IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2019, accepted May 29, 2019, date of publication June 20, 2019, date of current version July 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2923979

Memetic Particle Gravitation Optimization Algorithm for Solving Clustering Problems

KO-WEI HUANG, ZE-XUE WU, HSING-WEI PENG, MING-CHIA TSAI, YU-CHIEH HUNG, AND YU-CHIN LU

Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Kaohsiung 807, Taiwan

Corresponding author: Ko-Wei Huang (elone.huang@nkust.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, China, under Grant MOST 107-2218-E-992-003.

ABSTRACT Data clustering is a well-known data analysis technique for organizing unlabeled datapoints into clusters on the basis of similarity measures. The real-world applications of data clustering include bioinformatics, vector quantization, data mining, geographical information systems, pattern recognition, image processing, and wireless sensors. The data in a cluster are similar (minimizing the intra-cluster distance) and differ from the data in other clusters (maximizing the inter-cluster distance). The cluster problem has been proven to be NP-hard, but can be solved using meta-heuristic algorithms, such as ant colony optimization, genetic algorithms, gravitational search algorithm (GSA), and particle swarm optimization (PSO). This paper proposes a memetic clustering algorithm with efficient search and fast convergence, respectively, based on PSO and GSA, called the memetic particle gravitation optimization (MPGO) algorithm. The two main mechanisms of MPGO are hybrid operation and diversity enhancement. The former involves the exchange of individuals from two subpopulations after a predefined number of function evaluations (FEs), whereas the latter involves an enhancement operator, which is similar to the crossover process of differential evolution, for enhancing the diversity of each system. Individuals from the PSO and GSA systems are selected for the exchange of solutions by using the roulette-wheel approach. The performance of the proposed algorithm was evaluated on 52 benchmark test functions, six UCI machine learning benchmarks, and image segmentation of six well-known images. A comparison with existing algorithms verified the superior performance of the proposed algorithm in terms of a fitness value, an accuracy rate, and a peak signal-to-noise ratio.

INDEX TERMS Data clustering, gravitational search algorithm, image segmentation, memetic algorithm, particle swarm optimization.

I. INTRODUCTION

Data clustering has been attracting increasing attention in the field of data analysis. Typically, data clustering is used to organize data into relevant clusters on the basis of similarity criteria for identifying groupings that minimize intra-cluster distances and maximize inter-cluster distances. Each cluster comprises data that are similar and differ from data in other clusters, and the clustering problem has been proven to be an NP-hard problem [1], [2].

Clustering algorithms can be classified into two main types: hierarchical and partitional [3]. Hierarchical clusters can be presented as a dendrogram in which the input data are organized in a tree structure according to the agglomerative mode or divisive mode in a greedy manner [4]. A typical hierarchical algorithm comprises the following steps: (1) assigning of each datapoint to a separate cluster and (2) joining of the two most similar clusters until all the datapoints can be contained in a single cluster. Hierarchical clusters are employed in microarray informatics [5], [6] and signal processing [7], [8]. In contrast, partitional clusters are assigned according to the similarity between the data and each cluster centroid. The partitional approach has been one of the most important and complex research domains in data clustering since the proposal of the k-means algorithm [9]. The k-means algorithm involves the following steps: (1) determination of the initial K cluster centers with randomization, (2) assignment of each datapoint to the nearest cluster center, and (3) updating of the new cluster center. The algorithm repeats steps 2 and 3 until all the cluster centers stabilize. In recent years, the partitional approach has attracted considerable research attention [10]–[13].

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

Meta-heuristic algorithms have recently been extensively developed to solve large-scale NP-hard combinatorial optimization problems [14], [15]. Many of them are inspired by natural, physical, and biomedical phenomena, including the artificial bee colony (ABC) algorithm [16], bat algorithm (BA) [17], butterfly optimization algorithm (BOA) [18], crow search algorithm (CSA) [19], cuckoo search algorithm (CA) [20], differential evolution (DE) [21], elephant algorithm (EA) [22], farmland fertility algorithm (FFA) [23], firefly algorithm (FA) [24], grasshopper optimization algorithm (GOA) [25], gray wolf optimizer (GWO) [26], gravitational search algorithm (GSA) [27], lion optimization algorithm (LOA) [28], lightning search algorithm (LSA) [29], monkey algorithm (MA) [30], moth search (MS) [31], moth swarm algorithm (MSA) [32], particle swarm optimization (PSO) [33], sine cosine algorithm (SCA) [34], symbiotic organisms search (SOS) [35], shark smell optimization (SSO) [36], and whale optimization algorithm (WOA) [37].

In addition, meta-heuristic algorithms have attracted considerable research attention for use in partitional clustering models. These include ant colony optimization (ACO) [38], artificial bee colony algorithm (ABC) [39]–[41], bat algorithm (BA) [42], black hole (BH) [43], differential evolution (DE) [44], [45], elephant algorithm (EA) [46], firefly algorithm (FA) [47], [48], genetic algorithm (GA) [49], [50], k-means based genetic algorithm (GKA) [51], [52], gravitational search algorithm (GSA) [53], [54], gray wolf optimizer (GWO) [55], lion optimization algorithm (LOA) [56], monkey algorithm (MA) [57], moth swarm algorithm (MSA) [58], particle swarm optimization (PSO) [59], [60], simulated annealing (SA) [61], [62], symbiotic organism search (SOS) [63], and whale optimization algorithm (WOA) [64]. The clustering approach has been adopted in various research fields, such as bioinformatics [65]–[67], vector quantization [68]–[71], data mining [72]–[74], geographical information systems [75], [76], pattern recognition [77]–[79], and sensor applications [80]–[82]. Image segmentation is a well-known application of clustering in the field of computer vision [83]–[87], and it is an important research area in image processing. The objective of image segmentation is to partition pixels into several regions according their characteristics; all the pixels in a divided region are similar to each other but differ from those in other regions.

PSO is a well-known swarm-based intelligence algorithm that has the advantage of rapid convergence. However, this rapid convergence makes it susceptible to the critical issue of premature convergence during the evolutionary procedure when solving complex problems. In addition, the result of PSO is strongly dependent on inertial weight and social and cognitive coefficients. Consequently, the diversity of a population often decreases rapidly when approaching a global or local optimum, and in such cases, there is no efficient operator that can finetune the search space for PSO and improve the quality of its solution. GSA, a new meta-heuristic method based on the Newtonian laws of gravity and motion, has recently been proposed. Owing to the use of masses, GSA has a relatively high computing efficiency. Furthermore, previously published experimental studies have shown that GSA adopts an efficient search strategy for solving complex problems that enables it to achieve better performance than PSO [27]. However, when the population is in a convergence state, GSA exhibits poor performance and loses the ability to explore better solutions [88].

This study was conducted with the primary objective of developing a PSO and GSA hybrid that improves their individual search abilities, retains their advantages, and overcomes their disadvantages. To this end, this paper proposes a memetic clustering algorithm with the advantages of fast convergence based on PSO and efficient search based on GSA called the memetic particle gravitation optimization (MPGO) algorithm. The two main mechanisms of MPGO are hybrid operation and diversity enhancement. The former involves the exchange of individuals from two subpopulations after a predefined number of iterations, whereas the latter involves an enhancement operator, which is similar to the crossover process of DE, for enhancing the diversity of each system. Individuals from the PSO and GSA systems are selected for the exchange of solutions via the roulette-wheel approach [89].

The procedure employed by MPGO is as follows. First, the solution of each system is initialized randomly. Second, center particle swarm optimization (CPSO) [90] is adopted to obtain the center particle and center agent. Third, the PSO and GSA systems are executed simultaneously. Fourth, a global update hybridizes the PSO and GSA systems to enhance their exploitation and exploration abilities in order to obtain a better solution. Fifth, the diversity of the systems is enhanced using the crossover process of the DE algorithm [91]. Finally, specific individuals from the PSO and GSA systems are selected for exchange via roulette-wheel selection [89] after a predefined number of function evaluations (FEs). The performance of the proposed algorithm was compared with that of existing meta-heuristic algorithms on 52 benchmark test functions, six UCI machine learning benchmarks, and image segmentation of six well-known images. The results verified the superior performance of the proposed algorithm in terms of fitness value, accuracy rate, and peak signal-to-noise ratio (PSNR).

The remainder of this paper is organized as follows. Section II reviews background and related studies. Section III presents the problem definition of clustering. Section IV describes the proposed MPGO algorithm. Section V evaluates the performance of the proposed algorithm for data clustering and image segmentation. Finally, Section VI states the conclusions and briefly explores future research directions.

II. BACKGROUND KNOWLEDGE AND RELATED WORK

This section reviews relevant background and related studies, focusing on particle swarm optimization (PSO), gravitational search algorithm (GSA), and Center PSO (CPSO).

A. PARTICLE SWARM OPTIMIZATION

PSO is a stochastic swarm-based intelligence algorithm is inspired by the collective behavior of schools of fish or flocks of birds [33], [92]. It is inspired by the collective behavior of schools of fish or flocks of birds. In the PSO, the positions and velocities of N particles in d-dimensional space represent the potential solutions based on a randomly initialized. In PSO, the positions and velocities of N particles are represented in d-dimensional space to offer randomly initialized potential solutions. The solution for particle i in iteration t is given by Eq. (1). The current solution for each particle is updated with regard to the local and global optima, which are computed using Eqs. (2), and (3) respectively. Given N particles for which the positions are represented as potential solutions, the solution particle i in iteration t can be defined as follows:

$$X_i^t = \{x_{i,1}^t, x_{i,j}^t, \ldots, x_{i,d}^t\} \quad i = 1, 2 \ldots, N. \tag{1}$$

$$v_{i,j}^{t+1} = \omega v_{i,j}^t + c_1 r_1 (p_{i,j}^t - x_{i,j}^t) + c_2 r_2 (p_{gbest_j}^t - x_{i,j}^t) \tag{2}$$

$$x_{i,j}^{t+1} = x_{i,j}^t + v_{i,j}^{t+1} \tag{3}$$

where $x_{i,j}^t$ and $v_{i,j}^t$ denote the position and velocity, respectively, of particle i in dimension j, and ω is an inertial weight that influences the convergence speed. The local optimum p_i^t and global optimum p_{gbest}^t represent the current best position and best position in the swarm among all the particles at time t, respectively. The small constants c_1 and c_2 represent the cognitive parameter and social parameter, respectively, and r_1 and r_2 are random variables in the interval [0, 1].

B. GRAVITATIONAL SEARCH ALGORITHM

GSA is a population-based intelligence approach inspired by the laws of gravity and mass interactions [27]. In the GSA system, the mass aggregates are described as agents that achieve mutual interaction through Newtonian gravity and the laws of motion. First, each agent is randomly generated with a solution (called the position and velocity) by GSA. The algorithm computes the fitness values and updates the position and velocity of each agent among the current population. Then, the position of agent i in iteration t, which indicates a potential solution for N agents, is defined by Eq. (4):

$$X_i^t = \{x_{i,1}^t, x_{i,2}^t, x_{i,j}^t, \ldots, x_{i,d}^t\} \quad i = 1, 2, 3 \ldots, N. \tag{4}$$

where $x_{i,j}^t$ is the potential solution or position of agent i in dimension j, and d is the number of dimensions of the solution space.

The fitness can be evaluated in several steps, as expressed by Eqs. (5) to (10). First, the gravitation coefficient G^t in iteration t is calculated as follows:

$$G_t = G_o \times exp(-\beta \frac{t}{t_{max}}) \tag{5}$$

where β is the shrinking constant.

Next, the best and worst agents (denoted as $best^t$ and $worst^t$, respectively) are obtained using equations:

$$m_i^t = \frac{fit_i^t - worst^t}{best^t - worst^t} \tag{6}$$

$$M_i^t = \frac{m_i^t}{\sum_{c=1}^{N} m_c^t} \tag{7}$$

where fit_i^t represents the fitness value of agent i at time t. The masses and overall average mass are computed using Eqs. (6) and (7), respectively.

The total force acting on agent i from other agents is weighted randomly and calculated using Eqs. (8) and (9), respectively.

$$F_{i,j}^t = \sum_{kb \in Kbest, kb \neq j} rand_j \times \frac{Mpi^t * Mac^t}{R_{i,kb}^t + \varepsilon} (x_{c,j}^t - x_{i,j}^t) \tag{8}$$

$$M_{pi} = M_{ac} = M_{ii} = M_i \quad i = 1, 2, 3 \ldots, N. \tag{9}$$

where ε is a constant to avoid the division by zero exception, $rand_j$ is a random variable in the interval[0, 1], and $R_{i,kb}^t$ is the Euclidean distance between agent i and kb.

Finally, the acceleration in this iteration $a_{i,j}^t$ is computed using Eqs. (10) and (11):

$$a_{i,j}^t = \frac{F_{i,j}^t}{M_i^t} \tag{10}$$

$$a_{i,j}^t = \sum_{c \in Kbest, c \neq j} rand_j \times G_t \frac{Mc^t}{R_{i,c}^t + eps} (x_{c,j}^t - x_{i,j}^t) \tag{11}$$

The fittest agent $Kbest$ is the agent with the greatest mass among the top K agents. K is initialized to the number of agents N and decreases over time. Further, K will be updated in each iteration according to Eq. (12), as follows [93]:

$$K = \lfloor (\gamma + (1 - \frac{t}{t_{max}}))(1 - \gamma) N \rfloor \tag{12}$$

where γ imposes a controlled linear decrease on K. In the next iteration, the solution space of each agent is updated using Eqs. (13) and (14):

$$v_{i,j}^{t+1} = rand_j \times v_{i,j}^t + a_{i,j}^t \tag{13}$$

$$x_{i,j}^{t+1} = x_{i,j}^t + v_{i,j}^{t+1} \tag{14}$$

C. CENTRALIZED APPROACH

CPSO is an improved PSO approach involving a population of N particles, with their positions representing potential solutions [90]. After the positions of NâĹ'1 particles have been updated, a central individual ci is added to the population, as defined by Eq. (15):

$$x_{ci,j}^{t+1} = \frac{\sum_{i=1}^{N-1} x_{i,j}^t}{N - 1} \quad for\ j = 1, 2 \ldots, d. \tag{15}$$

where $x_{ci,j}^{t+1}$ is the position of the center particle in dimension j in iteration $t + 1$.

