A dynamic neighborhood learning-based gravitational search algorithm

Aizhu Zhang, Genyun Sun, *Member*, *IEEE*, Jinchang Ren, Xiaodong Li, *Senior Member*, *IEEE*, Zhenjie Wang and Xiuping Jia, *Senior Member*, *IEEE*

***Abstract*—Balancing exploration and exploitation according to evolutionary states is crucial to meta-heuristic search (M-HS) algorithms. Owing to its simplicity in theory and effectiveness in global optimization, gravitational search algorithm (GSA) has attracted increasing attention in recent years. However, the trade-off between exploration and exploitation in GSA is achieved mainly by adjusting the size of an archive, named K_{best}, which stores those superior agents after fitness sorting in each iteration. Since the global property of K_{best} remains unchanged in the whole evolutionary process, GSA emphasizes exploitation over exploration and suffers from rapid loss of diversity and premature convergence. To address these problems, in this paper, we propose a dynamic neighborhood learning (DNL) strategy to replace the K_{best} model and thereby present a DNL-based GSA, namely DNLGSA. The method incorporates the local and global neighborhood topologies for enhancing the exploration and obtaining adaptive balance between exploration and exploitation. The local neighborhoods are dynamically formed based on evolutionary states. To delineate the evolutionary states, two convergence criteria named limit value and population diversity, are introduced. Moreover, a mutation operator is designed for escaping from the local optima on the basis of evolutionary states. The proposed algorithm was evaluated on 27 benchmark problems with different characteristic and various difficulties. The results reveal that DNLGSA exhibits competitive performances when compared with a variety of state-of-the-art M-HS algorithms. Moreover, the incorporation of local neighborhood topology reduces the numbers of calculations of gravitational force and thus alleviates the high computational cost of GSA.**

***Index Terms*—Gravitational search algorithm (GSA), dynamic neighborhood, convergence criterion, topology, evolutionary states.**

I. Introduction

Finding the global optimum is a common and challenging task in the development of meta-heuristic search (M-HS) algorithms [1], for which a M-HS algorithm should provide an effective way to balance exploration and exploitation [2]. