right), \tag{5}$$

$$R_{ij}(t) = \left\| x_i(t), x_j(t) \right\|. \tag{6}$$

According to the Newton law of motion, the acceleration of an agent is proportional to the resultant force and inverse of its mass, so the acceleration of the ith agent in dth dimension at iteration t is denoted as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}. \tag{7}$$

The next velocity of agent i is defined as a fraction of its current velocity added to its acceleration. Sequentially, its next position is calculated based on the corresponding velocity:

$$\begin{aligned} v_i^d(t+1) &= \text{rand}_i v_i^d(t) + a_i^d(t), \\ x_i^d(t+1) &= x_i^d(t) + v_i^d(t+1). \end{aligned} \tag{8}$$

The mass of agent i is calculated by (9), and the normalization of the calculated mass is given as (10):

$$m_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\text{best}(t) - \text{worst}(t)}, \tag{9}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^N m_j(t)}, \tag{10}$$

$$M_{ai} = M_{pi} = M_{ii} = M_i, \quad i = 1, 2, ..., N, \tag{11}$$

where $\text{fit}_i(t)$ represents the fitness value of agent i at iteration t and best (t) and worst (t) represent the best and worst fitness value in the current population at iteration t, respectively. The gravitational mass $M_i(t)$ represents the mass of agent i at iteration t which embodies the fitness evaluation value of agent i.

For a minimization optimization problem, best (t) and worst (t) are defined as follows:

$$\begin{aligned} \text{best}(t) &= \min_{i \in \{1,...,N\}} \text{fit}_i(t), \\ \text{worst}(t) &= \max_{i \in \{1,...,N\}} \text{fit}_i(t). \end{aligned} \tag{12}$$

2.3. State-of-the-Art PSO and GSA Hybrid Variants. Extensive amounts of works have been performed to improve the original PSO and GSA performance. Among these works, studies on hybrid systems that combine skilled metaheuristic optimization algorithms to obtain good compromise between exploration and exploitation have gained extensive popularity. The most classical PSO variants have been reported in [14, 15, 23, 28–36]. Kao and Zahara [28] proposed the hybridization strategy of PSO and GA (GAPSO) for solving multimodal test functions. In GAPSO, individuals in a new generation are drawn not only from crossover and mutation operation in GA but also from movement mechanism in PSO. The results show the superiority of the hybrid GAPSO approach in terms of solution quality and convergence speed. Esmin et al. [29] introduced a PSO algorithm coupled with GA mutation operator, namely, HPSOM, for solving unconstrained global optimization problems. Shunmugalatha and Slochanal [30] proposed a hybrid particle swarm optimization (HPSO), which incorporated the crossover, mutation operators, and subpopulation process in the genetic algorithm into particle swarm optimization. The implementation of HPSO on test functions shows that it converges to better solution much faster. Zhang and Xie [31] introduced a hybrid particle swarm optimization with a differential evolution operator (DEPSO), which provides the bell-shaped mutation with consensus on the population diversity along with the evolution. A set of benchmark functions was applied to evaluate its efficiency. A hybrid algorithm named DE-PSO was proposed by Zhang et al. [32], which incorporates concepts from DE and PSO, updating particles not only by DE operators but also by mechanisms of PSO. The proposed algorithm was tested on several benchmark functions. Liu et al. [14] proposed a novel hybrid algorithm named PSODE,

where DE is incorporated to update the previous best positions of PSO particles to force them to jump out of local attractor in order to prevent stagnation of population. Besides GA and DE, PSO has been hybridized with extremal optimization (EO) [15], central force optimization (CFO) [23], estimation of distribution algorithm (EDA) [33], artificial immune system (AIS) [34], gravitational search algorithm [35], and teaching-learning-based optimization (TLBO) [36]. Overall, these PSO-based hybrid variants have been successfully utilized for solving global optimization problems.

Similarly, a novel hybrid version of GSA variants has also been reported in [18, 22, 37–39]. For instance, Mirjalili et al. [13, 22] proposed a novel hybrid PSOGSA algorithm by adopting a parallel model for solving benchmark function optimization and feedforward neural networks. A hybrid approach that integrated differential evolution into gravitational search algorithm (DE-GSA) for unconstrained optimization was introduced by Li et al. [37]. Chen et al. [38] proposed a hybrid gravitational search algorithm combined with simulated annealing (GSA-SA) for the traveling salesman problem. A new hybrid approach, namely, genetic algorithm-based gravitational search algorithm (GA-GSA), was proposed to solve image segmentation [18]. A novel GSA-SVM hybrid system which hybridizes the GSA with support vector machine (SVM) was proposed to improve classification accuracy with an appropriate feature subset in binary problems [39]. Apparently, these hybrid systems of GSA have demonstrated powerful results when compared with other approaches such as the original GSA itself, DE, GA, and PSO.

2.4. Comparison of Particle Swarm Optimization and Gravitational Search Algorithm. To thoroughly understand the two metaheuristic optimization methods, we have identified three similarities and four differences between the PSO and the GSA. The similarities are as follows: (1) both are population-based metaheuristic algorithms; (2) particles/agent positions are updated by iteration; and (3) both algorithms use velocity formulations for position updating. On the other hand, they differ in the following aspects: (1) PSO simulates the social behavior of birds, whereas GSA was inspired by a physical phenomenon; (2) PSO employs fitness values for the two best positions pbest and gbest, while GSA uses fitness values to calculate masses that are proportional to gravitational forces; (3) PSO particles update their positions by means of dynamic velocities with cognitive and social behaviors, while GSA agents calculate their positions using changing accelerations with the concept of Newtonian gravity; and (4) PSO uses memory to store and update the velocity with the pbest and gbest, while GSA is memoryless and is concerned exclusively with the current status. Therefore, PSO and GSA have their respective specialties and potentialities to find optimum solutions. It encourages us to further design a hybridization of these two techniques to obtain better optimization performance.

3. Sequential Hybrid Particle Swarm Optimization and Gravitational Search Algorithm

As is well known, PSO ensures that the optimization process converges faster, whereas GSA assures that the search can jump out of local optima by maintaining the diversity in the swarm [40]. Moreover, they have different search properties and movement mechanisms. Hence, we propose the new sequential hybrid version HPSO-GSA, where PSO is integrated with GSA to combine the merits of both algorithms. To be specific, first, to further balance between global and local search of the PSO, dependent random coefficients (DRCs) strategy (Section 3.1) is introduced into the HPSO-GSA. Second, to decrease the computational cost due to GSAs integration in the hybridization, two GSA-embedded strategies, namely, fixed iteration interval cycle (FIIC) (Section 3.2) and adaptive evolution stagnation cycle (AESC) (Section 3.3), are introduced into the algorithm. Finally, computational complexity of the algorithm in Section 3.4 is theoretically analyzed based on main operators.

3.1. Dependent Random Coefficients (DRCs). As shown in equation (1), the two random coefficients $r_1(t)$ and $r_2(t)$ are generated independently, so in some cases, the values of the $r_1(t)$ and $r_2(t)$ are too large or too small. For the former case, both the personal and social influences are excessively evaluated and the particles are driven too far away from the suboptimum solution. In the latter case, both the personal and social influences are negligible and the convergence speed of the algorithm is sharply reduced. To alleviate these phenomena, the dependent random coefficients strategies based on these random variables are introduced as follows:

$$v_i^d(t+1) = w(t)v_i^d(t) + c_1(t)r_1(t)\left(1-r_2(t)\right)\left(p_i^d(t) - x_i^d(t)\right) + c_2(t)r_2(t)\left(1-r_1(t)\right)\left(p_g^d(t) - x_i^d(t)\right). \tag{13}$$

To demonstrate the impact of the DRC strategy on the evolution of the population, an experiment was performed on the Rosenbrock and Ackley test functions. The average velocity of particles varying throughout iterations is shown in Figure 1. On the one hand, it is desirable that the particles with high velocities can explore large areas in the decision space to find new regions. From Figure 1, we can see that the DRC strategy relatively increases the average velocity of the population in the early iterations, thereby improving the diversity of the swarm that provides the particles with the ability to jump out of premature convergence. On the other hand, in the later stage, the particles in the population need to exploit local regions more precisely to improve their performances. It is obvious from the results that the average velocity of the algorithm with the DRC strategy has a lower value than that of the algorithm with independent random coefficients. In this case, the particles can find a better solution with a faster convergence speed in the last iterations.