III. PROBLEM DEFINITION

Clustering is an unsupervised learning process in which data are classified into groups, where each cluster comprises similar data that differ from the data in other clusters. For a set of n patterns (datapoints) and k clusters, let $\tau = \{\tau_1, \tau_2, \tau_3, ..., \tau_n\}$, where $\tau_{i,j}$ denotes the j-th dimension of

the i-th pattern. Each pattern exists in d-dimensional space. The partitional clustering algorithm creates a partition of $C = \{C_1, C_2, C_3, ..., C_k\}$ clusters such that similar patterns are partitioned into the same cluster. The partition process is defined by Eq. (16) [1], [94]:

$$C_i \neq \phi \quad \forall i \in \{1, 2, 3, ..., k\}. \tag{16a}$$

$$C_i \cap C_j = \phi \quad \forall i \neq j, \ i, j \in \{1, 2, 3, ..., k\}. \tag{16b}$$

$$\cup_{i=1}^{k} C_i = \tau \tag{16c}$$

A. OBJECTIVE FUNCTION

The Euclidean distance between two patterns i and j can be calculated using Eq. (17):

$$d(\tau_i, \tau_j) = \sqrt{\sum_{f=1}^{d} (\tau_{i,f} - \tau_{j,f})^2} \tag{17}$$

To find the optimal grouping, in this study, we combine the inter-cluster distance, intra-cluster distance, and mean squared error (MSE) as a multi-objective problem that must be minimized. The three parts of the fitness function are expressed as follows in Eq. (18) [95], [96]:

$$f(C, Z) = \omega_1 \overline{d}_{max}(Z, C) + \omega_2(z_{max} - d_{min}(C)) + \omega_3 \text{MSE} \tag{18}$$

where Z represents all the patterns; $\overline{d}_{max}(Z, C)$ is the maximum average Euclidean distance within each cluster; $zmax$ is the maximum pattern value among all the patterns; $d_{min}(C)$ is the minimum Euclidean distance between any two clusters; ω_1, ω_2, and ω_3 are constants; and MSE represents the compactness of the clusters. MSE is defined as the mean squared error of the distance of the patterns from the centroid of the cluster to which they belong as follows in Eq (19):

$$\text{MSE} = \frac{\sum_{j=1}^{k} \sum_{\tau_p \in C_j} (\tau_p - m_j)^2}{N} \tag{19}$$

where τ_p represents the $p - th$ pattern and m_j is the $j - th$ centroid of cluster C_j. In this study, ω_1, ω_2, and ω_3 were set as 0.3, 0.3, and 0.4, respectively.

IV. PROPOSED ALGORITHM

Partitional clustering is an NP-hard problem that involves dividing n patterns into k clusters on the basis of a predefined similarity measure. Thus, a meta-heuristic algorithm is suitable for solving this type of problem. The proposed MPGO algorithm combines PSO and GSA with a hybrid operator and enhancement operator to determine the best partition for dividing each pattern with a suitable clustering center.

A. SOLUTION REPRESENTATION

To use a meta-heuristic algorithm for solving the clustering problem, all the individuals need to be encoded into the appropriate solution as cluster centers. As both PSO and GSA are population-based algorithms that were originally proposed to solve continuous problems, we can easily encode the cluster center of each particle and agent. First, each particle or agent, including the position and velocity, is generated randomly. Each solution for the individual consists of a vector whose size is equal to the number of dimensions multiplied by k, the number of cluster centroids. Figure 1 shows an example for two cluster centroids, where each cluster center has three dimensions of data. In this example, we assume three particles and three agents in each system. From the encoding form of Particle 1, we can easily determine that the solution should be encoded as (1.2, 2.3, 4.5) and (6.5, 2.6, 1.8), respectively. From the encoding form of Agent 1, the solution should be encoded as (5.1, 0.3, 2.7) and (1.5, 4.7, 2.3), respectively.

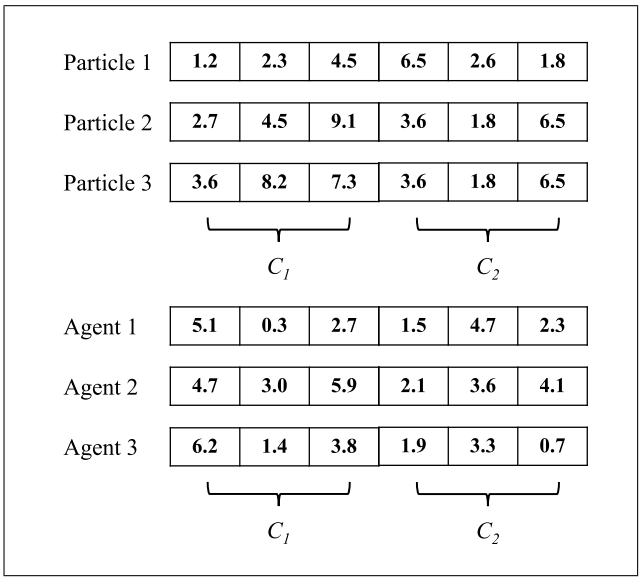

FIGURE 1. Example of solution encoding with two clusters and three dimensions.

B. INDIVIDUAL VELOCITY

Each individual (particle or agent) searches for and updates each solution according to Eqs. (2) and (13). In this study, the oscillations in the PSO and GSA systems are controlled by a time-varying maximum velocity V_{max}. The velocity thresholds [97] are expressed as follows in Eqs. (20) and (21):

$$V_{max} = (1 - (\frac{t}{t_{max}})^h) \times V_{max0} \tag{20}$$

$$V_{max0} = \alpha \times (x_{max} - x_{min}) \tag{21}$$

where the exponent h is a constant; α is used to control the maximum bounds of the search space, and x_{min} and x_{max} are the position thresholds set as 0.0 and 4.0, respectively, in this study. For example, if V_{max} is calculated as 6.0 and the velocity of Particle 1 is calculated as (1.0, 2.0, 3.0, 4.0, 5.0, 7.0), then according to the V_{max} value, the new velocity of Particle 1 will be (1.0, 2.0, 3.0, 4.0, 5.0, **6.0**).

C. GLOBAL UPDATE

This section presents the hybridization method, i.e., the global update, of PSO and GSA. Each particle and agent in

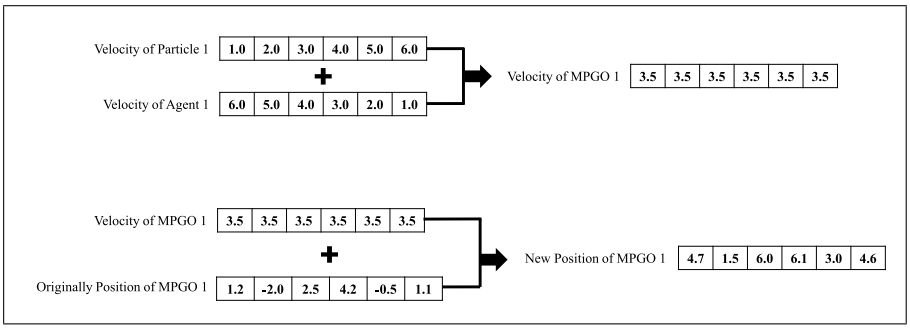

FIGURE 2. Example of the of global update process.

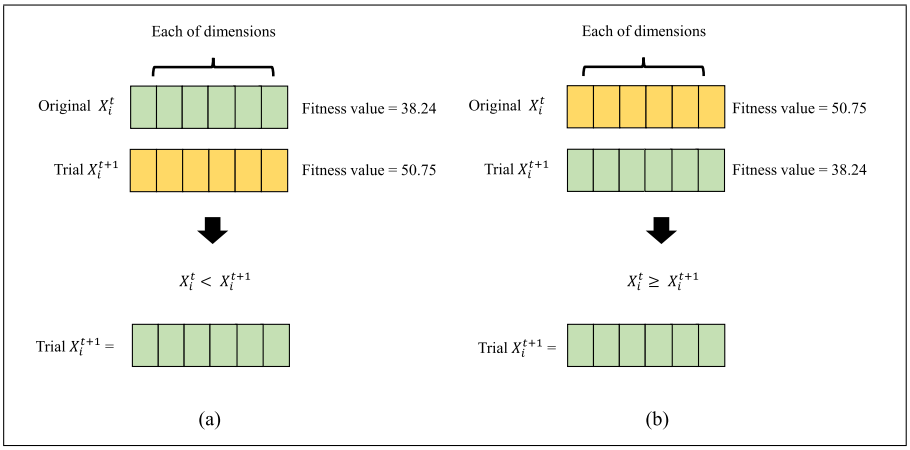

FIGURE 3. Example of the diversity enhancement operator.

the global update process, given by Eqs. (22) and (23), are integrated into the MPGO individual by combining the PSO velocity and GSA acceleration with social coefficients c_3 and c_4:

$$v_{i,j}^{t+1}(MPGO) = c_3 rand_i (v_{i,j}^{t+1})_{pso} + c_4(1 - rand_i)(v_{i,j}^{t+1})_{GSA} \tag{22}$$

$$x_{i,j}^{t+1}(MPGO) = x_{i,j}^{t}(MPGO) + (v_{i,j}^{t+1})(MPGO) \tag{23}$$

where c_3 and c_4 denote the cognitive parameter and social parameter, respectively, and $rand_i$ is a random number in the range [0,1].

Figure 2 shows an example of the global update process. In this example, we assume that the velocities of Particle 1 and Agent 1 are (1.0, 2.0, 3.0, 4.0, 5.0, 6.0) and (6.0, 5.0, 4.0, 3.0, 2.0, 1.0), respectively. According to Eq. (22), in step (a) we can obtain the new velocity of individual 1 of MPGO as (3.5, 3.5, 3.5, 3.5, 3.5, 3.5). In addition, we assume the original position of individual 1 of MPGO as (1.2, -2.0, 2.5, 4.2, -0.5, 1.1). According to Eq. (23), in step (b), we can obtain the new position of individual 1 of MPGO as (4.7, 1.5, 6.0, 6.1, 3.0, 4.6).

D. DIVERSITY ENHANCEMENT OPERATOR

Owing to its the fast convergence, the PSO algorithm may suffer from the critical issue of premature convergence during the evolutionary procedure when solving complex problems. Furthermore, when the population is in a convergence state, GSA exhibits poor performance and loses the ability to explore better solutions. Therefore, in this paper, we propose a diversity enhancement operator, which is similar to the crossover process of the DE algorithm [91], to enhance the diversity of each system. The main idea of the diversity enhancement operator is to replace the current solution by the previous one. For example, if the original particle (agent) is x_i^t and the new trial particle (agent) is x_i^{t+1}, the diversity enhancement operator will arrange the new trial particle (agent) in each dimension as follows in Eq. (24):

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} : & \text{if} \quad f(x_{i,j}^{t}) < f(x_{i,j}^{t+1}) \\ x_{i,j}^{t+1} : & \text{otherwise} \end{cases} \tag{24}$$

An example of the enhancement operator is shown in Figure 3. The figure shows two scenarios. In iteration t, assuming that the fitness value of individual i is 38.24, after the evolution in iteration $t + 1$, the fitness value is changed to 50.75. As the original individual is better than the trial one, the new trial solution will be replaced by the original one. On the other hand, assuming that the fitness value of individual i is 50.75, the new trial solution in the next iteration $t + 1$ is 38.24. According to the rule, the new trail solution is retained because it is better.

E. HYBRID OPERATOR

The diversity enhancement operator can improve the individual diversification of the PSO and GSA systems, but it cannot

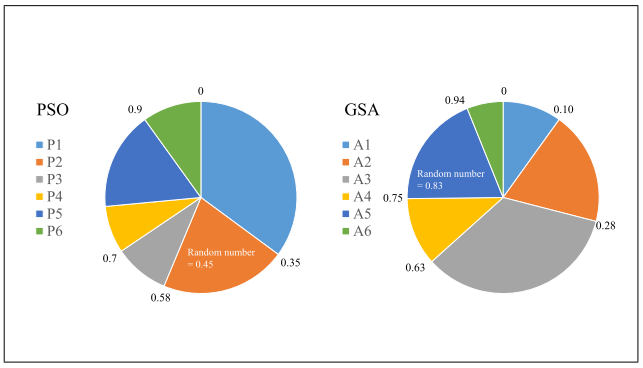

FIGURE 4. Example of the hybrid operator.

improve the overall diversity because the PSO and GSA systems easily fall into local optima when they are running independently. The main idea of the hybrid operator is to trigger the diversity of all the individuals of PSO and GSA. If we can exchange some individuals between the systems in a suitable period, the quality of the solution may be improved considerably. Thus, the hybrid operator is triggered after a specified number of function evaluations (FEs). At this point, certain individuals are selected and exchanged between the two systems via roulette-wheel selection [89], with probabilities that depend on their fitness values.

The roulette-wheel approach is expressed as follows in Eq. (25):

$$pn_i = \frac{fit_i}{\sum_{i=1}^{N} fit_i} \tag{25}$$

where pn_i represents the probability that each individual will be selected, and fit_i is the fitness value of particle/agent i.

An example of the hybrid operator is shown in Figure 4. In the PSO system, the random number is 0.45, which is located in region $P2$. In the GSA system, the random number is 0.83, which is located in region $A5$. Thus, in the hybrid operator, particle $P2$ and agent $A5$ will be selected for exchange between the two systems.

F. SUMMARY OF MPGO ALGORITHM

Figure 5 shows the procedure of the proposed MPGO algorithm. In the initial step, each particle (agent) is generated randomly. Second, CPSO is implemented to determine the center particle and agent. Third, the MPGO algorithm simultaneously executes the PSO algorithm and GSA along with the diversity enhancement operator. Fourth, a global update is performed by combining the PSO and GSA individuals to generate the new MPGO individual. Finally, the hybrid operator is triggered when a predefined maximum number of FEs is reached, and some individuals of the PSO and GSA systems will be exchanged via the roulette-wheel approach.

V. EXPERIMENTAL RESULTS

A. ENVIRONMENT SETTING

All the simulations were performed on a computer with an Intel Xeon E3-1225 (3.30 GHz) CPU and 16 GB main memory, running Windows 7 as the OS. All the programs were implemented in Python.

B. BENCHMARK FUNCTIONS

To evaluate the performance of the MPGO algorithm, we employed 30 simply test functions in our experiments. All the 30 simply benchmarks are summarized in Table 1 [98]–[100]. Here, the constant d denotes the number of dimensions of the function.

C. PARAMETER SETTINGS

The basic parameter settings of each algorithm are listed in Table 2. The parameters of the PSO and GSA systems are listed in the second and third rows, respectively, and the fourth row lists the parameters used in the global update (hybridization of the PSO and GSA systems). All the experimental results were collected from 20 independent runs.

According to the convergence curve for the selected test function f_2 shown in Figure 6, it is difficult for the function to find the optimal solution. Thus, in accordance with [101], we replaced the maximum number of iterations with maximum number of FEs for each experiment on the test function benchmarks. The performance of MPGO was compared with that of PSO [102], GSA [27], lightning search algorithm (LSA) [29], moth search (MS) [31], butterfly optimization algorithm (BOA) [18], symbiotic organisms search (SOS) [35], and moth swarm algorithm (MSA) [32]. The maximum number of (FEs) for all the benchmark test functions was 50000. The results indicated that for the PSO, BOA, LSA, and MS, the curve tends to fall gradually (convergence occurs after 25000 FEs), and for MPGO, the convergence speed remains in the descending state. On the basis of these experimental results the number of FEs was consequently set to 25000.