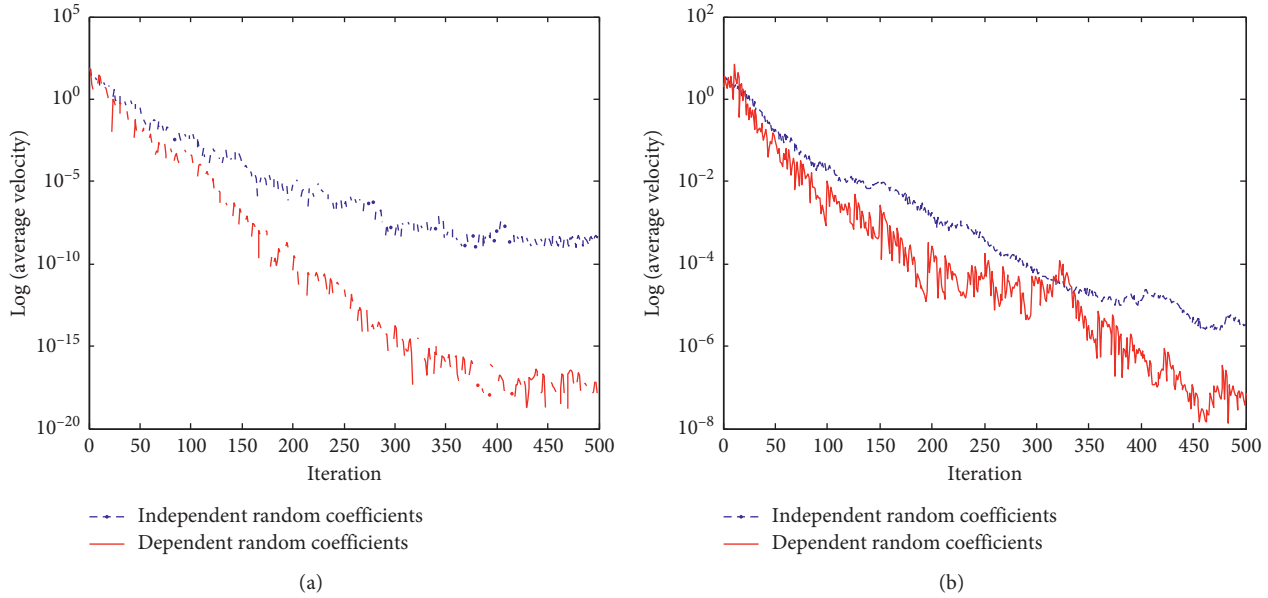

Figure 1: Comparison results of average velocity of particles for (a) Rosenbrock and (b) Ackley function.

Instead, the velocity of the algorithm with independent random variables decreases slowly. Then, the particles need more iterations or runtimes to seek an optimum solution.

3.2. Fixed Iteration Interval Cycle (FIIC). A FIIC strategy is that the GSA is implanted into the PSO evolution framework according to a fixed iteration interval frequency T_f. For instance, T_f is set to 10; that is, the GSA can be introduced into the PSO every ten iterations. However, the optimal setting of the parameter T_f is relatively challenging and dependent on the function of termination condition such as maximum iterations. In most cases, the parameter value T_f is usually determined through the experience or the parameter sensitivity analysis. Hence, in this paper, the parameter sensitivity analysis by means of the experiments was carried out and the proper parameter value is determined. Too large or too small values of T_f are not desirable, as the former may be difficult to utilize the contribution of the GSA, whereas the latter leads to waste computation resources, thereby degrading the algorithm convergence speed. According to the results of the multigroup experiments, the T_f can be set to a large value in the range [30, 50] to save the convergence time on condition that the test function is unimodal or multimodal with relatively less local optima. Similarly, the T_f can be set to a small value within the range [1, 32] to enhance the global convergence ability on condition that the optimization function is multimodal with many local optima. In this work, the T_f is fixed at 20 according to the parameter sensitivity analysis shown in Section 4.3.

3.3. Adaptive Evolution Stagnation Cycle (AESC). To adaptively implant the GSA into the PSO framework to guide the particles access to potential regions during the entire evolution process, we employ an AESC strategy in the HPSO-GSA to effectively judge whether the algorithm reaches the premature stagnation stage. That is, the particles are found to be trapped into local optima. The AESC strategy is defined as follows:

Definition 1 (evolution stagnation). Suppose the fitness evaluation function for a given optimization problem is defined as $f(\cdot)$, and the global best position of the population found so far at iteration t is described as $p_g(t)$. For a small positive constant $\delta \geq 0$, if and only if $|f(p_g(t)) - f(p_g(t+1))| \leq \delta$ is satisfied, in such a way, the population presents evolution stagnation situation at iteration t+1 in the evolving process, where δ is the evolution stagnation radius.

Definition 2 (evolution stagnation cycle). For a small position constant $\delta \geq 0$, if the population is unvaryingly kept in evolution stagnation at minimum successive iterations $S_{\min}$ for radius δ, then the value $S_{\min}$ is defined as evolution stagnation cycle for radius δ.

The evolution stagnation cycle is calculated by (14) at the successive evolving process:

$$S(t+1) = \begin{cases} S(t)+1, & \left|f\left(p_g(t)\right) - f\left(p_g(t+1)\right)\right| \leq \delta, \\ 0, & \left|f\left(p_g(t)\right) - f\left(p_g(t+1)\right)\right| > \delta. \end{cases} \tag{14}$$

In the HPSO-GSA, the AESC strategy is applied when the condition $S(t+1) \geq S_{\min}$ is satisfied; that is, evolution stagnation cycle $S(t+1)$ reaches or exceeds its threshold value $S_{\min}$. In this case, the GSA operator is added to the PSO to increase its flexibility for solving more complicated problems. Hence, the parameter $S_{\min}$ has a direct impact on the performance of the HPSO-GSA. Too small or too large value of $S_{\min}$ is undesirable for the HPSO-GSA. In our work,

we set $S_{\min} = 4$ in terms of the result of the parameter sensitivity analysis shown in Section 4.3.

3.4. Computational Complexity Analysis. Computational complexity is usually regarding the analysis of storage space requirements and computational time costs. In most cases, the time complexity analysis is the main issue for population-based metaheuristic algorithms [15, 23]. Generally, the time complexity of an algorithm is proportional to the number of dimensions D and the number of particles or agents N in the swarm when only considering a main loop and therefore can be calculated according to their main operations and worst case in a full iteration cycle. A greater number of dimensions D or a larger population size N will directly result in higher running time. As a result, the computational step analysis for PSO, GSA, and HPSO-GSA in an iterative loop is given in Table 1. From Table 1, the time consumed considering their main operations and worst case in an iterative loop for the PSO and the GSA is $O(2N + 2ND)$ and $O(2N + 3ND)$, respectively. Therefore, the worst-case consuming time for the HPSO-GSA in a single iterative loop is $O(3N + 3ND)$. Apparently, the consuming time of the algorithm is proportional to the number of particle N and the dimension of decision space D in the population. A larger population size N and/or a greater dimension D will directly result in more running time. Then, the time complexity for the HPSO-GSA is greater than that of the PSO or the GSA when the GSA is entered into the PSO. However, the hybrid approach can avoid premature convergence and thus is capable of escaping from local optima with the help of an increase in diversity. Therefore, the HPSO-GSA is still a very competitive optimization approach at the expense of a little higher time resource.

In order to describe clearly the steps of the proposed algorithm, the detailed pseudocode of the HPSO-GSA algorithm is summarized in Figure 2. It is obvious that the HPSO-GSA procedure is mainly dependent on the PSO algorithm with DRCS, and the GSA can be allowed to perform when the AESC and/or the FIIC strategies are satisfied.

4. Experimental Setup, Results, and Discussion

In this section, the experimental studies that have been performed to investigate the performance of the proposed HPSO-GSA method for classical benchmark test functions are presented. We first describe the benchmark test functions in Section 4.1. Second, the experimental design including parameter settings of the involved algorithms for comparison is described in Section 4.2. The parameter sensitivity analysis of T_f and $S_{\min}$ and the effect of different strategies in the HPSO-GSA are discussed in Sections 4.3 and 4.4, respectively. Finally, the performance investigation and comparison of the proposed algorithm is evaluated between the PSO, the GSA, and other variants of hybrid algorithm.

4.1. Benchmark Test Functions. The classical benchmark test functions with different complexities of the fitness landscape are shown in Table 2. These functions were considered in the study [15, 41] as well. This table consists of the brief descriptions of function expressions, their feasible domains, their optimal positions, and global minimum. All the functions are minimization problems. The variable D denotes the dimensions of the test functions. These functions, grouped into three sets, were designed to evaluate various aspects of algorithms. The first set of $f_1(x)$ to $f_3(x)$ consists of unimodal test functions. The second set of $f_4(x)$ to $f_9(x)$ is multimodal high-dimensional test functions with many local optima. Moreover, $f_8(x)$ and $f_9(x)$ are hybrid composition test functions. The third set, given by $f_{10}(x)$, consists of multimodal test functions with fixed dimensions.

4.2. Parameter Settings of the Involved Algorithms. In this section, we employ PSO [26], GSA [8], and five hybrid variants such as DEPSO [31], GAPSO [28], DE-GSA [37], GA-GSA [18], and PSOGSA [22] for comparison with the HPSO-GSA. These algorithms have a few parameters, some of which are common and others are specific to the algorithms.

Common parameters are the number of dimensions for the search space, the maximum number of iterations, the population size, and the total number of trials. For all test functions, except the $2D$ function Schaffer, we test the experiments with 30 dimensions, that is, $D = 30$. To assure the fair assessment between HPSO-GSA and its peers, all PSO variants are run independently 100 times on the test functions employed. The population size is also set to 60. The maximum number of function evaluations $\mathrm{FE}_{\max}$ is 5000 for the Schaffer function and 30,000 for the remainders, respectively. The evolution stagnation radius δ is chosen as $1.0e-2$. The convergence criterion ε for test functions in 30 dimensions is fixed at $1.0e-3$. Each run stopped when the maximum number of iterations or the convergence criterion is reached. Similarly, the population is initialized with its position and velocity, both of which are randomly selected from the range $[x_{\max}]$. We set $v_{\max} = 0.5x_{\max}$. The upper and lower bounds of the particle's position are limited to the interval $[x_{\min}, x_{\max}]$, and the maximum velocity is restricted to $v_{\max}$. Note that all the experiments are conducted in a Windows XP Professional OS environment using Intel Core i5, 2.67 GHz, 2G RAM, and the codes are performed in Matlab 7.0.1.