D. COMPARISON OF MPGO WITH SIMPLY TEST FUNCTIONS

1) COMPARISON RESULTS

In this section, the performance of MPGO is compared with that of PSO [102], GSA [27], lightning search algorithm (LSA) [29], moth search algorithm (MS) [31], butterfly optimization algorithm (BOA) [18], symbiotic organisms search (SOS) [35], and moth swarm algorithm (MSA) [32] for a maximum of 25000 function evaluations (FEs) on all the 30 simply benchmark test functions.

Table 3 lists the average best fitness value for each simulation. The mean values in bold font represent the algorithms that achieve superior performance. The proposed MPGO, MS and MSA showed the best overall performance. According to the results, MPGO is superior to PSO, GSA, LSA, BOA, and SOS on most of the functions, whereas it is inferior to LSA on functions f_{13} and f_{30}, and inferior to BOA on function f_{28}. MPGO is superior to MS, it performs worse than MS on function f_9, but outperforms MS on functions f_{18}, f_{26}, and f_{30}. In addition, MPGO is better than MSA,

TABLE 1. Test functions.

No.	Name	Formula	d	Search space	Optimal
f_1	Sphere	$f(x)=\sum_{i=1}^{d}x_i^2$	10	$[-100,100]^d$	0
f_2	Rosenbrock	$f(x)=\sum_{i=1}^{d-1}\left(100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right)$	10	$[-10,10]^d$	0
f_3	Ackley	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)$ $-\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right)+20+exp(1)$	10	$[-32,32]^d$	0
f_4	Griewank	$f(x)=\sum_{i=1}^{d}\frac{x_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	10	$[-600,600]^d$	0
f_5	Schwefel	$f(x)=\sum_{i=1}^{d}\lVert x_i\rVert+\prod_{i=1}^{d}\lVert x_i\rVert$	10	$[-10,10]^d$	0
f_6	Rastrigin	$f(x)=\sum_{i=1}^{d}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	10	$[-5.12,5.12]^d$	0
f_7	Cigar	$f(x)=x_1^2+10^6\sum_{i=2}^{d}x_i^2$	10	$[-100,100]^d$	0
f_8	Step	$f(x)=\sum_{i=1}^{d}(\lfloor x_i+0.5\rfloor)^2$	10	$[-100,100]^d$	0
f_9	Quartic	$f(x)=\sum_{i=1}^{d}ix_i^4+rand(0,1)$	10	$[-1.28,1.28]^d$	0
f_{10}	Alpine	$f(x)=\sum_{i=1}^{d}\lVert x_i\sin(x_i)+0.1x_i\rVert$	10	$[-10,10]^d$	0
f_{11}	Schwefel 1.2	$f(x)=\sum_{i=1}^{d}\left(\sum_{j=1}^{i}x_i\right)^2$	10	$[-10,10]^d$	0
f_{12}	Schwefel 2.21	$f(x)=max(\lVert x_i\rVert,1\le i\le d)$	10	$[-10,10]^d$	0
f_{13}	Schwefel 2.26	$f(x)=-\sum_{i=1}^{d}\left[x_i\sin(\sqrt{\lVert x_i\rVert})\right]$	10	$[-500,500]^d$	-418.982
f_{14}	Sum Squares	$f(x)=\sum_{i=1}^{d}ix_i^2$	10	$[-10,10]^d$	0
f_{15}	Trid	$f(x)=\sum_{i=1}^{d}(x_i-1)^2-\sum_{i=2}^{d}(x_ix_{i-1})$	10	$[-d^2,d^2]$	-210
f_{16}	Beale	$f(x)=(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2+(2.625-x_1+x_1x_2^3)^2$	2	$[-4.5,4.5]$	0
f_{17}	Bohachevsky	$f(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)-0.4\cos(4\pi x_2)+0.7$	2	$[-100,100]$	0
f_{18}	Levy	$f(x)=\sin^2(3\pi x_1)+(x_1-1)^2\left[1+sin^2(3x_2)\right]$ $+(x_1-1)^2\left[1+sin^2(2\pi x_2)\right]$	2	$[-10,10]$	0
f_{19}	Michalewicz	$f(x)=-\sum_{i=1}^{d}\sin(x_i)\left[\sin\left(\frac{ix_i^2}{\pi}\right)\right]^{2m}$; $m=10$	10	$[0,\pi]$	-9.66015
f_{20}	Schaffer	$f(x)=\left(x_1^2+x_2^2\right)^{0.25}\left[50(x_1^2+x_2^2)^{0.1}+1\right]$	2	$[-100,100]$	0
f_{21}	Easom	$f(x)=-cos(x_1)cos(x_2)exp\left[-(x_1-\pi)^2-(x_2-\pi)^2\right]$	2	$[-100,100]$	-1
f_{22}	Shubert	$f(x)=\left(\sum_{i=1}^{5}i\cos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5}i\cos((i+1)x_2+i)\right)$	2	$[-10,10]$	-186.7309
f_{23}	Booth	$f(x)=(x_1+2x_2-7)^2+(2x_1+x_2-5)^2$	2	$[-10,10]$	0
f_{24}	Goldstein price	$f(x)=\left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]\times$ $\left[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	2	$[-2,2]$	3
f_{25}	Matyas	$f(x)=0.26(x_1^2+x_2^2)-0.48x_1x_2$	2	$[-10,10]$	0
f_{26}	Powell	$f(x)=\sum_{i=1}^{d/4}(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2$ $+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}-x_{4i})^4$	32	$[-4,5]$	0
f_{27}	Power Sum	$f(x)=\sum_{i=1}^{d}\left[\left(\sum_{k=1}^{d}x_k^i\right)-b_i\right]^2$	4	$[0,d]$	0
f_{28}	Shekel 4.5	$f(x)=\sum_{i=1}^{d}\frac{1}{(x-A_i)^t(x-A_i)+c_i}$	4	$[0,10]$	-10.5364
f_{29}	Zettl	$f(x)=(x_1^2+x_2^2-2x_1)^2+0.25x_1$	2	$[-1,5]$	-0.00379
f_{30}	Leon	$f(x)=100(x_2-x_1^3)^2+(1-x_1)^2$	2	$[-1.2,1.2]$	0

Step	Description
1.	Initialize each solution of the PSO and GSA systems.
2.	Obtain the center particle and agent according to Eq. (15).
3.	Execute the PSO and GSA processes and apply the enhancement operator to increase diversity.
4.	Execute the global update to obtain the position and velocity in the MPGO algorithm.
5.	Apply the hybrid operator after a specified number of function evaluations.
6.	If the stopping criterion is not satisfied, repeat Steps 2 to 5.
7.	The best cluster center of the MPGO algorithm is determined.

FIGURE 5. Proposed MPGO algorithm.

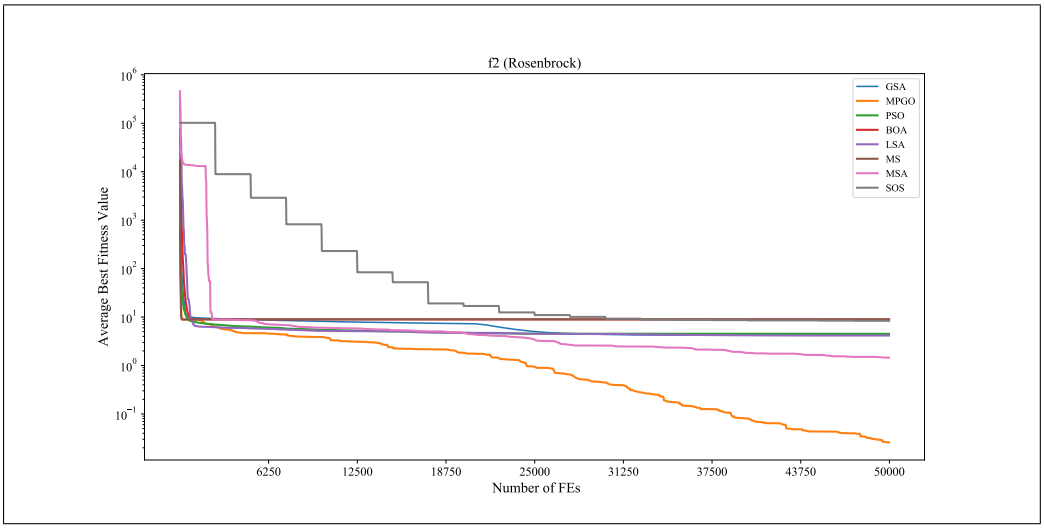

FIGURE 6. Comparison of convergence curves of MPGO with different meta-heuristic algorithms for function f_2 (Rosenbrock).

TABLE 2. Parameter settings for the proposed algorithm.

Method	Parameter	Value
PSO System	Number of Particles	25
	Inertia ω	0.9 to 0.2
	c_1	2
	c_2	2.4
GSA System	Number of Agents	25
	initial value of G	150
	β	20
	γ	0.02
	α	0.1
	h	0.05
	ε	0.00001
global update	c_3, c_4	1.4

it performs worse than MSA on functions f_{13}, f_{26} and f_{30}, but outperforms MSA on functions f_3, f_8, f_{18}, and f_{23}. Table 3 shows that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS and MSA for the overall performance evaluation. On the other hand, the performance of MPGO is degraded on functions f_9, f_{13}, and f_{30}.

Table 4 reports the results of two-sided Wilcoxon rank-sum tests [103] of MPGO, PSO, GSA, LSA, MS, BOA, SOS, and MSA at a significance level of $\alpha = 0.05$ on the basis of the performance results presented in Table 3. The Wilcoxon rank-sum test was conducted between MPGO and each compared algorithm on every test function. The sign + indicates that MPGO is significantly better than the compared algorithm, the sign - indicates that MPGO is significantly worse than the compared algorithm, and the sign $\simeq$ indicates that there is no significant difference between their performances. The results show that MPGO also dominates PSO, GSA, LSA, BOA and SOS. In addition, MPGO achieves slightly better results than the MS and MSA algorithm. In summary, Table 4 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation. On the other hand, the performance of MPGO is degraded on functions f_{28}, and f_{30}.

E. SCALABILITY OF MPGO WITH TEST FUNCTIONS ON MEDIAN DIMENSIONS

1) COMPARISON RESULT

In this section, the performances of MPGO is compared with the that of PSO [102], GSA [27], lightning search algorithm (LSA) [29], moth search algorithm (MS) [31], butterfly optimization algorithm (BOA) [18], symbiotic organisms search

TABLE 3. Comparison of MPGO versus various meta-heuristic algorithms.

Dataset	MPGO	PSO [102]	GSA [27]	LSA [29]	MS [31]	BOA [18]	SOS [35]	MSA [32]
f_1	0.0000E+00	1.6965E-08 (+)	2.4968E-11 (+)	4.6957E-97 (+)	0.0000E+00 (≃)	2.0579E-12 (+)	2.8246E+00 (+)	3.6135E-239 (+)
f_2	2.1972E+00	7.4576E+00 (≃)	6.5652E+00 (+)	4.3353E+00 (+)	8.8554E+00 (+)	8.9192E+00 (+)	1.2446E+01 (+)	3.0232E+00 (≃)
f_3	0.0000E+00	4.9573E+00 (+)	8.2315E-02 (+)	1.2867E+00 (+)	0.0000E+00 (≃)	1.8465E-09 (+)	1.7272E+00 (+)	8.8818E-16 (+)
f_4	0.0000E+00	6.6428E-01 (+)	1.1776E-01 (+)	1.3151E-01 (+)	0.0000E+00 (≃)	2.8297E-11 (+)	9.4177E-01 (+)	0.0000E+00 (≃)
f_5	0.0000E+00	7.2121E-01 (+)	5.6901E-06 (+)	1.2878E-06 (+)	0.0000E+00 (≃)	1.0524E-09 (+)	5.6983E-01 (+)	6.8613E-123 (+)
f_6	0.0000E+00	7.4277E+01 (+)	4.5113E+01 (+)	1.5024E+01 (+)	0.0000E+00 (≃)	1.9619E+01 (+)	4.1202E+01 (+)	0.0000E+00 (≃)
f_7	0.0000E+00	1.4032E-06 (+)	3.9396E+01 (+)	4.3471E-91 (+)	0.0000E+00 (≃)	3.2117E-12 (+)	1.3911E+06 (+)	1.0153E-237 (+)
f_8	0.0000E+00	8.8500E+00 (+)	0.0000E+00 (≃)	4.2944E-30 (+)	0.0000E+00 (≃)	1.0000E-01 (≃)	3.4412E+00 (+)	5.3786E-05 (+)
f_9	3.2951E-04	1.0019E-02 (+)	5.2551E-03 (+)	3.4701E-03 (+)	1.0512E-04 (-)	4.1697E-03 (+)	2.3410E-02 (+)	3.5276E-04 (≃)
f_{10}	0.0000E+00	6.9067E+00 (+)	2.6190E+00 (+)	6.8105E-09 (+)	0.0000E+00 (≃)	1.6494E-01 (+)	5.5679E-01 (+)	1.4105E-123 (+)
f_{11}	0.0000E+00	3.0260E-05 (+)	1.4531E-05 (+)	9.9245E-10 (+)	0.0000E+00 (≃)	1.8294E-12 (+)	1.0358E+00 (+)	4.9291E-201 (+)
f_{12}	0.0000E+00	1.0183E-03 (+)	7.9127E-07 (≃)	1.8363E-02 (+)	0.0000E+00 (≃)	1.1019E-10 (≃)	1.5576E-01 (+)	2.0276E-106 (+)
f_{13}	-2.6339E+03	-1.5627E+03 (+)	-1.4481E+03 (+)	-3.0646E+03 (-)	-1.3485E+03 (+)	-2.2268E+03 (+)	-1.5420E+03 (+)	-4.0640E+03 (-)
f_{14}	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	4.0888E-80 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.4479E-01 (+)	1.5416E-239 (+)
f_{15}	-2.1000E+02	-2.0597E+02 (+)	-2.0999E+02 (+)	-2.0937E+02 (+)	2.6257E+00 (+)	-8.4070E+00 (+)	-9.9127E+01 (+)	-2.1000E+02 (+)
f_{16}	0.0000E+00	2.2862E-01 (+)	1.1906E-01 (+)	1.9875E-32 (+)	2.4444E-01 (+)	2.0721E-01 (+)	2.1352E-04 (+)	0.0000E+00 (≃)
f_{17}	0.0000E+00	0.0000E+00 (≃)	6.6613E-17 (≃)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.4476E-01 (+)	1.5562E-03 (+)	0.0000E+00 (≃)
f_{18}	1.3498E-31	8.0564E-03 (≃)	6.1820E-03 (+)	1.3498E-31 (+)	2.4047E-01 (+)	6.0097E-03 (+)	1.8278E-03 (+)	1.0987E-02 (+)
f_{19}	-8.7569E+00	-4.4484E+00 (+)	-4.6153E+00 (+)	-8.8596E+00 (≃)	-3.6706E+00 (+)	-4.6720E+00 (+)	-5.0298E+00 (+)	-8.3204E+00 (+)
f_{20}	0.0000E+00	2.8434E-02 (+)	4.7740E-02 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.0914E-02 (+)	2.3722E-03 (+)	0.0000E+00 (≃)
f_{21}	-1.0000E+00	-5.3344E-06 (+)	-1.5623E-01 (+)	-7.0000E-01 (+)	-8.8447E-01 (+)	-3.8146E-01 (+)	-7.7510E-01 (+)	-9.0000E-01 (≃)
f_{22}	-1.8673E+02	-1.4800E+02 (+)	-1.4416E+02 (+)	-1.8673E+02 (≃)	-1.7188E+02 (+)	-1.8650E+02 (+)	-1.8100E+02 (+)	-1.8673E+02 (≃)
f_{23}	0.0000E+00	0.0000E+00 (≃)	3.8757E-17 (+)	0.0000E+00 (≃)	7.7451E-02 (+)	2.0814E-01 (+)	6.2439E-04 (+)	8.1208E-11 (+)
f_{24}	3.0000E+00	3.0622E+00 (≃)	5.8161E+00 (≃)	3.0000E+00 (≃)	2.9999E+00 (≃)	3.2361E+00 (+)	3.0026E+00 (+)	3.0000E+00 (≃)
f_{25}	0.0000E+00	6.3152E-67 (+)	2.5958E-18 (+)	7.8314E-235 (+)	0.0000E+00 (≃)	1.0636E-13 (+)	1.4472E-06 (+)	0.0000E+00 (≃)
f_{26}	-1.3490E-09	6.9342E-05 (+)	-2.9438E-04 (+)	2.6641E+00 (+)	3.0839E-06 (≃)	9.0831E-07 (≃)	4.5873E+00 (+)	3.4840E-209 (≃)
f_{27}	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	2.1071E-03 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	4.8188E-01 (+)	2.8566E-02 (+)
f_{28}	-7.2974E+00	-5.4891E+00 (+)	-8.7999E+00(-)	-7.1551E+00 (≃)	-2.4950E+00 (+)	-9.5767E+00(≃)	-5.8719E+00 (+)	-9.5664E+00 (-)
f_{29}	-3.7912E-03	-3.7912E-03(≃)	-3.7908E-03(≃)	-3.7912E-03 (+)	-3.7912E-03 (+)	-3.7912E-03 (+)	-3.7904E-03 (+)	-3.7912E-03 (+)
f_{30}	2.3936E-21	1.6928E-03 (+)	4.0088E-02 (+)	3.3656E-30 (-)	2.8741E-01 (+)	3.7240E-02 (+)	5.5421E-03 (+)	0.0000E+00 (-)

TABLE 4. Two-tailed Wilcoxon rank-sum test of MPGO versus compared algorithms with 10 dimensions.

Compared algorithm	Significantly better(+)	Significantly worse(-)	No significantly difference (≃)
(MPGO, PSO)	22	0	8
(MPGO, GSA)	22	1	7
(MPGO, LSA)	21	2	7
(MPGO, MS)	12	1	17
(MPGO, BOA)	24	0	6
(MPGO, SOS)	30	0	0
(MPGO, MSA)	15	3	12

(SOS) [35], and moth swarm algorithm (MSA) [32] ffor scalability to 30 and 50 dimensions for a maximum of 25000 FEs on benchmark functions f_1 to f_{15}.

For scalability of the median size to 30 and 50 dimensions, Table 5 lists the average best fitness value for each simulation. The proposed MPGO, MS, and MSA showed the best overall performance. According to the results, MPGO is superior to PSO, GSA, LSA, BOA, and SOS on most of the functions, whereas it is inferior to LSA on function f_{13}. MPGO is superior to MS, it performs worse than MS on function f_9, but outperforms on functions f_{13}, and f_{15}. In addition, it performs worse than MSA on functions f_{13}, and f_{15}, it outperforms MSA on functions f_2 and f_8. In summary, Table 5 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation, but is weak on functions f_9, f_{13}, and f_{15}.

Table 6 reports the results of two-sided Wilcoxon rank-sum tests [103] of MPGO, PSO, GSA, LSA, MS, BOA, SOS, and MSA at $\alpha = 0.05$ significance level on the basis of the performance results presented in Table 5. The Wilcoxon rank-sum test was performed between MPGO and each compared algorithm on every test function. The results show that MPGO also dominates PSO, GSA, LSA, BOA, SOS, and MSA algorithm. In addition, MPGO achieves slightly better results than MS. In summary, Table 6 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the

TABLE 5. Comparison of MPGO versus various meta-heuristic algorithms with larger dimensions (30, 50).

Dataset	Dims	MPGO	PSO [102]	GSA [27]	LSA [29]	MS [31]	BOA [18]	SOS [35]	MSA [32]
f_1	30	0.0000E+00	1.5300E-02 (+)	8.3938E-05 (+)	3.0706E-03 (+)	0.0000E+00 (≃)	2.8756E-12 (+)	6.1438E+01 (+)	4.7826E-232 (+)
	50	0.0000E+00	1.5238E-01 (+)	2.5169E-01 (+)	2.3378E-01 (+)	0.0000E+00 (≃)	3.0704E-12 (+)	1.8902E+02 (+)	4.7996E-216 (+)
f_2	30	2.2518E+01	5.8459E+01 (+)	2.7623E+01 (+)	7.4772E+01 (+)	2.8904E+01 (+)	2.8955E+01 (+)	1.2520E+02 (+)	2.6615E+01 (+)
	50	4.3341E+01	1.2586E+02 (+)	5.0143E+01 (+)	2.4590E+02 (+)	4.8923E+01 (+)	4.8945E+01 (+)	2.9934E+02 (+)	4.8083E+01 (+)
f_3	30	1.7764E-16	8.3398E+00 (+)	2.1190E+00 (+)	3.2290E+00 (+)	0.0000E+00 (≃)	1.4172E-09 (+)	3.3363E+00 (+)	1.0658E-15 (+)
	50	1.7764E-16	9.0498E+00 (+)	3.5610E+00 (+)	4.0765E+00 (+)	5.3291E-16 (≃)	1.4859E-09 (+)	3.8272E+00 (+)	8.8818E-16 (+)
f_4	30	0.0000E+00	9.0326E-02 (+)	3.1632E+00 (+)	1.0857E-02 (+)	0.0000E+00 (≃)	5.1296E-12 (+)	1.6994E+00 (+)	0.0000E+00 (≃)
	50	0.0000E+00	6.1179E-01 (+)	2.1707E+01 (+)	1.3445E-02 (+)	0.0000E+00 (≃)	4.7323E-12 (+)	2.4734E+00 (+)	0.0000E+00 (≃)
f_5	30	0.0000E+00	7.3368E+00 (+)	8.9361E-01 (+)	4.2114E-01 (+)	0.0000E+00 (≃)	1.2514E-08 (+)	3.8738E+00 (+)	1.3916E-121 (+)
	50	0.0000E+00	1.5688E+01 (+)	2.8541E+00 (+)	2.9425E+00 (+)	0.0000E+00 (≃)	3.0554E+23 (+)	7.3443E+00 (+)	1.7654E-119 (+)
f_6	30	0.0000E+00	2.8661E+02 (+)	2.0778E+02 (+)	7.4023E+01 (+)	0.0000E+00 (≃)	6.2713E+01 (+)	1.5322E+02 (+)	0.0000E+00 (≃)
	50	0.0000E+00	4.9061E+02 (+)	3.5943E+02 (+)	1.3590E+02 (+)	0.0000E+00 (≃)	7.6400E+01 (+)	2.7986E+02 (+)	0.0000E+00 (≃)
f_7	30	0.0000E+00	2.8948E+07 (+)	4.5995E+01 (+)	1.2319E+02 (+)	0.0000E+00 (≃)	3.6154E-12 (+)	6.2262E+07 (+)	2.3251E-226 (+)
	50	0.0000E+00	7.1949E+06 (+)	2.1217E+05 (+)	1.2261E+05 (+)	0.0000E+00 (≃)	3.7535E-12 (+)	1.5152E+08 (+)	1.9808E-221 (+)
f_8	30	0.0000E+00	2.2045E+02 (+)	0.0000E+00 (≃)	2.4565E-04 (+)	0.0000E+00 (≃)	5.0000E-02 (≃)	6.5035E+01 (+)	2.2747E-02 (+)
	50	0.0000E+00	1.6147E+03 (+)	1.4000E+00 (+)	1.1875E-01 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.8541E+02 (+)	1.0471E-01 (+)
f_9	30	2.6927E-04	8.2622E-02 (+)	2.3158E-02 (+)	2.9653E-02 (+)	6.6593E-05 (-)	4.3021E-03 (+)	4.7632E-02 (+)	3.7194E-04 (≃)
	50	1.3426E-04	2.6410E-01 (+)	4.9231E-02 (+)	8.6511E-02 (+)	8.5307E-05 (≃)	4.5370E-03 (+)	9.6093E-02 (+)	3.5491E-04 (+)
f_{10}	30	0.0000E+00	2.0799E+01 (+)	4.6505E+00 (+)	3.4325E-01 (+)	0.0000E+00 (≃)	6.2355E-09 (+)	2.5334E+00 (+)	1.8659E-122 (+)
	50	0.0000E+00	2.3602E+01 (+)	4.6568E+00 (+)	2.1189E+00 (+)	0.0000E+00 (≃)	5.4231E-09 (+)	3.3828E+00 (+)	5.4910E-119 (+)
f_{11}	30	0.0000E+00	2.1656E+00 (+)	1.2872E+00 (+)	2.2038E+00 (+)	0.0000E+00 (≃)	2.5571E-12 (+)	5.0992E+01 (+)	2.2533E-188 (+)
	50	0.0000E+00	1.1822E+01 (+)	7.0727E+00 (+)	3.1527E+01 (+)	0.0000E+00 (≃)	2.6892E-12 (+)	1.6049E+02 (+)	1.1111E-186 (+)
f_{12}	30	0.0000E+00	-1.9200E-01 (≃)	-1.0633E-04 (≃)	1.4971E+00 (+)	0.0000E+00 (≃)	3.6437E-10 (≃)	7.8693E-01 (+)	3.2991E-109 (+)
	50	-1.3820E-129	5.2541E-02 (≃)	3.9343E-02 (≃)	2.9591E+00 (+)	0.0000E+00 (≃)	3.7286E-10 (≃)	1.1183E+00 (+)	2.6706E-108 (+)
f_{13}	30	-6.4948E+03	-2.7604E+03 (+)	-2.4961E+03 (+)	-7.5009E+03 (-)	-2.2140E+03 (+)	-3.8674E+03 (+)	-2.5189E+03 (+)	-1.2437E+04 (-)
	50	-1.1175E+04	-3.7367E+03 (+)	-3.3168E+03 (+)	-1.1955E+04 (-)	-2.8332E+03 (+)	-5.0198E+03 (+)	-3.5036E+03 (+)	-2.0902E+04 (-)
f_{14}	30	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	2.5098E-01 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	8.8337E+00 (+)	9.7271E-230 (+)
	50	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	1.3551E+01 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	3.9909E+01 (+)	7.8613E-227 (+)
f_{15}	30	-2.4133E+03	-1.9007E+03 (≃)	6.6502E+04 (+)	-3.5526E+03 (-)	2.3740E+01 (+)	2.2195E+01 (+)	7.4829E+03 (+)	-4.8433E+03 (-)
	50	-1.7076E+03	5.0069E+03 (+)	1.8554E+07 (+)	-1.1318E+03 (≃)	4.4322E+01 (+)	4.2836E+01 (+)	1.2360E+05 (+)	-1.4712E+04 (-)

TABLE 6. Two-tailed Wilcoxon rank-sum test of MPGO versus compared algorithms with 30 and 50 dimensions.

Compared algorithm	Dims	Significantly better(+)	Significantly worse(-)	No significantly difference (≃)
(MPGO, PSO)	30	12	0	3
	50	13	0	2
(MPGO, GSA)	30	12	0	3
	50	13	0	2
(MPGO, LSA)	30	13	2	0
	50	13	1	1
(MPGO, MS)	30	3	1	11
	50	3	0	12
(MPGO, BOA)	30	12	0	3
	50	12	0	3
(MPGO, SOS)	30	15	0	0
	50	15	0	0
(MPGO, MSA)	30	10	2	3
	50	11	2	2

overall performance evaluation, and that MPGO can be scaled to higher dimensions (30 and 50).

F. SCALABILITY OF MPGO WITH TEST FUNCTIONS ON LARGE DIMENSIONS

1) COMPARISON RESULTS

In this section, the performances of MPGO is compared with the that of PSO [102], GSA [27], lightning search algorithm (LSA) [29], moth search algorithm (MS) [31], butterfly optimization algorithm (BOA) [18], symbiotic organisms search (SOS) [35], and moth swarm algorithm (MSA) [32] ffor scalability to 100 and 200 dimensions for a maximum of 25000 FEs on benchmark f_1 to f_{15}.

For the scalability of median size of 100 and 200 dimensions, Table 7 lists the average best fitness value for each simulation. The proposed MPGO, MS, and MSA showed the best overall performance. According to the results, MPGO is superior to PSO, GSA, LSA, BOA, and SOS on most of the functions, whereas it is inferior to LSA on function f_{13}. MPGO performs slightly better than MS. In addition,

TABLE 7. Comparison of MPGO versus various meta-heuristic algorithms with larger dimensions (100, 200).