Besides the common parameters, the parameter configurations for all variants employed are extracted from their optimized suggestions in the corresponding publications and are described in Table 3. For our HPSO-GSA, it is important to note that the evolution stagnation cycle $S_{\min}$ is selected as 4 and the fixed iteration interval cycle T_f is set to 20 according to the analysis of parameter sensitivity observation.

4.3. Parameter Sensitivity Analysis. The key parameters T_f and $S_{\min}$ have a direct effect on the performance of the HPSO-GSA. Hence, the experiments are performed to

Table 1: Computational complexity analysis for PSO, GSA, and HPSO-GSA.

Main step for PSO	Complexity	Main step for GSA	Complexity	Main step for HPSO-GSA	Complexity
Evaluate objectives	N	Evaluate objectives	N	Evaluate objectives	N
Update best positions	N	Compute masses	N	Update best positions	N
Update velocities	$N \times D$	Compute accelerations	$N \times D$	Compute masses	N
Compute new positions	$N \times D$	Update velocities	$N \times D$	Compute accelerations	$N \times D$
		Compute new positions	$N \times D$	Update velocities	$N \times D$
				Compute new positions	$N \times D$

Pseudo-code of the HPSO-GSA algorithm:
Initialization
1: Initialize the parameters: $x_{\max}$, $x_{\min}$,$v_{\max}$, N, D, $w_{\min}$, $w_{\max}$, $c_{1\min}$, $c_{1\max}$, $c_{2\min}$, $c_{2\max}$, α ,G_0, T_f, $S_{\min}$, $iter_{\max}$, ε, and δ
2: **for** each particle **do**
3: Initialize randomly the position $X_i^0 \in \Omega$
4: Initialize randomly the velocity $V_i^0 \le V_{\max}$
5: **end for**
6: Evaluate the fitness value of each agent (particle) f_i^0
7: Find the personal best positions $p_i^0 \leftarrow f_i^0$ and the global best position $p_g^0 \leftarrow f_{best}^0$
8: Set $t = 0$ // t for iterations
9: Set $S(t) = 0$ // $S(t)$ for evolution stagnation cycle
Loop
10: **while** (termination condition is not met) **do**
11: $t = t + 1$
12: **for** (i = 1 to N) **do** // i for agents
13: **for** (j = 1 to D) **do** // j for dimensions
14: Update the velocity (v_{id}) and position (x_{id}) of each particle using equations (15) and (2), respectively
15: **end for**
16: Evaluate the fitness value of each agent f_i^t
17: Update $p_i^t \leftarrow f_i^t$ and $p_g^t \leftarrow f_{best}^t$
18: **end for**
19: Compute the evolution stagnation cycle $S(t)$ by Eq. (16)
20: **if** ((t mod T_f) = 0) or ($S(t) \geqslant S_{\min}$)) **then** // T_f for fixed interval cycle and $S_{\min}$ for adaptive interval cycle
//GSA is added to PSO when the conditions are met
21: Perform the GSA processes using equations (8) and (9), respectively
22: Evaluate the fitness value of each agent f_i^{t+1}
23: Update $p_i^{t+1} \leftarrow f_i^{t+1}$ and $p_g^{t+1} \leftarrow f_{best}^{t+1}$
24: **end if**
25: when the termination condition is met, output results. Otherwise, go to step 10
26: **end while**
Termination

Figure 2: Pseudocode of the HPSO-GSA algorithm.

investigate the effect of different parameters on the proposed HPSO-GSA. For simplicity, the maximum number of iterations is set as 1000, and other parameters are the same as previously mentioned. Four well-known test functions, namely, the Sphere, Rosenbrock, Rastrigrin, and Griewank problems, have been employed to observe how the algorithm is affected by these parameters.

First, the parameter T_f of the proposed HPSO-GSA is tuned. Each test function has been tested on HPSO-GSA with different values of T_f; however, it is impossible to evaluate all the cases of the parameter. Hence, the eight selected values for this parameter $T_f = 2$, $T_f = 5$, $T_f = 10$, $T_f = 15$, $T_f = 20$, $T_f = 30$, $T_f = 40$, and $T_f = 50$ are considered, respectively. The results are averaged over 100 independent runs, and the normalized average best fitness values achieved for each parameter T_f are shown in Figure 3. Note that the normalized average best fitness is defined by means of the variable ($H_i(\tau) \in [0, 1]$) calculated as follows:

$$\mathrm{H}_i(\tau) = \frac{\mathrm{fit}_i(\tau) - \mathrm{fit}_{\min}(\tau)}{\mathrm{fit}_{\max}(\tau) - \mathrm{fit}_{\min}(\tau)}, \tag{15}$$

where i represents the group index of different parameters T_f (apparently, $i = 1, 2, \ldots, 8$) and τ denotes the Sphere, Rosenbrock, Rastrigrin, and Griewank functions. $\mathrm{fit}_i(\tau)$ is the average best fitness under the ith parameter T_f for function τ. $\mathrm{fit}_{\min}(\tau)$ and $\mathrm{fit}_{\max}(\tau)$ denote the minimum and maximum fitness under all of the cases for function τ, respectively. The aim of normalizing is to reduce the influence of different magnitudes in the same coordinate. From Figure 3, it is apparent that the searching capabilities of the HPSO-GSA are influenced by different parameter T_f. It is an interesting conclusion that too small or two large values of T_f tend to compromise the convergence accuracy of the HPSO-GSA. The best results are obtained when T_f is fixed at 20 for most of test functions. Hence, in our simulations, the HPSO-GSA uses the parameter $T_f = 20$.

Table 2: Benchmark test functions used in the experiments.

Function name	Function expression	Domain	Opt. position	Opt. value	Trait
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	$(0, 0, \ldots, 0)$	0	Unimodal
Rosenbrock	$f_2(x) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	$[-30, 30]$	$(1, 1, \ldots, 1)$	0	Unimodal
Quartic noise	$f_3(x) = \sum_{i=1}^{D} ix_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]$	$(0, 0, \ldots, 0)$	0	Unimodal
Schwefel	$f_4(x) = 418.98 \times D - \sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]$	(420.96,. . .,420.96)	0	Multimodal
Rastrigin	$f_5(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	$(0, 0, \ldots, 0)$	0	Multimodal
Ackley	$f_6(x) = -20\exp(-0.2\sqrt{(1/D)\sum_{i=1}^{D} x_i^2}) - \exp((1/D)\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	$[-30, 30]$	$(0, 0, \ldots, 0)$	0	Multimodal
Griewank	$f_7(x) = (1/4000)\sum_{i=1}^{D} (x_i)^2 - \prod_{i=1}^{D}\cos(x_i/\sqrt{i}) + 1$	$[-600, 600]$	$(0, 0, \ldots, 0)$	0	Multimodal
Penalized1	$f_8(x) = (\pi/D)\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\} + \sum_{i=1}^{D} u_i(x_i, 10, 100, 4)$	$[-50, 50]$	$(1, 1, \ldots, 1)$	0	Multimodal
Penalized2	$f_9(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2 \cdot [1 + \sin^2(3\pi x_i + 1)] + (x_D - 1)^2 \cdot [1 + \sin^2(2\pi x_D)]\} + \sum_{i=1}^{D} u_i(x_i, 5, 100, 4)$	$[-50, 50]$	$(1, 1, \ldots, 1)$	0	Multimodal
Schaffer	$f_{10}(x) = 0.5 + (sin\sqrt{(x_1^2 + x_2^2)})^2 - 0.5/(1 + 0.001(x_1^2 + x_2^2))^2$	$[-100, 100]$	$(0, 0, \ldots, 0)$	0	Multimodal

Remark: in the Penalized1 and Penalized2, $y_i = 1 + (1/4)(x_i + 1))$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & \text{if } x_i > a, \\ 0 & \text{if } -a \le x_i \le a, \\ k(-x_i - a)^m & \text{if } x_i < -a. \end{cases}$

Table 3: Parameter settings of the involved algorithms.