Dataset	Dims	MPGO	PSO [102]	GSA [27]	LSA [29]	MS [31]	BOA [18]	SOS [35]	MSA [32]
f_1	100	0.0000E+00	3.2060E+01 (+)	8.4106E+01 (+)	2.8649E+00 (+)	0.0000E+00 (≃)	3.2227E-12 (+)	4.5389E+02 (+)	1.6256E-226 (+)
	200	0.0000E+00	1.9257E+03 (+)	2.1209E+03 (+)	1.8614E+01 (+)	0.0000E+00 (≃)	3.3910E-12 (+)	1.0050E+03 (+)	5.7905E-221 (+)
f_2	100	9.4269E+01	2.9365E+02 (+)	2.6692E+02 (+)	7.2631E+02 (+)	9.8944E+01 (+)	9.8955E+01 (+)	9.5727E+02 (+)	9.8179E+01 (+)
	200	1.9475E+02	3.2030E+03 (+)	1.6229E+03 (+)	9.5977E+02 (+)	1.9893E+02 (+)	1.9895E+02 (+)	2.3562E+03 (+)	1.9774E+02 (+)
f_3	100	1.3422E-09	1.0603E+01 (+)	5.0741E+00 (+)	7.2605E+00 (+)	2.1316E-15 (+)	1.5463E-09 (+)	4.0579E+00 (+)	8.8818E-16 (-)
	200	2.4332E-09	1.1830E+01 (+)	6.6425E+00 (+)	1.3088E+01 (+)	1.7764E-15 (≃)	1.5916E-09 (+)	4.2015E+00 (+)	8.8818E-16 (≃)
f_4	100	0.0000E+00	1.3091E+00 (+)	9.0865E+01 (+)	1.1370E-01 (+)	0.0000E+00 (≃)	3.9232E-12 (+)	5.6020E+00 (+)	0.0000E+00 (≃)
	200	0.0000E+00	1.5932E+01 (+)	2.4144E+02 (+)	6.0463E-01 (+)	0.0000E+00 (≃)	3.7487E-12 (+)	1.3872E+01 (+)	0.0000E+00 (≃)
f_5	100	0.0000E+00	3.4522E+01 (+)	1.3670E+01 (+)	1.8277E+01 (+)	0.0000E+00 (≃)	2.1475E+50 (+)	1.5378E+01 (+)	3.5665E-115 (+)
	200	0.0000E+00	5.5948E+01 (+)	4.4787E+01 (+)	4.6605E+01 (+)	0.0000E+00 (≃)	9.5176E+103 (+)	3.1596E+01 (+)	1.3947E-116 (+)
f_6	100	0.0000E+00	9.4403E+02 (+)	8.3233E+02 (+)	3.3262E+02 (+)	0.0000E+00 (≃)	7.1049E+01 (+)	5.2910E+02 (+)	0.0000E+00 (≃)
	200	0.0000E+00	1.9092E+03 (+)	1.7492E+03 (+)	8.9127E+02 (+)	0.0000E+00 (≃)	6.1508E-10 (+)	9.0247E+02 (+)	0.0000E+00 (≃)
f_7	100	0.0000E+00	1.9245E+07 (+)	1.0269E+08 (+)	3.5817E+06 (+)	0.0000E+00 (≃)	3.9713E-12 (+)	3.6702E+08 (+)	2.1467E-227 (+)
	200	0.0000E+00	1.5178E+09 (+)	2.1732E+09 (+)	1.7741E+07 (+)	0.0000E+00 (≃)	4.0313E-12 (+)	1.1356E+09 (+)	1.1977E-225 (+)
f_8	100	0.0000E+00	5.4509E+03 (+)	9.0550E+01 (+)	2.9120E+00 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	4.9241E+02 (+)	5.8711E-01 (+)
	200	0.0000E+00	1.1225E+04 (+)	2.4549E+03 (+)	1.8740E+01 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.3425E+03 (+)	2.4349E+00 (+)
f_9	100	2.1959E-04	7.0738E-01 (+)	1.8091E-01 (+)	6.6524E-01 (+)	9.3720E-05 (≃)	5.0705E-03 (+)	1.7571E-01 (+)	4.5570E-04 (+)
	200	4.4852E-04	2.5356E+00 (+)	5.2277E-01 (+)	5.7973E+01 (+)	7.0378E-05 (-)	4.8643E-03 (+)	4.8178E-01 (+)	3.3190E-04 (≃)
f_{10}	100	0.0000E+00	3.9324E+01 (+)	7.1913E+00 (+)	1.8236E+01 (+)	0.0000E+00 (≃)	4.2260E-09 (+)	5.8238E+00 (+)	1.2449E-119 (+)
	200	0.0000E+00	6.5236E+01 (+)	1.7916E+01 (+)	7.3245E+01 (+)	0.0000E+00 (≃)	2.6377E-09 (+)	1.1698E+01 (+)	4.8103E-118 (+)
f_{11}	100	0.0000E+00	3.7927E+02 (+)	3.2402E+01 (+)	3.6880E+02 (+)	0.0000E+00 (≃)	2.9197E-12 (+)	6.9631E+02 (+)	3.4357E-189 (+)
	200	0.0000E+00	1.7800E+03 (+)	1.2909E+02 (+)	1.8234E+03 (+)	0.0000E+00 (≃)	3.1141E-12 (+)	3.4230E+03 (+)	6.3400E-175 (+)
f_{12}	100	0.0000E+00	-5.5320E-01 (≃)	-1.3879E-01(≃)	5.3250E+00 (+)	0.0000E+00 (≃)	-1.1158E-10 (≃)	1.7099E+00 (+)	2.0413E-107 (+)
	200	0.0000E+00	8.3946E-02 (≃)	9.4088E-02 (≃)	6.8297E+00 (+)	1.9702E+00 (≃)	2.8477E-10 (≃)	3.0641E+00 (+)	3.5251E-106 (+)
f_{13}	100	-1.9140E+04	-5.0595E+03 (+)	-5.0447E+03(+)	-2.3497E+04 (-)	-4.0494E+03 (+)	-6.8550E+03 (+)	-5.0480E+03 (+)	-4.1808E+04 (-)
	200	-3.6388E+04	-7.3001E+03 (+)	-7.5360E+03(+)	-4.5921E+04 (-)	-5.5727E+03 (+)	-1.0184E+04 (+)	-6.7539E+03 (+)	-8.3633E+04 (-)
f_{14}	100	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	6.8269E+01 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	2.0883E+02 (+)	1.3154E-222 (+)
	200	0.0000E+00	0.0000E+00 (≃)	0.0000E+00 (≃)	1.2698E+02 (+)	0.0000E+00 (≃)	0.0000E+00 (≃)	1.1127E+03 (+)	2.0458E-230 (+)
f_{15}	100	-1.1374E+03	1.4911E+06 (+)	2.0539E+09 (+)	2.0645E+06 (+)	9.4615E+01 (+)	9.5286E+01 (+)	5.5334E+06 (+)	-1.1143E+04 (-)
	200	-7.9001E+02	2.5774E+08 (+)	8.3418E+10 (+)	4.2157E+08 (+)	1.9494E+02 (+)	1.9703E+02 (+)	1.8724E+08 (+)	-2.7984E+04 (-)

although it performs worse than MSA on functions f_3, f_{13} and f_{15}, it is better than MSA on functions f_1, f_2, f_5, f_7, f_8, f_{10}, f_{11}, f_{12} and f_{14}. In summary, Table 7 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation, whereas it is degraded on functions degraded on functions f_3, f_9, f_{13}, and f_{15}.

Table 8 reports the results of two-sided Wilcoxon rank-sum tests [103] of MPGO, PSO, GSA, LSA, MS, BOA, SOS, and MSA at $\alpha = 0.05$ significance level based on the performance results presented in Table 7. The results show that MPGO also dominates PSO, GSA, LSA, BOA, SOS, and MSA. In addition, MPGO achieves marginally better results than MS. In summary, Table 8 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation, and that MPGO can be scaled to higher dimensions (100 and 200).

G. COMPARISON OF MPGO ON CEC BENCHMARK

In this section, the performances of MPGO is compared with the that of PSO [102], GSA [27], lightning search algorithm (LSA) [29], moth search algorithm (MS) [31], butterfly optimization algorithm (BOA) [18], symbiotic organisms search (SOS) [35], and moth swarm algorithm (MSA) [32] for scalability to 10 and 100 dimensions for the 22 more complex CEC 2014/2017 [100], [104] testing benchmarks shown in Table 9. The maximum number of FEs for the CEC 2014/2017 benchmark functions with 10, 30, 50, and 100 decision dimensions were set to (10000 × number of dimensions); that is 100000, 300000, 500000, and 1000.000, respectively.

For scalability of the size to 10 and 30 dimensions, Table 10 lists the average best fitness value for each simulation. The proposed MPGO, and GSA showed the best overall performance. According to the results, MPGO is superior to PSO, LSA, MS, BOA, SOS and MSA on most of the functions. In addition, although MPGO performs worse than GSA on functions $f_{32}, f_{36}, f_{40}, f_{41}, f_{43}, f_{45}, f_{47}$ and f_{48}, it outperforms GSA on functions f_{35}, f_{46}, f_{50}, and f_{52}. In summary, Table 10 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation. whereas it is degraded on functions f_{32}, f_{36}, f_{45}, and f_{52}.

For the scalability of size 50 and 100 dimensions, Table 11 lists the average best fitness value for each simulation. The

TABLE 8. Two-tailed Wilcoxon rank-sum test of MPGO versus compared algorithms with 100 and 200 dimensions.

Compared algorithm	Dims	Significantly better(+)	Significantly worse(-)	No significantly difference ($\simeq$)
(MPGO, PSO)	100	13	0	2
	200	13	0	2
(MPGO, GSA)	100	13	0	2
	200	13	0	2
(MPGO, LSA)	100	14	1	0
	200	14	1	0
(MPGO, MS)	100	4	1	10
	200	3	1	11
(MPGO, BOA)	100	12	0	3
	200	12	0	3
(MPGO, SOS)	100	15	0	0
	200	15	0	0
(MPGO, MSA)	100	10	3	2
	200	9	2	4

TABLE 9. CEC2014/2017 Complex Test functions.

No.	Name	No.	Name
f_{31}	Rotated High Conditioned Elliptic Function	f_{42}	Shifted and Rotated Katsuura Function
f_{32}	Rotated Bent Cigar Function	f_{43}	Shifted and Rotated HappyCat Function
f_{33}	Rotated Discus Function	f_{44}	Shifted and Rotated HGBat Function
f_{34}	Shifted and Rotated Rosenbrock's Function	f_{45}	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function
f_{35}	Shifted and Rotated Ackley's Function	f_{46}	Shifted and Rotated Expanded Scaffer's F6 Function
f_{36}	Shifted and Rotated Weierstrass Function	f_{47}	Shifted and Rotated Bent Cigar Function
f_{37}	Shifted and Rotated Griewank's Function	f_{48}	Shifted and Rotated Sum of Different Power Function
f_{38}	Shifted Rastrigin's Function	f_{49}	Shifted and Rotated Zakharov Function
f_{39}	Shifted and Rotated Rastrigin's Function	f_{50}	Shifted and Rotated Lunacek Bi-Rastrigin Function
f_{40}	Shifted Schwefel's Function	f_{51}	Shifted and Rotated Non-Continuous Rastrigin's Function
f_{41}	Shifted and Rotated Schwefel's Function	f_{52}	Shifted and Rotated Levy Function

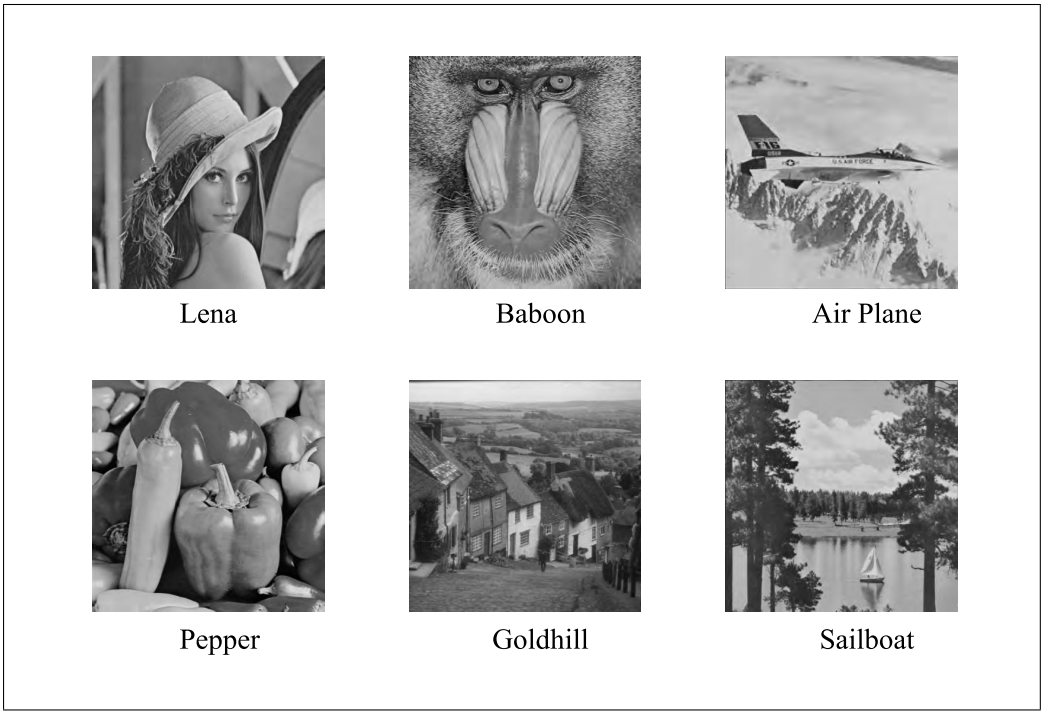

FIGURE 7. Six well-known images used for image segmentation.

proposed MPGO, GSA and MSA showed the best overall performance. According to the results, MPGO is superior to PSO, LSA, MS, BOA, and SOS on most of the functions. In addition, although MPGO performs worse than MSA on functions f_{44} and f_{45}, it is better than MSA on functions f_{31}, f_{38}, f_{39}, f_{42}, f_{46}, and f_{51}. In summary, Table 11 indicates that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the

TABLE 10. Comparison of MPGO versus different meta-heuristic algorithms with larger dimensions (10, 30).

Dataset	Dims	MPGO	PSO [102]	GSA [27]	LSA [29]	MS [31]	BOA [18]	SOS [35]	MSA [32]
f_{31}	10	5.87E+06	8.71E+07 (+)	1.47E+05 (+)	3.59E+03 (-)	1.05E+08 (+)	3.03E+08 (+)	7.80E+05 (+)	2.73E+05 (+)
	30	2.08E+06	1.35E+09 (+)	3.65E+05 (-)	1.56E+06 (≃)	1.66E+09 (+)	1.68E+09 (+)	1.14E+07 (+)	1.05E+05 (-)
f_{32}	10	3.21E+02	3.46E+09 (+)	2.29E+02 (≃)	1.17E+03 (+)	7.31E+09 (+)	6.60E+09 (+)	1.66E+04 (+)	3.74E+03 (+)
	30	1.19E+04	6.23E+10 (+)	7.21E+03 (≃)	1.20E+04 (≃)	7.59E+10 (+)	7.25E+10 (+)	2.03E+06 (+)	4.32E+03 (-)
f_{33}	10	6.27E+02	1.67E+04 (+)	6.93E+03 (+)	5.77E+02 (≃)	1.67E+04 (+)	1.35E+04 (+)	5.06E+03 (+)	2.37E+03 (+)
	30	5.79E+03	7.56E+04 (+)	1.89E+04 (+)	2.31E+03 (-)	8.19E+04 (+)	8.07E+04 (+)	1.65E+04 (+)	7.86E+02 (-)
f_{34}	10	2.00E+01	1.27E+03 (+)	2.64E+01 (+)	4.23E+02 (+)	2.12E+03 (+)	2.68E+03 (+)	4.17E+02 (+)	4.20E+02 (+)
	30	1.44E+02	1.01E+04 (+)	7.55E+01 (-)	4.96E+02 (+)	1.38E+04 (+)	1.53E+04 (+)	5.62E+02 (+)	4.16E+02 (+)
f_{35}	10	2.00E+01	2.06E+01 (+)	2.05E+01 (+)	5.20E+02 (+)	2.04E+01 (+)	2.04E+01 (+)	5.20E+02 (+)	5.20E+02 (+)
	30	2.02E+01	2.11E+01 (+)	4.44E+01 (+)	5.20E+02 (+)	2.10E+01 (+)	2.09E+01 (+)	5.21E+02 (+)	5.20E+02 (+)
f_{36}	10	7.87E+00	1.13E+01 (+)	2.45E-01 (+)	6.05E+02 (+)	1.09E+01 (+)	6.82E+00 (-)	6.03E+02 (+)	6.08E+02 (+)
	30	3.59E+01	4.25E+01 (+)	6.03E-03 (+)	6.25E+02 (+)	4.32E+01 (+)	3.21E+01 (-)	6.16E+02 (+)	6.07E+02 (+)
f_{37}	10	7.69E-01	1.35E+02 (+)	1.13E+01 (-)	7.00E+02 (+)	1.75E+02 (+)	2.02E+02 (+)	7.01E+02 (+)	7.04E+02 (+)
	30	1.37E-01	6.05E+02 (+)	3.48E+02 (-)	7.00E+02 (+)	7.17E+02 (+)	8.05E+02 (+)	7.01E+02 (+)	7.04E+02 (+)
f_{38}	10	3.08E+01	8.97E+01 (+)	2.45E-01 (+)	8.27E+02 (+)	9.33E+01 (+)	5.07E+01 (+)	8.34E+02 (+)	8.20E+02 (+)
	30	1.31E+02	3.32E+02 (+)	3.65E+02 (+)	9.45E+02 (+)	3.76E+02 (+)	2.43E+02 (+)	9.52E+02 (+)	8.18E+02 (+)
f_{39}	10	3.77E+01	9.48E+01 (+)	9.39E+01 (+)	9.38E+02 (+)	7.43E+01 (+)	4.24E+01 (+)	9.44E+02 (+)	9.36E+02 (+)
	30	1.67E+02	3.76E+02 (+)	7.56E+03 (+)	1.14E+03 (+)	3.61E+02 (+)	2.63E+02 (+)	1.13E+03 (+)	9.30E+02 (+)
f_{40}	10	6.30E+02	1.58E+03 (+)	8.71E+01 (+)	1.49E+03 (+)	1.81E+03 (+)	1.25E+03 (+)	1.81E+03 (+)	1.33E+03 (+)
	30	4.66E+03	7.26E+03 (+)	7.84E+03 (+)	4.27E+03 (≃)	8.02E+03 (+)	6.45E+03 (+)	5.20E+03 (≃)	1.34E+03 (-)
f_{41}	10	1.00E+03	1.87E+03 (+)	1.83E+03 (+)	2.06E+03 (+)	1.51E+03 (+)	1.19E+03 (+)	2.54E+03 (+)	2.00E+03 (+)
	30	5.34E+03	8.19E+03 (+)	4.02E+00 (+)	5.14E+03 (≃)	7.80E+03 (+)	6.87E+03 (+)	7.45E+03 (+)	1.99E+03 (-)
f_{42}	10	7.36E-01	3.19E+00 (+)	1.72E+03 (+)	1.20E+03 (+)	1.62E+00 (+)	1.25E+00 (+)	1.20E+03 (+)	1.20E+03 (+)
	30	1.32E+00	4.37E+00 (+)	4.04E-01 (+)	1.20E+03 (+)	3.02E+00 (+)	2.66E+00 (+)	1.20E+03 (+)	1.20E+03 (+)
f_{43}	10	6.88E-01	2.93E+00 (+)	2.56E+00 (-)	1.30E+03 (+)	4.09E+00 (+)	4.91E+00 (+)	1.30E+03 (+)	1.30E+03 (+)
	30	4.28E-01	7.28E+00 (+)	2.42E-01 (-)	1.30E+03 (+)	8.21E+00 (+)	9.10E+00 (+)	1.30E+033 (+)	1.30E+03 (+)
f_{44}	10	4.21E-01	2.26E+01 (+)	4.02E-01 (-)	1.40E+03 (+)	3.14E+01 (+)	4.09E+01 (+)	1.40E+03 (+)	1.40E+03 (+)
	30	3.21E-01	2.28E+02 (+)	1.78E+01 (-)	1.40E+03 (+)	2.79E+02 (+)	2.97E+02 (+)	1.40E+03 (+)	1.40E+03 (+)
f_{45}	10	4.82E+00	7.15E+02 (+)	3.84E-01 (≃)	1.50E+03 (+)	3.26E+03 (+)	5.30E+03 (+)	1.50E+03 (+)	1.52E+03 (+)
	30	6.90E+01	1.93E+05 (+)	1.34E+01 (-)	1.52E+03 (+)	3.75E+05 (+)	3.34E+05 (+)	1.54E+03 (+)	1.52E+03 (+)
f_{46}	10	3.19E+00	4.01E+00 (+)	3.84E+00 (+)	1.60E+03 (+)	3.91E+00 (+)	3.38E+00 (≃)	1.60E+03 (+)	1.60E+03 (+)
	30	1.25E+01	1.37E+01 (+)	1.34E+01 (+)	1.61E+03 (+)	1.36E+01 (+)	1.27E+01 (≃)	1.61E+03 (+)	1.60E+03 (+)
f_{47}	10	1.38E+02	7.94E+09 (+)	3.79E+00 (-)	5.05E+03 (+)	1.33E+10 (+)	1.20E+10 (+)	1.43E+04 (+)	
	30	2.85E+03	4.50E+10 (+)	2.04E+03 (≃)	2.83E+05 (+)	5.85E+10 (+)	5.20E+10 (+)	1.41E+06 (+)	
f_{48}	10	1.32E-04	1.18E+13 (+)	2.42E+00 (≃)	5.71E+03 (+)	4.45E+13 (+)	1.14E+11 (+)	7.18E+03 (+)	
	30	8.27E+26	1.68E+48 (+)	7.35E+00 (+)	2.73E+03 (+)	1.81E+52 (+)	9.11E+52 (+)	6.64E+03 (+)	
f_{49}	10	1.78E-09	1.34E+04 (+)	3.93E+02 (+)	1.90E+03 (+)	1.42E+04 (+)	1.51E+04 (+)	1.90E+03 (+)	
	30	1.23E+04	1.02E+05 (+)	2.94E+04 (+)	1.92E+03 (+)	8.61E+04 (+)	7.98E+04 (+)	1.92E+03 (+)	
f_{50}	10	5.14E+01	7.38E+01 (+)	1.12E+02 (+)	2.06E+03 (+)	1.31E+02 (+)	9.12E+01 (+)	5.52E+03 (+)	
	30	4.57E+02	5.52E+02 (+)	5.28E+02 (+)	2.54E+03 (+)	7.43E+02 (+)	6.58E+02 (+)	2.21E+04 (+)	
f_{51}	10	4.28E+01	9.43E+01 (+)	7.43E+01 (+)	3.24E+03 (+)	7.36E+01 (+)	3.74E+01 (+)	6.11E+03 (+)	
	30	1.46E+02	3.70E+02 (+)	3.97E+02 (+)	7.52E+04 (+)	3.68E+02 (+)	2.55E+02 (+)	2.07E+05 (+)	
f_{52}	10	2.57E+02	9.28E+02 (+)	7.72E+02 (+)	2.28E+03 (+)	9.37E+02 (+)	6.17E+02 (+)	2.25E+03 (+)	
	30	3.15E+03	9.20E+03 (+)	9.31E+03 (+)	2.86E+03 (≃)	1.13E+04 (+)	8.52E+03 (+)	2.78E+03 (≃)	

overall performance evaluation. On the other hand, the performance of MPGO is degraded on functions f_{31}, f_{32}, f_{33}, and f_{48}.

Table 12 and Table 13 reports the results of the two-sided Wilcoxon rank-sum tests [103] of MPGO, PSO, GSA, LSA, MS, BOA, SOS, and MSA at the $\alpha = 0.05$ significance

TABLE 11. Comparison of MPGO versus different meta-heuristic algorithms with larger dimensions (50, 100).

Dataset	Dims	MPGO	PSO [102]	GSA [27]	LSA [29]	MS [31]	BOA [18]	SOS [35]	MSA [32]
f_{31}	50	5.00E+06	3.85E+09 (≃)	6.40E+06 (≃)	2.85E+06 (-)	1.69E+09 (+)	5.82E+09 (+)	2.52E+07 (+)	5.43E+04 (+)
	100	5.73E+07	7.51E+09 (+)	7.22E+06 (-)	2.53E+07 (-)	1.03E+10 (+)	9.81E+09 (+)	1.24E+08 (+)	3.14E+04 (-)
f_{32}	50	6.33E+06	1.32E+11 (+)	1.96E+09 (+)	1.16E+05 (-)	1.59E+11 (+)	1.50E+11 (+)	1.67E+08 (+)	6.82E+03 (-)
	100	1.08E+08	2.82E+11 (+)	2.06E+04 (-)	4.13E+05 (-)	3.05E+11 (+)	2.94E+11 (+)	4.05E+09 (+)	5.77E+03 (-)
f_{33}	50	4.37E+04	1.50E+05 (-)	2.97E+03 (-)	2.55E+03 (-)	1.49E+05 (+)	1.44E+05 (+)	5.14E+04 (≃)	4.34E+02 (-)
	100	1.20E+05	3.72E+05 (+)	1.03E+05 (≃)	5.80E+03 (-)	3.10E+05 (+)	2.99E+05 (+)	1.06E+05 (≃)	3.11E+02 (-)
f_{34}	50	1.35E+02	3.71E+04 (+)	4.65E+02 (+)	5.63E+02 (+)	4.78E+04 (+)	5.13E+04 (+)	6.94E+02 (+)	4.15E+02 (+)
	100	4.66E+02	8.52E+04 (+)	2.96E+02 (-)	7.31E+02 (+)	1.01E+05 (+)	1.06E+05 (+)	1.24E+03 (+)	4.23E+02 (-)
f_{35}	50	2.03E+01	2.12E+01 (+)	2.11E+01 (+)	5.20E+02 (+)	2.12E+01 (+)	2.11E+01 (+)	5.21E+02 (+)	5.20E+02 (+)
	100	2.04E+01	2.14E+01 (+)	2.13E+01 (+)	5.20E+02 (+)	2.13E+01 (+)	2.13E+01 (+)	5.21E+02 (+)	5.20E+02 (+)
f_{36}	50	6.63E+01	7.49E+01 (-)	4.62E+01 (-)	6.50E+02 (+)	7.78E+01 (+)	6.35E+01 (≃)	6.36E+02 (+)	6.07E+02 (+)
	100	1.49E+02	1.59E+02 (+)	1.67E+02 (+)	7.19E+02 (+)	1.65E+02 (+)	1.45E+02 (-)	6.98E+02 (+)	6.07E+02 (+)
f_{37}	50	3.18E-01	1.33E+03 (+)	2.12E+01 (+)	7.00E+02 (+)	1.45E+03 (+)	1.56E+03 (+)	7.03E+02 (+)	7.03E+02 (+)
	100	2.66E+00	2.77E+03 (+)	1.24E-02 (-)	7.00E+02 (+)	3.04E+03 (+)	3.11E+03 (+)	7.37E+02 (+)	7.04E+02 (+)
f_{38}	50	2.53E+02	6.03E+02 (+)	6.51E+02 (+)	1.14E+03 (+)	7.04E+02 (+)	4.87E+02 (+)	1.10E+03 (+)	8.19E+02 (+)
	100	6.05E+02	1.23E+03 (+)	1.20E+03 (+)	1.66E+03 (+)	1.48E+03 (+)	1.11E+03 (+)	1.39E+03 (+)	8.24E+02 (+)
f_{39}	50	3.41E+02	6.80E+02 (+)	7.58E+02 (+)	1.43E+03 (+)	7.57E+02 (+)	5.69E+02 (+)	1.33E+03 (+)	9.28E+02 (+)
	100	7.48E+02	1.43E+03 (+)	1.01E+03 (+)	2.16E+03 (+)	1.53E+03 (+)	1.24E+03 (+)	1.67E+03 (+)	9.25E+02 (+)
f_{40}	50	9.33E+03	1.33E+04 (+)	1.45E+04 (+)	6.71E+03 (-)	1.48E+04 (+)	1.28E+04 (+)	8.87E+03 (≃)	1.24E+03 (-)
	100	2.19E+04	2.95E+04 (+)	3.15E+04 (+)	1.43E+04 (-)	3.20E+04 (+)	2.97E+04 (+)	1.99E+04 (-)	1.28E+03 (-)
f_{41}	50	1.02E+04	1.48E+04 (+)	1.47E+04 (+)	8.62E+03 (-)	1.49E+04 (-)	1.35E+04 (-)	1.25E+04 (+)	1.97E+03 (-)
	100	2.37E+04	3.30E+04 (+)	3.18E+04 (+)	1.73E+04 (-)	3.10E+04 (+)	2.86E+04 (+)	2.54E+04 (+)	1.95E+03 (-)
f_{42}	50	1.60E+00	5.48E+00 (+)	5.29E+00 (+)	1.20E+03 (+)	3.86E+00 (+)	3.53E+00 (+)	1.20E+03 (+)	1.20E+03 (+)
	100	2.06E+00	5.67E+00 (+)	5.30E+00 (+)	1.20E+03 (+)	4.75E+00 (+)	4.17E+00 (+)	1.20E+03 (+)	1.20E+03 (+)
f_{43}	50	5.39E-01	7.66E+00 (-)	4.72E-01 (-)	1.30E+03 (+)	8.62E+00 (+)	8.75E+00 (+)	1.30E+03 (+)	1.30E+03 (+)
	100	5.24E-01	8.97E+00 (+)	5.40E-01 (≃)	1.30E+03 (+)	9.49E+00 (+)	9.66E+00 (+)	1.30E+03 (+)	1.30E+03 (+)
f_{44}	50	7.26E+00	3.12E+02 (≃)	3.19E-01 (≃)	1.40E+03 (+)	3.65E+02 (+)	3.98E+02 (+)	1.40E+03 (+)	1.40E+03 (+)
	100	3.52E-01	8.47E+02 (+)	3.51E-01 (≃)	1.40E+03 (+)	8.99E+02 (+)	9.40E+02 (+)	1.40E+03 (+)	1.40E+03 (+)
f_{45}	50	1.67E+02	2.83E+06 (-)	2.64E+01 (-)	1.54E+03 (+)	4.71E+06 (+)	6.29E+06 (+)	1.63E+03 (+)	1.53E+03 (+)
	100	5.11E+02	1.71E+07 (+)	6.16E+01 (-)	1.60E+03 (+)	3.03E+07 (+)	2.75E+07 (+)	3.01E+03 (+)	1.53E+03 (+)
f_{46}	50	2.21E+01	2.33E+01 (+)	2.30E+01 (+)	1.62E+03 (+)	2.30E+01 (+)	2.24E+01 (+)	1.62E+03 (+)	1.60E+03 (+)
	100	4.53E+01	4.75E+01 (+)	4.75E+01 (+)	1.64E+03 (+)	4.75E+01 (+)	4.65E+01 (+)	1.65E+03 (+)	1.60E+03 (+)
f_{47}	50	1.22E+06	9.46E+10 (-)	4.19E+03 (-)	7.68E+04 (-)	1.13E+11 (+)	1.10E+11 (+)	2.47E+06 (+)	
	100	2.09E+08	2.46E+11 (+)	2.01E+08 (-)	1.39E+05 (-)	2.68E+11 (+)	2.61E+11 (+)	1.23E+07 (-)	
f_{48}	50	6.40E+39	1.09E+82 (+)	5.73E+09 (+)	1.35E+04 (-)	1.86E+84 (-)	1.73E+94 (-)	3.93E+03 (+)	
	100	7.38E+115	1.52E+181 (+)	1.01E+51 (-)	2.44E+04 (-)	4.53E+177 (+)	8.78E+195 (+)	1.40E+05 (-)	
f_{49}	50	1.03E+05	2.38E+05 (-)	8.48E+04 (-)	1.96E+03 (+)	1.87E+05 (+)	1.55E+05 (+)	1.97E+03 (+)	
	100	3.05E+05	4.56E+05 (+)	2.18E+05 (-)	2.04E+03 (-)	3.43E+05 (+)	3.29E+05 (+)	2.12E+03 (-)	
f_{50}	50	8.74E+02	1.24E+03 (+)	1.02E+03 (+)	2.62E+03 (+)	1.37E+03 (+)	1.21E+03 (+)	3.00E+04 (+)	
	100	2.43E+03	3.25E+03 (+)	2.36E+03 (≃)	2.00E+04 (+)	3.32E+03 (+)	3.08E+03 (+)	9.10E+04 (+)	
f_{51}	10	-3.31E+02	6.87E+02 (+)	7.27E+02 (+)	1.04E+06 (+)	7.39E+02 (+)	5.75E+02 (+)	1.30E+06 (+)	
	30	-9.38E+02	1.67E+03 (+)	1.21E+03 (+)	2.91E+05 (+)	1.80E+03 (+)	1.53E+03 (+)	5.30E+06 (+)	
f_{52}	10	-1.23E+04	3.55E+04 (+)	3.29E+04 (+)	3.55E+03 (-)	3.69E+04 (+)	3.31E+04 (+)	3.48E+03 (-)	
	30	-3.01E+04	7.49E+04 (+)	3.66E+04 (≃)	5.50E+03 (-)	8.13E+04 (+)	7.12E+04 (+)	4.65E+03 (-)	