Algorithms	Parameter settings	Reference
PSO	$w_{\max} = 0.9$, $w_{\min} = 0.4$, $c_1 = c_2 = 2.0$	[26]
GSA	$G_0 = 100$, $\alpha = 20$	[8]
DEPSO	$c_1 = c_2 = 2.0$, $w = 0.4$, CR = 0.9, $F = 0.5$	[31]
GA-PSO	$c_1 = c_2 = 2.0$, $w = 0.4$, $P_c = 0.55$, $P_m = 0.01$	[28]
DE-GSA	$G_0 = 100$, $\alpha = 20$, CR = 0.9, $F = 0.5$	[37]
GA-GSA	$G_0 = 100$, $\alpha = 20$, $P_c = 0.65$, $P_m = 0.01$	[18]
PSOGSA	$G_0 = 1.0$, $\alpha = 20$, $c_1 = c_2 = 1.0$, $w_{\max} = 0.9$, $w_{\min} = 0.4$	[22]
HPSO-GSA	$G_0 = 100$, $\alpha = 20$, $w_{\max} = 0.9$, $w_{\min} = 0.4$, $c_{1\,\min} = 0.5$, $c_{1\,\max} = 2.5$, $c_{2\,\min} = 0.5$, $c_{2\,\max} = 2.5$, $T_f = 20$, $S_{\min} = 4$	Present work

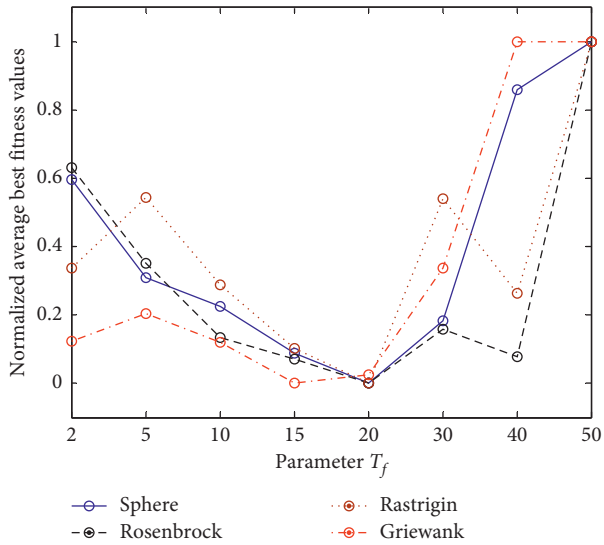

Figure 3: Results of the HPSO-GSA for different parameter T_f.

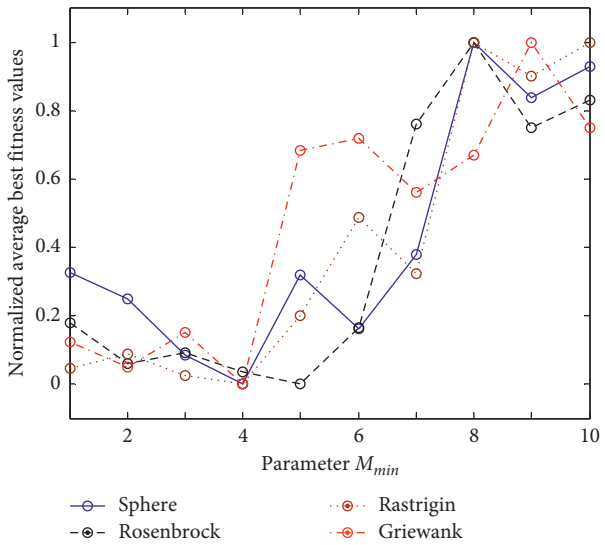

Figure 4: Results of the HPSO-GSA for different parameter $M_{\min}$.

Second, the parameter $S_{\min}$ is similarly considered. The possible ten scenarios are tested when the range of this parameter is selected from [1, 10] by step size 1. The results obtained by the HPSO-GSA for test functions are given in Figure 4. It is clear that, for the Rosenbrock function, $S_{\min} = 5$ produces the best average best fitness. For other functions, the best results are obtained for $S_{\min} = 4$. Hence, $S_{\min} = 4$ is a desirable choice for the proposed algorithm in the following experiments.

4.4. Effectiveness of Different Strategies. The proposed algorithm employs three strategies, namely, DRC, FIIC, and AESC. To evaluate the impact on the performance improvement incurred by each of these strategies, we investigate the performance of (1) HPSO-GSA without the DRC strategy and with the FIIC strategy (HPSO-GSA1), (2) HPSO-GSA without the DRC strategy and with the FIIC strategy (HPSO-GSA2), (3) HPSO-GSA with the DRC strategy and the FIIC strategy (HPSO-GSA3), (4) HPSO-GSA with the DRC strategy and the AESC strategy (HPSO-GSA4), and (5) the complete HPSO-GSA. For HPSO-GSA, the three strategies are integrated with the algorithm. In this experiment, seven test functions, except for function Schaffer, in 10 dimensions are considered. The maximum number of iterations is set to 500, and a total of 30 runs for each algorithm are conducted.

To assure a fair comparison, we compare the average best fitness $\text{fit}_{\text{average}}$ values obtained by HPSO-GSA1, HPSO-GSA2, HPSO-GSA3, HPSO-GSA4, and HPSO-GSA with those obtained by the PSO. The comparison results are expressed according to the percentage improvement (%improve) computed as follows [36]:

$$\%\text{improve} = \frac{\text{fit}_{\text{average}}(\text{PSO}) - \text{fit}_{\text{average}}(\eta)}{\left|\text{fit}_{\text{average}}(\text{PSO})\right|} \times 100\%, \quad (16)$$

where η represents HPSO-GSA variants. If η has better performance (that is smaller average fitness) than PSO, %improve is positive. Otherwise, it is negative. The results of $\text{fit}_{\text{average}}$ and %improve for all involved algorithms are given in Table 4.

From Table 4, all of HPSO-GSA variants have obtained performance improvement in comparison with the PSO, implying that the utilization of any strategies, namely, DRC, FIIC, and AESC, can contribute to improving the PSOs searching accuracy. Among all the HPSO-GSA variants, the HPSO-GSA shows best performance according to the largest average %improve, followed by the HPSO-GSA4, HPSO-GSA3, HPSO-GSA2, and HPSO-GSA1. This conclusion suggests that the integration of three strategies into the

Table 4: F_{average} and %improve results obtained by PSO and HPSO-GSA variants for 10 dimensions in test functions.

Function	F_{average} (%improve)					
	PSO	HPSO-GSA1	HPSO-GSA2	HPSO-GSA3	HPSO-GSA4	HPSO-GSA
f_{Sph}	$1.68e-13$	$1.62e-13$ (3.02)	$1.56e-13$ (7.16)	$1.36e-13$ (20.04)	$1.02e-13$ (40.28)	**$2.61e-14$ (84.52)**
f_{Ros}	$2.92e+01$	$6.87e-01$ (97.26)	$4.63e-01$ (98.40)	$3.42e-03$ (100.00)	$3.56e-03$ (100.00)	**$2.18e-03$ (100.00)**
f_{Qua}	$8.63e+00$	$8.21e+00$ (4.86)	$6.52e+00$ (24.45)	$4.06e+00$ (52.95)	$3.62e+00$ (58.05)	**$1.06e+00$ (87.72)**
f_{Schw}	−3521.6	−3862.6 (9.68)	−4104.3 (16.55)	−4043.1 (14.81)	−4176.4 (18.59)	**−4189.8 (18.97)**
f_{Ras}	$6.04e-02$	$2.63e-02$ (56.46)	$3.62e-04$ (99.42)	$5.26e-04$ (99.15)	$6.02e-06$ (100.00)	**$5.08e-08$ (100.00)**
f_{Ack}	$2.68e-03$	$1.93e-03$ (27.99)	$3.72e-04$ (86.12)	$3.11e-06$ (99.88)	$8.63E-08$ (100.00)	**$7.82e-10$ (100.00)**
f_{Gri}	$4.28e-03$	$8.32e-04$ (80.56)	$2.36e-05$ (99.45)	$5.87e-06$ (99.86)	$2.31E-06$ (99.95)	**$2.08e-10$ (100.00)**
Average %improve		39.97	61.65	69.52	73.84	**84.45**

HPSO-GSA has significantly improved its optimizing performance. Hence, we use the HPSO-GSA for comparative study in the following section.

As shown in Table 4, HPSO-GSA3 and HPSO-GSA4 obtain higher average %improve than HPSO-GSA1 and HPSO-GSA2, especially in multimodal functions. This implies that the DRC strategy helps to escape from the local optima. On the other hand, HPSO-GSA2 and HPSO-GSA4 show better results against HPSO-GSA1 and HPSO-GSA3, respectively. This observation suggests that our AESC strategy performs better than the FIIC strategy. Finally, we observe some performance deteriorations in the functions f_{Schw} and f_{Ras} for HPSO-GSA2 and HPSO-GSA3, as the %improve value of HPSO-GSA3 is lower than that of HPSO-GSA2. It indicates that the AESC strategy can effectively improve the searching accuracy in certain functions in the absence of having the DRC strategy. The AESC strategy is more likely to help the HPSO-GSA to enhance the performance in some cases.

4.5. Comparative Study. To assess the performance of the HPSO-GSA compared to seven other optimization algorithms, the simulation experiments based on different measures are presented. These measures provide the ability to evaluate algorithms from different points. First, the performance results of the HPSO-GSA compared to PSO, GSA, and other state-of-the-art hybrid variants like DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA are presented in Section 4.5.1. Following the experiments, the statistical comparisons among the involved algorithms are also given to determine whether improvements of the proposed method are significant, as shown in Section 4.5.2.

4.5.1. Performance Results. The simulation results for each test function are recorded in Tables 5 and 6 based on three evaluation criteria such as accuracy, reliability, and efficiency by means of the mean best fitness (mean), standard deviation (std. dev.), success rate (SR), and searching time (ST). In these tables, mean is defined as the average result of best fitness generated by each algorithm for each function in 100 independent trials. A low mean is desirable as it indicates the algorithm has better optimizing accuracy. Std. dev. measures the amount of variation from the average. A small std. dev. indicates an algorithm has good stability. SR represents the consistency of an algorithm to achieve a predefined convergence level ε among the maximum iterations. A larger SR implies that an algorithm is more reliable as it can consistently solve a problem with the accuracy level ε. Finally, the computational cost can be evaluated by the mean ST which represents the algorithm's convergence speed with the predefined solution accuracy. The best results obtained by the algorithms are shown in bold for each metric.

From Table 5, we observe that HPSO-GSA has the lowest optimizing accuracy for solving all test functions except for functions quartic noise and Schwefel. Specifically, for the previous three unimodal functions, HPSO-GSA shows no much superior searching accuracy for function quartic noise, as the smallest mean values have been obtained by GA-GSA. The HPSO-GSA significantly outperforms other algorithms for function Rosenbrock. It implies that the proposed algorithm provides better exploration ability for avoiding the premature convergence, as the global optimum of this function is located in a narrow, long, and parabolic shaped flat valley. So it is often used to evaluate the ability of an algorithm in mitigating the stagnation problem. For the great majority of multimodal functions including complex hybrid composition, the HPSO-GSA surpasses all the other contenders as it has the smallest convergence accuracy. Hence, the HPSO-GSA can provide an appropriate level of global search escaping from many local optima.

Meanwhile, we present the SR and ST results produced by all the involved algorithms, as shown in Table 6, in order to compare the algorithm's reliability and computational cost, respectively. First, we observe that the HPSO-GSA and DE-GSA have more superior searching reliability than their peers for all the problems, as it converges successfully to the acceptable accuracy level with success rate 1. It is mentioned that the PSO never converges to the criteria for test functions at the predefined level ε. The remaining algorithms, especially the DEPSO, GAPSO, GA-GSA, and PSOGSA, are able to partially solve all the test functions. As a consequence, the proposed algorithm provides appropriate balance between exploration and exploitation abilities, guaranteeing to converge towards the predefined criteria. Second, as for the searching times, it is clear from Table 6 that the involved algorithms show various ST values for each test function. For instance, the computational overload of the PSO is the lowest, as it has advantage of fast convergence with simple implementation and global searching guide, whereas the second lowest ST values are achieved by the HPSO-GSA for all the employed functions. This is because GSA operator is

Table 5: Mean best fitness (mean) and standard deviation (std. dev.) results for benchmark test functions.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	Mean	$1.47e+00$	$5.72e-07$	$1.97e-02$	$2.06e-17$	$2.18e-23$	$5.18e-21$	$2.52e-23$	**$1.83e-24$**
	Std. dev.	$3.36e-01$	$1.46e-07$	$6.26e-03$	$2.82e-18$	$4.52e-23$	$6.22e-21$	$1.35e-23$	**$2.42e-24$**
f_{Ros}	Mean	$8.92e-01$	$2.62e-08$	$3.06e-10$	$3.69e-09$	$1.36e-13$	$1.11e-14$	$1.84e-16$	**$5.19e-19$**
	Std. dev.	$6.83e-01$	$3.26e-08$	$2.02e-10$	$6.26e-09$	$2.74e-13$	$2.41e-14$	$2.76e-17$	**$9.25e-20$**
f_{Qua}	Mean	$3.38e+00$	$1.36e-04$	$5.42e-02$	$4.05e-04$	$9.72e-04$	**$3.12e-16$**	$2.47e-06$	$2.12e-06$
	Std. dev.	$1.56e+00$	$8.54e-05$	$1.63e-02$	$4.61e-04$	$6.82e-04$	**$2.41e-16$**	$6.25e-06$	$8.35e-06$
f_{Schw}	Mean	$2.88e-02$	$3.81e+03$	$1.18e+02$	$8.72e+00$	**$2.61e-08$**	$2.88e-05$	$1.11e-02$	$2.55e-05$
	Std. dev.	$5.42e-01$	$5.52e+02$	$4.54e+02$	$3.65e+00$	**$9.52e-08$**	$3.51e-06$	$7.26e-02$	$1.37e-06$
f_{Ras}	Mean	$3.12e+02$	$5.87e+01$	$1.09e+01$	$1.76e-15$	$6.91e-12$	$1.98e+00$	$8.95e+00$	**$4.02e-22$**
	Std. dev.	$3.57e+01$	$5.52e+00$	$7.26e+00$	$3.25e-15$	$4.21e-12$	$4.56e-01$	$2.73e+00$	**$2.84e-22$**
f_{Ack}	Mean	$2.01e+01$	$3.56e-10$	$2.59e-09$	$2.66e-15$	$1.92e-10$	$8.77e-10$	$8.88e-16$	**$1.86e-19$**
	Std. dev.	$5.42e-00$	$4.65e-10$	$3.81e-09$	$6.26e-15$	$4.26e-10$	$2.15e-10$	$3.55e-16$	**$2.37e-19$**
f_{Gri}	Mean	$1.67e-01$	$1.97e-02$	$5.91e-02$	$1.21e-01$	$3.06e-07$	$7.39e-03$	$9.85e-03$	**$1.21e-09$**
	Std. dev.	$6.83e-02$	$8.21e-02$	$3.31e-02$	$6.24e-01$	$4.65e-07$	$1.26e-04$	$5.73e-03$	**$6.52e-09$**
f_{Pen1}	Mean	$2.68e+00$	$6.22e-01$	$6.39e-09$	$9.37e-23$	$3.26e-21$	$1.52e-$ **19**	$5.21e-23$	**$2.54e-32$**
	Std. dev.	$1.34e+00$	$6.37e-01$	$3.62e-09$	$5.36e-23$	$4.25e-21$	$5.62e-19$	$4.26e-23$	**$1.53e-32$**
f_{Pen2}	Mean	$3.11e-01$	$1.09e-02$	$1.29e-18$	$4.08e-18$	$2.18e-22$	$1.52e-08$	$2.86e-20$	**$1.25e-22$**
	Std. dev.	$9.64e-02$	$4.22e-02$	$4.62e-18$	$6.85e-18$	$3.25e-22$	$5.23e-08$	$3.65e-20$	**$9.64e-22$**
f_{Sch}	Mean	$9.72e-02$	$5.77e-04$	$1.57e-04$	$3.94e-09$	$1.54e-09$	$2.24e-05$	$1.99e-17$	**$9.13e-19$**
	Std. dev.	$2.23e-03$	$4.62e-04$	$3.64e-04$	$2.12e-09$	$6.73e-09$	$6.42e-05$	$2.12e-17$	**$8.35e-19$**

Best results are provided in bold.

Table 6: Success rate (SR) and searching time (ST, in seconds) results for benchmark test functions.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	SR	0.58	**1.00**	0.86	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**2.54**	15.62	18.46	12.62	22.85	20.26	20.24	4.83
f_{Ros}	SR	0.12	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**2.66**	16.06	26.21	16.52	25.85	21.34	19.42	6.33
f_{Qua}	SR	0.00	**1.00**	0.52	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**4.23**	23.74	22.46	16.88	30.22	24.62	25.42	8.32
f_{Schw}	SR	0.34	0.00	0.00	0.00	**1.00**	**1.00**	0.75	**1.00**
	ST	4.63	18.22	24.86	15.32	24.52	20.68	20.66	4.68
f_{Ras}	SR	0.00	0.00	0.00	**1.00**	**1.00**	0.00	0.00	**1.00**
	ST	**3.22**	19.68	24.63	16.22	26.53	23.52	20.08	5.26
f_{Ack}	SR	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**3.56**	18.86	20.31	13.94	25.44	20.28	19.51	4.36
f_{Gri}	SR	0.00	0.85	0.32	0.00	**1.00**	0.90	0.92	**1.00**
	ST	**3.34**	19.08	22.35	14.82	26.73	23.90	19.06	5.43
f_{Pen1}	SR	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**5.83**	20.52	16.85	19.73	25.06	22.52	22.05	9.17
f_{Pen2}	SR	0.00	0.26	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**4.32**	16.53	15.66	16.08	25.23	23.55	20.04	8.26
f_{Sch}	SR	0.72	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**0.92**	3.06	4.52	3.44	5.21	4.81	4.52	1.08

Best results are provided in bold.

added to our algorithm, resulting in the smaller computational overload. Other algorithms such as GSA, DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA require higher computational times compared to PSO and HPSO-GSA. The excellent performance of the HPSO-GSA in terms of accuracy and success rate also confirms that the HPSO-GSA is more computationally efficient than other hybrid variants.