level on the basis of the performance results presented in Tables 10 and 11. The results show that MPGO also dominates PSO, GSA, LSA, BOA, SOS, and MSA algorithm. In addition, MPGO achieves slightly better results than the MS. In summary, Tables 12 and 13 indicate that the superior performance of MPGO is statistically significant with PSO, GSA, LSA, MS, BOA, SOS, and MSA for the overall performance evaluation.

H. COMPARISON OF MPGO WITH DIFFERENT VARIANTS

The performance of MPGO in terms of application to data clustering was also evaluated. We employed six UCI

TABLE 12. Two-tailed Wilcoxon rank-sum test of MPGO versus compared algorithms with 10 and 30 dimensions on CEC benchmark.

Compared algorithm	Dims	Significantly better(+)	Significantly worse(-)	No significantly difference (≃)
(MPGO, PSO)	10	22	0	0
	30	22	0	0
(MPGO, GSA)	10	15	4	3
	30	14	6	2
(MPGO, LSA)	10	20	1	1
	30	16	1	3
(MPGO, MS)	10	22	0	0
	30	22	0	0
(MPGO, BOA)	10	20	1	1
	30	20	1	1
(MPGO, SOS)	10	22	0	0
	30	20	0	2
(MPGO, MSA)	10	16	0	0
	30	11	5	0

TABLE 13. Two-tailed Wilcoxon rank-sum test of MPGO versus compared algorithms with 50 and 100 dimensions on CEC benchmark.

Compared algorithm	Dims	Significantly better(+)	Significantly worse(-)	No significantly difference (≃)
(MPGO, PSO)	50	22	0	0
	100	22	0	0
(MPGO, GSA)	50	14	6	2
	100	9	7	5
(MPGO, LSA)	50	14	8	0
	100	13	9	0
(MPGO, MS)	50	20	2	0
	100	22	0	0
(MPGO, BOA)	50	19	2	1
	100	21	1	0
(MPGO, SOS)	50	19	1	2
	100	16	5	1
(MPGO, MSA)	50	12	4	0
	100	10	6	0

TABLE 14. UCI instances.

Dataset	Number of Patterns	Number of Dims	Groups
Iris	150	4	3
Wine	178	13	3
Breast cancer	683	9	2
Car evaluation	1728	6	4
Statlog	6435	36	7
Yeast	1484	8	10

TABLE 15. Parameter settings for six variants of the MPGO.

Variant	Specific FEs	Specific individuals
1	250	2
2	250	5
3	1250	2
4	1250	5
5	1875	2
6	2500	10

benchmarks (http://archive.ics.uci.edu/ml/datasets.html), with pattern numbers ranging from 150 to 1728, as listed in Table 14. Several variants of the MPGO algorithm were compared in the simulations. The number of FEs and individuals exchanged among the composite algorithms of each variant are listed in Table 15. MPGO exchanged two individuals between PSO and GSA every 250 FEs and 10 individuals in a maximum of 25000 FEs in the 20 runs.

In this experiment, the datasets Iris, Wine, and Breast cancer were selected to verify each MPGO variant. In summary, the performances of all the variants were approximately equal for 25000 FEs. The average accuracy rates are summarized in Table 16; the results indicate that MPGO2 exhibited the highest average accuracy on the Wine and Breast cancer benchmarks, whereas MPGO1 exhibited the highest accuracy on the Wine benchmark. Thus, MPGO2 was used for comparison with the other algorithms in the remaining experiments.

TABLE 16. Comparison of different variants of MPGO.

Variants	Iris	Wine	Breast cancer
1	97.7	78.6	97.9
2	98.2	77.6	97.9
3	97.1	77.7	97.8
4	97.3	77.2	97.9
5	97.5	76.7	97.8
6	97.8	76.6	97.9

I. EXPERIMENTAL RESULTS OF DATA CLUSTERING

The performance of MPGO was compared with that of the k-means algorithm, PSO, GSA, black hole (BH) algorithm [43], and WOA [64] on six UCI datasets (Iris, Wine, Breast cancer, Car evaluation, Statlog, and Yeast). The performance measure is defined by Eq. (26):

$$\Delta Accuracy = \frac{Q_{algo} - Q_{mpgo}}{Q_{mpgo}} \times 100\% \tag{26}$$

where, $\Delta Accuracy$ denotes the improvement in the accuracy rate, Q_{mpgo} denotes the accuracy of the MPGO algorithm, and Q_{algo} denotes the accuracy of k-means algorithm [9],

TABLE 17. Comparison of accuracy values obtained with k-means, PSO, GSA, BH, and WOA.

Dataset	k-means [9]	PSO [59]	GSA [105]	BH [43]	WOA [64]
Iris	56.6	15.1	5.1	2.4	0.9
Wine	50.3	10.9	8.3	4.6	4.2
Breast Cancer	44.3	1.9	1.7	0.3	0.3
Car evaluation	62.6	41.5	37.0	11.4	4.6
Statlog	80.8	41.5	27.4	14.1	23.8
Yeast	78.4	53.0	38.9	26.7	22.2
Average	62.2	27.3	19.7	9.9	9.3

TABLE 18. Comparison of PSNR values obtained with MPGO, k-means, PSO, GSA, BH and WOA.

Dataset	k-means [9]	PSO [59]	GSA [105]	BH [43]	WOA [64]	MPGO
Lena	31.40	31.42	31.42	31.65	31.70	**31.70**
Baboon	30.45	30.50	30.52	31.59	31.64	**31.78**
Air plane	33.08	33.55	33.62	33.76	33.81	**33.99**
Peppers	30.43	30.55	30.53	30.71	30.76	**30.77**
Goldhill	30.41	30.72	30.70	30.95	30.77	**30.99**
Sailboat	32.00	32.30	32.29	32.29	32.34	**32.34**
Average	31.29	31.50	31.51	31.83	31.84	**31.93**

PSO algorithm [59], GSA [105], BH algorithm [43], and WOA [64].

The experimental results are summarized in Table 17. The $\Delta Accuracy$ values of the MPGO algorithm are better than those of the k-means (from 44.3% to 80.8%), PSO (from 1.9% to 53.0%), GSA (from 1.7% to 37.0%), BH (from 0.3% to 26.7%), and WOA (from 0.3% to 23.8%). Furthermore, MPGO is more effective than all the algorithms on the Statlog and Yeast benchmarks (from 14.1% to 80.8%, respectively).

J. APPLICATION TO IMAGE SEGMENTATION

In the image segmentation process, an image is divided into a set of regions. Each pixel is classified and assigned to a region on the basis of similarity. Thus, meta-heuristic algorithms can be easily applied to the image segmentation problem via partitional models [95], [106]. The performance of the proposed MPGO algorithm was verified by segmenting six well-known 8-bit gray images having a resolution 256×256, as shown in Figure 7. The metric used to compare the results was the PSNR value, which is defined by Eq. 27:

$$PSNR = 10 * \log_{10}\left(\frac{255^2}{\text{MSE}}\right) \tag{27}$$

The experimental results are listed in Table 18. According to the results, the average PSNR value of MPGO is better than that of k-means, PSO, GSA, BH, and WOA by 0.64, 0.43, 0.42, 0.10, and 0.09, respectively.

VI. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This paper proposed the MPGO algorithm for solving clustering problems. MPGO combines PSO and GSA to produce a hybrid optimization algorithm with an efficient search strategy (based on GSA) and fast convergence (based on PSO). The performance of MPGO was evaluated on 52 benchmark test functions, six UCI machine learning benchmarks, and image segmentation of six well-known images. Comparisons conducted with five existing algorithms verified the superior performance of the proposed algorithm in terms of fitness value, accuracy rate, and PSNR. In future research, we will investigate the following four aspects. (1) Use of MPGO for image enhancement in computed tomography (CT); (2) pattern reduction or dimension reduction to improve computing performance; (3) updating of MPGO individuals via the Lévy flight strategy to enhance diversity; and (3) implementation of MPGO on a Raspberry Pi to enable automatic object tracking and text recognition

REFERENCES

[1] R. Xu and D. Wunsch, II, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[2] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[3] A. K. Jain, R. P. W. Duin, and J. C. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, Jan. 2000.

[4] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2012.

[5] J. Ramos, J. A. Castellanos-Garzón, A. González-Briones, J. F. de Paz, and J. M. Corchado, "An agent-based clustering approach for gene selection in gene expression microarray," *Interdiscipl. Sciences: Comput. Life Sci.*, vol. 9, no. 1, pp. 1–13, 2017.

[6] Y. Yang, Z. Wu, and W. Kong, "Improving clustering of microrna microarray data by incorporating functional similarity," *Current Bioinf.*, vol. 13, no. 1, pp. 34–41, 2018.

[7] S. Rosati, V. Agostini, M. Knaflitz, and G. Balestra, "Muscle activation patterns during gait: A hierarchical clustering analysis," *Biomed. Signal Process. Control*, vol. 31, pp. 463–469, Jan. 2017.

[8] G. Bruno, E. M. Carlini, and C. Pisani, "Signal processing techniques for sensing based generator coherency analysis," *Int. J. Elect. Power Energy Syst.*, vol. 104, pp. 215–221, Jun. 2019.

[9] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.

[10] J. C. Dunn, "Well-separated clusters and optimal fuzzy partitions," *J. Cybern.*, vol. 4, no. 1, pp. 95–104, 2008.

[11] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Appl. Statist.*, vol. 28, no. 1, pp. 100–108, 1979.

[12] J. C. Bezdek, W. Full, and R. Ehrlich, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, nos. 2–3, pp. 191–203, 1984.