To further evaluate the algorithm's convergence speed qualitatively, the convergence curves of all algorithms for all test functions are presented in Figure 5. The evolution tendency of an algorithm represents its convergence behavior and its convergence speed throughout the iterations. From Figure 5, the rapid convergence properties of HPSO-GSA are reflected by the convergence curves except for quartic noise and Schwefel functions; that is, HPSO-GSA

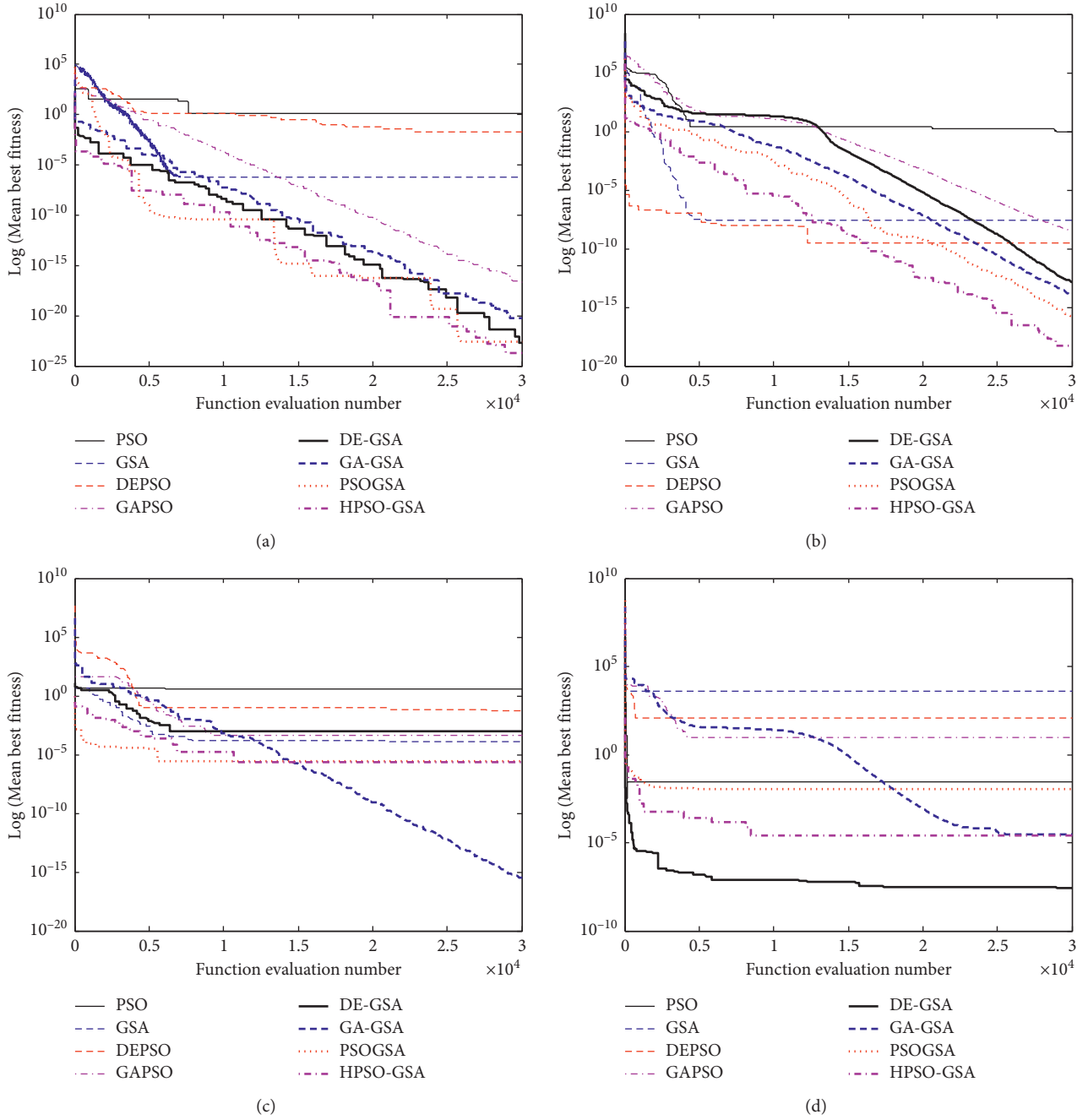

Figure 5: Continued.

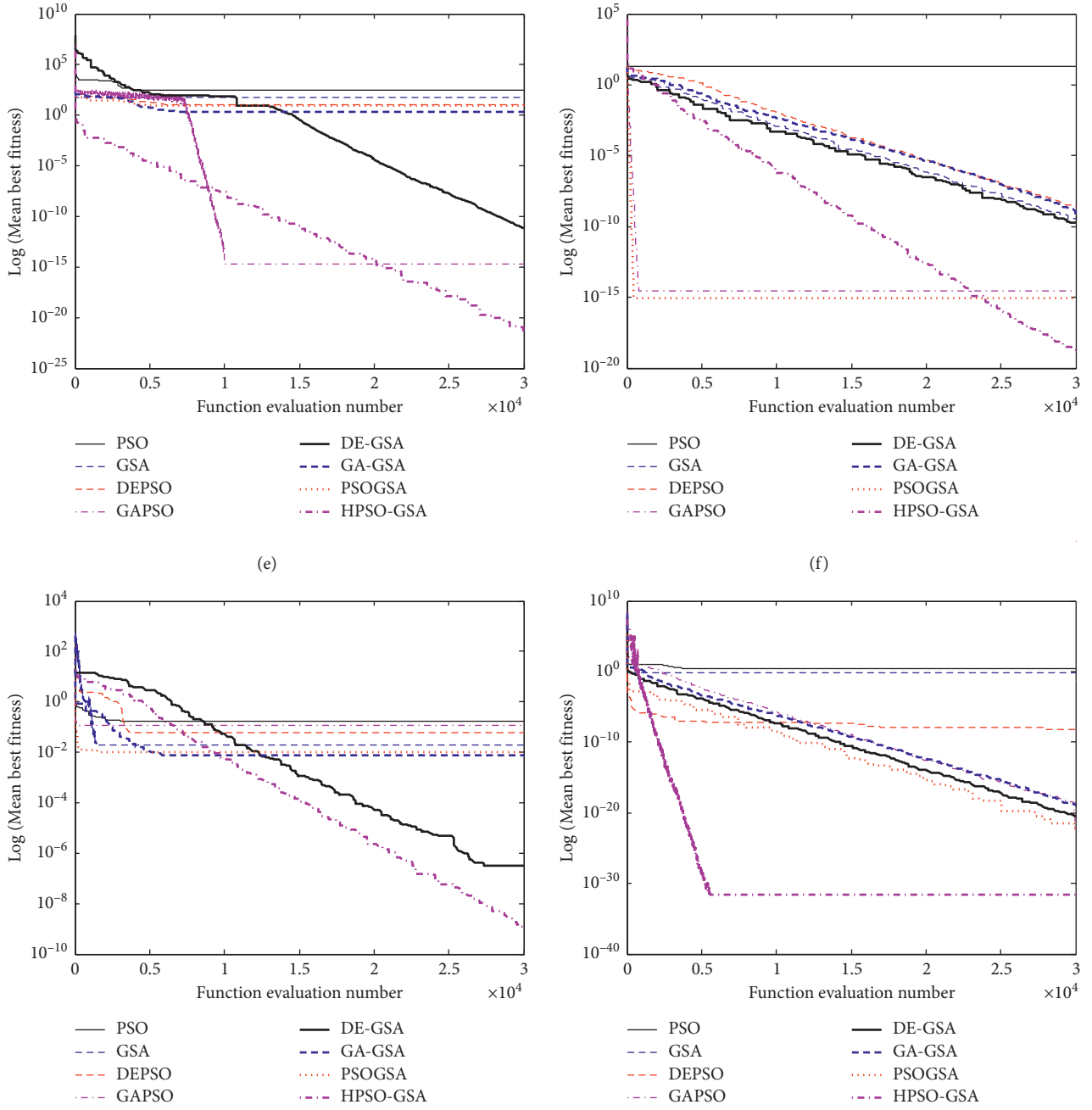

Figure 5: Continued.