[13] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.
[14] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics*, 2nd ed. New York, NY, USA: Springer, 2010.
[15] R. Martí, P. M. Pardalos, and M. G. Resende, *Handbook of Heuristics*. New York, NY, USA: Springer, 2018.
[16] Y. Xue, J. Jiang, B. Zhao, and T. Ma, "A self-adaptive artificial bee colony algorithm based on global best for global optimization," *Soft Comput.*, vol. 22, no. 9, pp. 2935–2952, 2017.
[17] T. Jayabarathi, T. Raghunathan, and A. H. Gandomi, "The bat algorithm, variants and some practical engineering applications: A review," in *Nature-Inspired Algorithms and Applied Optimization*. Cham, Switzerland: Springer, 2018, pp. 313–330.
[18] S. Arora and S. Singh, "Butterfly optimization algorithm: A novel approach for global optimization," *Soft Comput.*, vol. 23, pp. 715–734, Feb. 2018.
[19] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Comput. Struct.*, vol. 169, pp. 1–12, Jun. 2016.
[20] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems," *Eng. Comput.*, vol. 29, no. 1, pp. 17–35, 2011.
[21] N. H. Awad, M. Z. Ali, R. Mallipeddi, and P. N. Suganthan, "An improved differential evolution algorithm using efficient adapted surrogate model for numerical optimization," *Inf. Sci.*, vols. 451–452, pp. 326–347, 2018.
[22] S. Deb, S. Fong, and Z. Tian, "Elephant search algorithm for optimization problems," in *Proc. IEEE 10th Int. Conf. Digit. Inf. Manage. (ICDIM)*, Oct. 2015, pp. 249–255.
[23] H. Shayanfar and F. S. Gharehchopogh, "Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems," *Appl. Soft Comput.*, vol. 71, pp. 728–746, Oct. 2018.
[24] I. Fister, I. Fister, Jr., X.-S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm Evol. Comput.*, vol. 13, pp. 34–46, Dec. 2013.
[25] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.
[26] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[27] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *J. Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009.
[28] M. Yazdani and F. Jolai, "Lion optimization algorithm (LOA): A nature-inspired metaheuristic algorithm," *J. Comput. Des. Eng.*, vol. 3, no. 1, pp. 24–36, 2016.
[29] H. Shareef, A. A. Ibrahim, and A. H. Mutlag, "Lightning search algorithm," *Appl. Soft Comput. J.*, vol. 36, pp. 315–333, Nov. 2015.
[30] R.-Q. Zhao and W.-S. Tang, "Monkey algorithm for global numerical optimization," *J. Uncertain Syst.*, vol. 2, no. 3, pp. 165–176, 2008.
[31] G.-G. Wang, "Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Comput.*, vol. 10, no. 2, pp. 151–164, 2018.
[32] A.-A. A. Mohamed, Y. S. Mohamed, A. A. M. El-Gaafary, and A. M. Hemeida, "Optimal power flow using moth swarm algorithm," *Electr. Power Syst. Res.*, vol. 142, pp. 190–206, Jan. 2017.
[33] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci.*, 1995, pp. 39–43.
[34] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.
[35] M.-Y. Cheng and D. Prayogo, "Symbiotic Organisms Search: A new metaheuristic optimization algorithm," *Comput. Struct.*, vol. 139, pp. 98–112, Jul. 2014.
[36] O. Abedinia, N. Amjady, and A. Ghasemi, "A new metaheuristic algorithm based on shark smell optimization," *Complexity*, vol. 21, no. 5, pp. 97–116, 2016.
[37] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[38] H. D. Menéndez, F. E. B. Otero, and D. Camacho, "Medoid-based clustering using ant colony optimization," *Swarm Intell.*, vol. 10, no. 2, pp. 123–145, 2016.
[39] D. Karaboga and C. Ozturk, "A novel clustering approach: Artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 11, pp. 652–657, Jan. 2011.
[40] G. Sahoo and Y. Kumar, "A two-step artificial bee colony algorithm for clustering," *Neural Comput. Appl.*, vol. 28, no. 3, pp. 537–551, 2017.
[41] P. Das, D. K. Das, and S. Dey, "A modified Bee Colony Optimization (MBCO) and its hybridization with k-means for an application to data clustering," *Appl. Soft Comput.*, vol. 70, pp. 590–603, Sep. 2018.
[42] T. Ashish, S. Kapil, and B. Manju, "Parallel bat algorithm-based clustering using MapReduce," in *Networking Communication and Data Knowledge Engineering*. Singapore: Springer, 2018, pp. 73–82.
[43] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Inf. Sci.*, vol. 222, pp. 175–184, Feb. 2012.
[44] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 1, pp. 218–237, Jan. 2008.
[45] J. Tvrdík and I. Křivý, "Hybrid differential evolution algorithm for optimal clustering," *Appl. Soft Comput.*, vol. 35, pp. 502–512, Oct. 2015.
[46] Z. Tian, S. Fong, R. Wong, and R. Millham, "Elephant search algorithm on data clustering," in *Proc. IEEE 12th Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery (ICNC-FSKD)*, Aug. 2016, pp. 787–793.
[47] J. Senthilnath, S. N. Omkar, and V. Mani, "Clustering using firefly algorithm: Performance study," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 164–171, 2011.
[48] M. Demir and A. Karci, "Data clustering on breast cancer data using firefly algorithm with golden ratio method," *Adv. Elect. Comput. Eng.*, vol. 15, no. 2, pp. 75–84, 2015.
[49] M. H. Hajeer and D. Dasgupta, "Distributed genetic algorithm to big data clustering," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2016, pp. 1–9.
[50] S. H. Razavi, E. O. M. Ebadati, S. Asadi, and H. Kaur, "An efficient grouping genetic algorithm for data clustering and big data analysis," in *Computational Intelligence for Big Data Analysis*. Cham, Switzerland: Springer, 2015, pp. 119–142.
[51] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 29, no. 3, pp. 433–439, Jun. 1999.
[52] C. Pizzuti and N. Procopio, "A K-means based genetic algorithm for data clustering," in *Proc. Int. Joint Conf. SOCO-CISIS-ICEUTE*, 2016, pp. 211–222.
[53] K.-W. Huang, J.-L. Chen, C.-S. Yang, and C.-W. Tsai, "A memetic gravitation search algorithm for solving clustering problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 751–757.
[54] X. Han, L. Quan, X. Xiong, M. Almeter, J. Xiang, and Y. Lan, "A novel data clustering algorithm based on modified gravitational search algorithm," *Eng. Appl. Artif. Intell.*, vol. 61, pp. 1–7, May 2017.
[55] A. K. Tripathi, K. Sharma, and M. Bala, "A novel clustering method using enhanced grey wolf optimizer and MapReduce," *Big Data Res.*, vol. 14, pp. 93–100, Dec. 2018.
[56] S. Chander, P. Vijaya, and P. Dhyani, "Multi kernel and dynamic fractional lion optimization algorithm for data clustering," *Alexandria Eng. J.*, vol. 57, no. 1, pp. 267–276, 2018.
[57] X. Chen, Y. Zhou, and Q. Luo, "A hybrid monkey search algorithm for clustering analysis," *Sci. World J.*, vol. 2014, Mar. 2014, Art. no. 938239.
[58] X. Yang, Q. Luo, J. Zhang, X. Wu, and Y. Zhou, "Moth swarm algorithm for clustering analysis," in *Proc. Int. Conf. Intell. Comput.*, 2017, pp. 503–514.
[59] D. W. van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 1, Dec. 2003, pp. 215–220.
[60] A. A. A. Esmin, R. A. Coelho, and S. Matwin, "A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data," *Artif. Intell. Rev.*, vol. 44, no. 1, pp. 23–45, 2015.
[61] S. Z. Selim and K. Alsultan, "A simulated annealing algorithm for the clustering problem," *Pattern Recognit.*, vol. 24, no. 10, pp. 1003–1008, 1991.
[62] S. S. Kim, J. Y. Baek, and B. S. Kang, "Hybrid simulated annealing for data clustering," *J. Soc. Korea Ind. Syst. Eng.*, vol. 40, no. 2, pp. 92–98, 2017.
[63] Y. Zhou, H. Wu, Q. Luo, and M. Abdel-Baset, "Automatic data clustering using nature-inspired symbiotic organism search algorithm," *Knowl.-Based Syst.*, vol. 163, pp. 546–557, Jan. 2019.
[64] J. Nasiri and F. M. Khiyabani, "A whale optimization algorithm (WOA) approach for clustering," *Cogent Math. Statist.*, vol. 5, no. 1, 2018, Art. no. 1483565.

[65] A. Kamburov, M. S. Lawrence, P. Polak, I. Leshchiner, K. Lage, T. R. Golub, E. S. Lander, and G. Getz, "Comprehensive assessment of cancer missense mutation clustering in protein structures," *Proc. Nat. Acad. Sci. USA*, vol. 112, no. 40, pp. E5486–E5495, 2015.

[66] H. Liu, R. Zhao, H. Fang, F. Cheng, Y. Fu, and Y.-Y. Liu, "Entropy-based consensus clustering for patient stratification," *Bioinformatics*, vol. 33, no. 17, pp. 2691–2698, 2017.

[67] C. Wiwie, J. Baumbach, and R. Röttger, "Comparing the performance of biomedical clustering methods," *Nature Methods*, vol. 12, no. 11, pp. 1033–1038, 2015.

[68] J. Z. Lai, Y. C. Liaw, and J. Liu, "A fast VQ codebook generation algorithm using codeword displacement," *Pattern Recognit.*, vol. 41, no. 1, pp. 315–319, 2008.

[69] K. Chiranjeevi, U. Jena, and P. M. K. Prasad, "Hybrid cuckoo search based evolutionary vector quantization for image compression," in *Proc. Artif. Intell. Comput. Vis.* Cham, Switzerland: Springer, 2017, pp. 89–114.

[70] A. De and C. Guo, "An adaptive vector quantization approach for image segmentation based on SOM network," *Neurocomputing*, vol. 149, pp. 48–58, Feb. 2015.

[71] C. Karri and U. Jena, "Fast vector quantization using a Bat algorithm for image compression," *Eng. Sci. Technol., Int. J.*, vol. 19, pp. 769–781, Jun. 2016.

[72] S. Hussain, R. Atallah, A. Kamsin, and J. Hazarika, "Classification, clustering and association rule mining in educational datasets using data mining tools: A case study," in *Proc. Comput. Sci. On-Line Conf.*, 2018, pp. 196–211.

[73] M. K. Najafabadi, M. N. Mahrin, S. Chuprat, and H. M. Sarkan, "Improving the accuracy of collaborative filtering recommendations using clustering and association rules mining on implicit data," *Comput. Hum. Behav.*, vol. 67, pp. 113–128, Feb. 2017.

[74] N. Sheydaei, M. Saraee, and A. Shahgholian, "A novel feature selection method for text classification using association rules and clustering," *J. Inf. Sci.*, vol. 41, no. 1, pp. 3–15, 2015.

[75] A. M. Al-Abadi, "A novel geographical information system-based ant miner algorithm model for delineating groundwater flowing artesian well boundary: A case study from iraqi southern and western deserts," *Environ. Earth Sci.*, vol. 76, no. 15, p. 534, 2017.

[76] L. Zhao, L. Chen, R. Ranjan, K.-K. R. Choo, and J. He, "Geographical information system parallelization for spatial big data processing: A review," *Cluster Comput.*, vol. 19, no. 1, pp. 139–152, 2016.

[77] K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, and B. Fielding, "A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1496–1509, Jun. 2017.

[78] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. New York, NY, USA: Academic, 2008.

[79] S. Zhang, H. Wang, and W. Huang, "Two-stage plant species recognition by local mean clustering and weighted sparse representation classification," *Cluster Comput.*, vol. 20, no. 2, pp. 1517–1525, 2017.

[80] H. Medeiros, J. Park, and A. Kak, "A light-weight event-driven protocol for sensor clustering in wireless camera networks," in *Proc. 1st ACM/IEEE Int. Conf. Distrib. Smart Cameras*, Sep. 2007, pp. 203–210.

[81] S. A. Sert, H. Bagci, and A. Yazici, "MOFCA: Multi-objective fuzzy clustering algorithm for wireless sensor networks," *Appl. Soft Comput.*, vol. 30, pp. 151–165, May 2015.

[82] Y. Zhou, N. Wang, and W. Xiang, "Clustering hierarchy protocol in wireless sensor networks using an improved PSO algorithm," *IEEE Access*, vol. 5, pp. 2241–2253, 2017.

[83] H. Çataloluk and F. V. Çelebi, "A novel hybrid model for two-phase image segmentation: GSA based Chan–Vese algorithm," *Eng. Appl. Artif. Intell.*, vol. 73, pp. 22–30, Aug. 2018.

[84] L. Chen, C. L. P. Chen, and M. Lu, "A multiple-kernel fuzzy C-means algorithm for image segmentation," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 41, no. 5, pp. 1263–1274, Oct. 2011.

[85] S. K. Choy, S. Y. Lam, K. W. Yu, W. Y. Lee, and K. T. Leung, "Fuzzy model-based clustering and its application in image segmentation," *Pattern Recognit.*, vol. 68, pp. 141–157, Aug. 2017.

[86] Y. Guo, R. Xia, and A. Şengür, and K. Polat, "A novel image segmentation approach based on neutrosophic c-means clustering and indeterminacy filtering," *Neural Comput. Appl.*, vol. 28, no. 10, pp. 3009–3019, 2017.

[87] T. X. Pham, P. Siarry, and H. Oulhadj, "Integrating fuzzy entropy clustering with an improved PSO for MRI brain image segmentation," *Appl. Soft Comput.*, vol. 65, pp. 230–242, Apr. 2018.

[88] P. K. Das, H. S. Behera, and B. K. Panigrahi, "A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning," *Swarm Evol. Comput.*, vol. 28, pp. 14–28, Jun. 2016.

[89] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley, 1989.

[90] Y. Liu, Z. Qin, Z. Shi, and J. Lu, "Center particle swarm optimization," *Neurocomputing*, vol. 70, nos. 4–6, pp. 672–679, 2007.

[91] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[92] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4. Nov./Dec. 1995, pp. 1942–1948.

[93] S. Gao, C. Vairappan, Y. Wang, Q. Cao, and Z. Tang, "Gravitational search algorithm combined with chaos for unconstrained numerical optimization," *Appl. Math. Comput.*, vol. 231, pp. 48–62, Mar. 2014.

[94] S. J. Nanda and G. Panda, "A survey on nature inspired metaheuristic algorithms for partitional clustering," *Swarm Evol. Comput.*, vol. 16, pp. 1–18, Jun. 2014.

[95] M. Omran, A. P. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 19, no. 3, pp. 297–321, 2005.

[96] M. T. Wong, X. He, and W.-C. Yeh, "Image clustering using particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2011, pp. 262–268.

[97] M. Khajehzadeh, M. R. Taha, A. El-Shafie, and M. Eslami, "A modified gravitational search algorithm for slope stability analysis," *Eng. Appl. Artif. Intell.*, vol. 25, no. 8, pp. 1589–1597, 2012.

[98] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[99] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[100] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization," Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou, China, Nanyang Technol. Univ., Singapore, Tech. Rep. 201311, 2013.

[101] A. P. Engelbrecht, "Fitness function evaluations: A fair stopping condition?" in *Proc. IEEE Symp. Swarm Intell. (SIS)*, Dec. 2014, pp. 1–8.

[102] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Symp. Swarm Intell.*, Apr. 2007, pp. 120–127.

[103] 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, 2011.

[104] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization," IEEE Congr. Evol. Comput., Donostia-San Sebastian, Spain, Tech. Rep., 2016.

[105] A. Hatamlou, S. Abdullah, and Z. Othman, "Gravitational search algorithm with heuristic search for clustering problems," in *Proc. 3rd Conf. Data Mining Optim. (DMO)*, Jun. 2011, pp. 190–193.

[106] A. Sepas-Moghaddam, D. Yazdani, and J. Shahabi, "A novel hybrid image segmentation method," *Prog. Artif. Intell.*, vol. 3, no. 1, pp. 39–49, 2014.

KO-WEI HUANG received the Ph.D. degree from the Department of Electrical Engineering, Institute of Computer and Communication Engineering, National Cheng Kung University, Tainan, Taiwan, in 2015. He is currently an Assistant Professor with the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan. His current research interests mainly include data mining, evolutionary computing, and image processing.

ZE-XUE WU received the B.Sc. degree from the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan. His research interests include machine learning, data mining, and artificial intelligence.

HSING-WEI PENG received the B.Sc. degree from the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan. His research interests include image processing and Raspberry Pi.

MING-CHIA TSAI received the B.Sc. degree from the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan.

YU-CHIEH HUNG received the B.Sc. degree from the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan.

YU-CHIN LU received the B.Sc. degrees from the Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Taiwan.