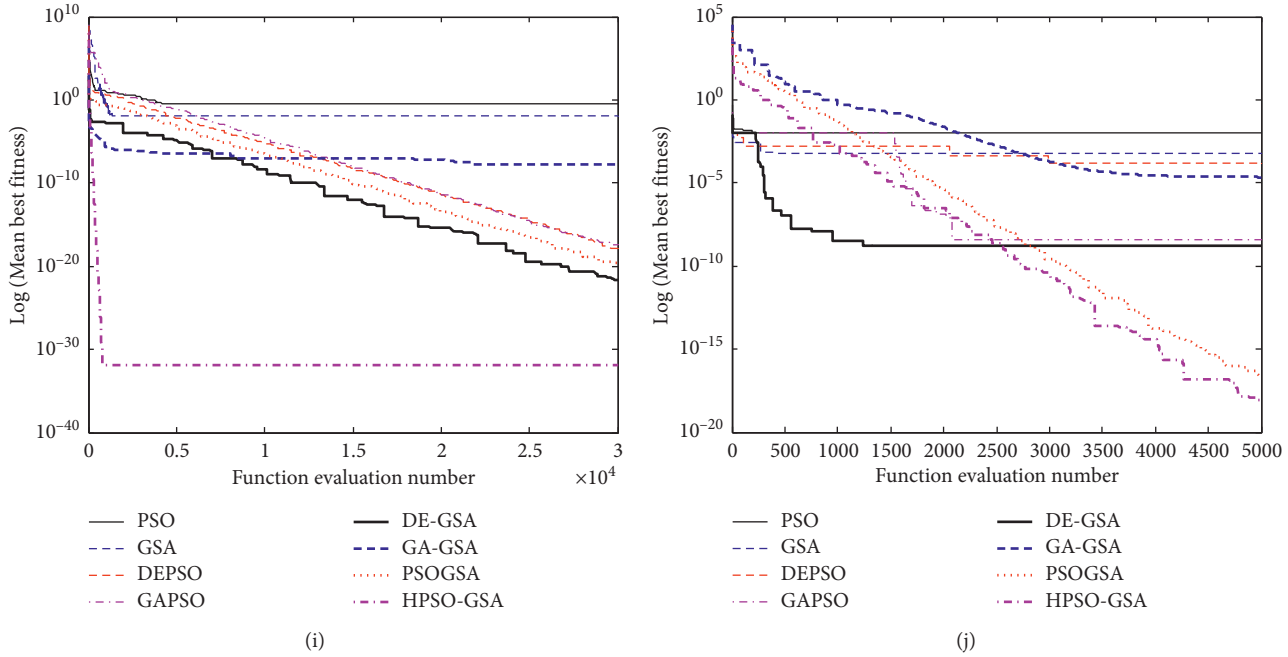

Figure 5: Convergence curves of mean best fitness obtained by all algorithms for benchmark test functions: (a) Sphere; (b) Rosenbrock; (c) quartic noise; (d) Schwefel; (e) Rastrigin; (f) Ackley; (g) Griewank; (h) Penalized1; (i) Penalized2; (j) Schaffer.

requires the less computational time to converge to acceptable level ε with less iteration. To be specific, PSO and DEPSO do not reach the specified convergence level if the function evaluation numbers are extended for function sphere. For Rosenbrock function, all the algorithms except for PSO converge to reach the criterion. Moreover, the best results are obtained by HPSO-GSA. As for multimodal functions, DE-GSA and HPSO-GSA show similar convergence curves for function Rastrigin; however, HPSO-GSA achieves better optimization accuracy. Meanwhile, GSA, DEPSO, DE-GSA, GA-GSA, and HPSO-GSA can continuously optimize the function Ackley throughout the function evaluation numbers, though they only provide a slower convergence speed. It reveals that the convergence accuracy may improve if the evolutionary process is continued. It is worth mentioning that more than half of hybrid variants including GAPSO, DE-GSA, GA-GSA, PSOGSA, and HPSO-GSA exhibit better convergence characteristics by extending the evolutionary process. They can converge to the predefined level ε in the early or middle stages of optimization. This implies that the incorporation of different algorithms can enhance optimization ability of the original algorithm, as it provides more mechanisms to mitigate the stagnation problem. Generally, the HPSO-GSA provides more competitive performance in terms of global accuracy and success rate, compared to other hybrid variants.

4.5.2. Statistical Comparisons of Different Algorithms. To thoroughly compare the HPSO-GSA with its competitors, we perform a two-tailed Taillard test (t-test) [42] with 58 degrees of freedom at a 0.05 level of significance and the Wilcoxon test [43]. Results of t-test (T) achieved by all the involved algorithms based on mean best fitness and success rate are reported in Tables 7 and 8, respectively. Additionally, we rank (R) the algorithms from the smallest mean value to the largest one for each function. The algorithms that are not statistically different from each other are given the same rank. The corresponding ranking R values are also listed in Table 7. To obtain the overall performance, we summarize the T values among HPSO-GSA and other peers as "+/ = /−" in the last row of the table. The "+/ = /−" denotes the number of test functions that HPSO-GSA performs significantly better, almost the same as and considerably worse than its competitors, respectively. Meanwhile, the overall average ranks over the number of test functions and the order of the average ranks are listed in Table 7.

From Table 7, we observe that the number of test functions where HPSO-GSA performs considerably better than its contenders (T = "+") is much larger than the number of test functions where HPSO-GSA obtains significantly worse results than its peers (T = "−"). Then, the best searching accuracy of HPSO-GSA among 8 algorithms is further validated by the t-test results. Particularly, the HPSO-GSA significantly surpasses all of its peers for functions Sphere, Rosenbrock, Rastrigin, Ackley, Griewank, Penalized1, Penalized2, and Schaffer. Moreover, the t-test results indicate that HPSO-GSA is statistically different in all the compared algorithms including PSO, GSA, DEPSO, and GAPSO. Also, based on the average ranks, the algorithms are sorted into the following order: HPSO-GSA, PSOGSA, DE-GSA, GA-GSA, GAPSO, DEPSO, GSA, and PSO. Due to the total and average ranks of HPSO-GSA are smaller than those of other algorithms, HPSO-GSA obtained a better overall performance than all other algorithms. Moreover, the

Table 7: Results of t-test (T) and ranking (R) of all the involved algorithms based on mean best fitness.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	T	+	+	+	+	+	+	+	
	R	8	6	7	5	2	4	2	1
f_{Ros}	T	+	+	+	+	+	+	+	
	R	8	7	5	6	4	3	2	1
f_{Qua}	T	+	+	+	+	+	−	=	
	R	8	4	7	4	4	1	2	2
f_{Schw}	T	+	+	+	+	−	=	+	
	R	6	8	6	5	1	2	4	2
f_{Ra}	T	+	+	+	+	+	+	+	
	R	8	6	6	2	3	4	4	1
f_{Ack}	T	+	+	+	+	+	+	+	
	R	8	4	7	3	4	4	2	1
f_{Gri}	T	+	+	+	+	+	+	+	
	R	7	5	5	7	2	3	3	1
f_{Pen1}	T	+	+	+	+	+	+	+	
	R	8	7	6	2	4	5	2	1
f_{Pen2}	T	+	+	+	+	+	+	+	
	R	8	7	4	4	1	6	3	1
f_{Sch}	T	+	+	+	+	+	+	+	
	R	8	6	6	3	3	5	2	1
Average ranks		8 (9.63)	7 (7.50)	6 (7.37)	5 (5.13)	3 (3.50)	4 (4.63)	2 (3.25)	**1 (1.50)**
+/=/−		**10/0/0**	**10/0/0**	**10/0/0**	**10/0/0**	9/0/1	8/1/1	9/1/0	

Table 8: Results of t-test (T) of all the involved algorithms based on success rate.

Function	PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	+	=	+	=	=	=	=	
f_{Ros}	+	=	=	=	=	=	=	
f_{Qua}	+	=	+	=	=	=	=	
f_{Schw}	+	+	+	+	=	=	+	
f_{Ras}	+	+	+	=	=	+	+	
f_{Ack}	+	=	=	=	=	=	=	
f_{Gri}	+	+	+	+	=	+	+	
f_{Pen1}	+	+	=	=	=	=	=	
f_{Pen2}	+	+	=	=	=	=	=	
f_{Sch}	+	=	=	=	=	=	=	
+/=/−	**10/0/0**	5/5/0	5/5/0	2/8/0	0/10/0	2/8/0	3/7/0	

analysis indicates that HPSO-GSA (with both the private thinking parts of the PSO and sequential mode) performs better than PSOGSA (without the personal thinking part of the PSO).

It is apparent from Table 8 that the number of test functions where HPSO-GSA performs considerably better than and almost the same as its contenders (T = "+" and "=") is much larger than the number of test functions where HPSO-GSA obtains significantly worse results than its peers (T = "−"). It reveals that HPSO-GSA is statistically different from GAPSO, DE-GSA, GA-GSA, and PSOGSA. If we increase the convergence level in the experiments, the "+" values in this table would increase and HPSO-GSA exhibits more excellent performance compared to other algorithms.

To compare the performance difference between HPSO-GSA and the other nine algorithms, we also conduct a Wilcoxon signed-rank test. Table 9 shows the resultant p values when comparing HPSO-GSA with other algorithms. The p values below 0.05 are shown in bold. The results show that HPSO-GSA is significantly better than other algorithms

Table 9: Results of Wilcoxon test between HPSO-GSA and other algorithms on test functions.

HPSO-GSA	p values
PSO	**0.003**
GSA	**0.016**
DEPSO	**0.012**
GAPSO	**0.036**
DE-GSA	0.263
GA-GSA	**0.042**
PSOGSA	0.292

The p values below 0.05 are shown in bold.

except for DE-GSA and PSOGSA. However, HPSO-GSA significantly outperforms DE-GSA and PSOGSA according to the searching accuracy in Table 5 and the average ranks in Table 7.

Based on the aforementioned performance evaluation and statistical results, we conclude that the proposed hybrid algorithm performs better overall in the involved test functions compared to PSO, GSA, DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA.

5. Conclusion

In this paper, a novel hybridization approach of PSO and GSA through sequential pattern, namely, HPSO-GSA, which consists of three learning strategies, dependent random coefficients, fixed iteration interval cycle, and adaptive evolution stagnation cycle, is proposed to solve the global optimization problems. The employment of the dependent random coefficients enhances the better balance between global and local searches, as it improves the diversity of the swarm. The fixed iteration interval cycle and adaptive evolution stagnation cycle is proposed for seamlessly integrating PSO and GSA in a less computational cost, thereby enhancing the algorithm's convergence speed. Meanwhile, the GSA operators encourage exploration and thus improve the premature stagnation problem. The experimental studies were performed to assess the performance of the proposed HPSO-GSA for solving benchmark test functions, as well as the impact of each employed strategy on the performance of the algorithm. The results indicate that the HPSO-GSA achieves better performance than its contenders investigated in this paper in terms of searching accuracy, algorithm reliability, and computational cost. Hence, the HPSO-GSA is a promising alternative solution to optimization problem. Possible future work includes extending the application of HPSO-GSA in the real-world optimization problem. In addition, we will investigate the suitable hybridization strategies to alleviate the stagnation tendency of HPSO-GSA.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant no. 61572238, Anhui Provincial Natural Science Foundation under Grant nos. 1608085QF157 and 1708085ME132, and Key Project of Natural Science Research of Anhui Provincial Department of Education under Grant no. KJ2016A431.

References

[1] A. Gupta, D. Singh, and M. Kaur, "An efficient image encryption using non-dominated sorting genetic algorithm-III based 4-D chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2020.

[2] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference Neural Networks*, IEEE Press, Perth, Australia, pp. 1942–1948, November 1995.

[3] M. Wang, S. Feng, C. He, Z. Li, and Y. Xue, "An artificial immune system algorithm with social learning and its application in industrial PID controller design," *Mathematical Problems in Engineering*, vol. 2017, Article ID 3959474, 13 pages, 2017.

[4] M. Kaur, H. K. Gianey, D. Singh, and M. Sabharwal, "Multi-objective differential evolution based random forest for e-health applications," *Modern Physics Letters B*, vol. 33, no. 5, Article ID 1950022, 2019.

[5] S. Tabakhi, A. Najafi, R. Ranjbar, and P. Moradi, "Gene selection for microarray data classification using a novel ant colony optimization," *Neurocomputing*, vol. 168, pp. 1024–1036, 2015.

[6] D. Nelson Jayakumar and P. Venkatesh, "Glowworm swarm optimization algorithm with topsis for solving multiple objective environmental economic dispatch problem," *Applied Soft Computing*, vol. 23, pp. 375–386, 2014.

[7] F. Wahid and D. H. Kim, "An efficient approach for energy consumption optimization and management in residential building using artificial bee colony and fuzzy logic," *Mathematical Problems in Engineering*, vol. 2016, Article ID 9104735, 13 pages, 2016.

[8] E. Rashedi, H. Nezamabadi-pour, and S. Saeid, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 1, pp. 2232–2248, 2009.

[9] T. Jiang, C. Zhang, and H. Zhu, "Energy-efficient scheduling for a job shop using grey wolf optimization algorithm with double-searching mode," *Mathematical Problems in Engineering*, vol. 2018, Article ID 8574892, 12 pages, 2018.

[10] L. Guo, Z. Meng, Y. Sun, and L. Wang, "Parameter identification and sensitivity analysis of solar cell models with cat swarm optimization algorithm," *Energy Conversion and Management*, vol. 108, no. 2, pp. 520–528, 2016.

[11] Z. Li and D. Li, "An improved global harmony search algorithm for the identification of nonlinear discrete-time systems based on volterra filter modeling," *Mathematical Problems in Engineering*, vol. 2016, Article ID 3102845, 13 pages, 2016.

[12] Y. Tian and Z. Lu, "Chaotic s-box: intertwining logistic map and bacterial foraging optimization," *Mathematical Problems in Engineering*, vol. 2017, Article ID 6969312, 11 pages, 2017.

[13] S. Mirjalili and S. Z. M. Hashim, "A new hybrid PSOGSA algorithm for function optimization,," in *Proceeding of the IEEE International Conference on Computer and Information Application*, pp. 374–377, Tianjin, China, 2010.

[14] H. Liu, Z. Cai, and Y. Wang, "Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization," *Applied Soft Computing*, vol. 10, no. 2, pp. 629–640, 2010.

[15] M.-R. Chen, X. Li, X. Zhang, and Y.-Z. Lu, "A novel particle swarm optimizer hybridized with extremal optimization," *Applied Soft Computing*, vol. 10, no. 2, pp. 367–373, 2010.

[16] H. S. Pannu, D. Singh, and A. K. Malhi, "Multi-objective particle swarm optimization-based adaptive neuro-fuzzy

inference system for benzene monitoring," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2195–2205, 2019.

[17] H. S. Pannu, D. Singh, and A. K. Malhi, "Improved particle swarm optimization based adaptive neuro-fuzzy inference system for benzene detection," *Clean-Soil, Air, Water*, vol. 46, no. 5, Article ID 1700162, 2018.

[18] G. Sun and A. Zhang, "A hybrid genetic algorithm and gravitational search algorithm for image segmentation using multilevel thresholding,," in *Pattern Recognition and Image Analysis*, Springer, Heidelberg, Germany, 2013.

[19] M. Kaur, D. Singh, and R. Singh Uppal, "Parallel strength pareto evolutionary algorithm-II based image encryption," *IET Image Processing*, 2019.

[20] S. Jiang, Z. Ji, and Y. Shen, "A novel hybrid particle swarm optimization and gravitational search algorithm for solving economic emission load dispatch problems with various practical constraints," *International Journal of Electrical Power & Energy Systems*, vol. 55, pp. 628–644, 2014.

[21] S. Mallick, S. P. Ghoshal, P. Acharjee, and S. S. Thakur, "Optimal static state estimation using improved particle swarm optimization and gravitational search algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 52, pp. 254–265, 2013.

[22] S. Mirjalili, S. Z. Mohd Hashim, and H. Moradian Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11125–11137, 2012.

[23] R. C. Green II, L. Wang, and M. Alam, "Training neural networks using central force optimization and particle swarm optimization: insights and comparisons," *Expert Systems with Applications*, vol. 39, no. 1, pp. 555–563, 2012.

[24] F. vandenBergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[25] R. Thangaraj, M. Pant, A. Abraham, and P. Bouvry, "Particle swarm optimization: hybridization perspectives and experimental illustrations," *Applied Mathematics and Computation*, vol. 217, no. 12, pp. 5208–5226, 2011.

[26] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization,," in *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1945–1950, Washington, DC, USA, 1999.

[27] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

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

[29] A. A. A. Esmin, G. Lambert-Torres, and G. B. Alvarenga, "Hybrid evolutionary algorithm based on PSO and GA mutation,," in *Proceedings of 6th International Conference on Hybrid Intelligent Systems*, pp. 57–62, Rio de Janeiro, Brazil, 2006.

[30] A. Shunmugalatha and S. M. R. Slochanal, "Optimum cost of generation for maximum load ability limit of power system using hybrid particle swarm optimization," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 8, pp. 486–490, 2008.

[31] W. J. Zhang and X. F. Xie, "DEPSO: Hybrid particle swarm with differential evolution operator," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMCC)*, pp. 3816–3821, Washington, DC, USA, 2003.

[32] C. Zhang, J. Ning, S. Lu, D. Ouyang, and T. Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization," *Operations Research Letters*, vol. 37, no. 2, pp. 117–122, 2009.

[33] I. Mendialdua, A. Arruti, E. Jauregi, E. Lazkano, and B. Sierra, "Classifier Subset Selection to construct multi-classifiers by means of estimation of distribution algorithms," *Neurocomputing*, vol. 157, pp. 46–60, 2015.

[34] H. W. Ge, L. Sun, Y. C. Liang, and F. Qian, "An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 38, no. 2, pp. 358–368, 2008.

[35] S. Jiang, Y. Wang, and Z. Ji, "A new design method for adaptive IIR system identification using hybrid particle swarm optimization and gravitational search algorithm," *Nonlinear Dynamics*, vol. 79, no. 4, pp. 2553–2576, 2015.

[36] W. H. Lim and N. A. Mat Isa, "Teaching and peer-learning particle swarm optimization," *Applied Soft Computing*, vol. 18, pp. 39–58, 2014.

[37] X. T. Li, M. Yin, and Z. Q. Ma, "Hybrid differential evolution and gravitation search algorithm for unconstrained optimization," *International Journal of Physical Sciences*, vol. 6, no. 25, pp. 5961–5981, 2011.

[38] H. Q. Chen, S. Li, and Z. Tang, "Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing," *International Journal of Computer Science and Network Security*, vol. 11, no. 6, pp. 208–217, 2011.

[39] S. Sarafrazi and H. Nezamabadi-pour, "Facing the classification of binary problems with a GSA-SVM hybrid system," *Mathematical and Computer Modelling*, vol. 57, no. 1-2, pp. 270–278, 2013.

[40] S. H. Jiang, C. L. Zhang, W. W. Wu, and Y. M. Li, "An improved hybrid particle swarm optimization with dependent random coefficients for global optimization," in *Proceedings of the 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 666–672, Nanjing, China, 2018.

[41] H. Huang, H. Qin, Z. Hao, and A. Lim, "Example-based learning particle swarm optimization for continuous optimization," *Information Sciences*, vol. 182, no. 1, pp. 125–138, 2012.

[42] H. Wang, Z. J. Wu, S. Rahnamayan, Y. Liu, and M. Ventresca, "Enhancing particle swarm optimization using generalized opposition-based learning," *Information Sciences*, vol. 181, no. 2, pp. 4699–4714, 2011.

[43] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.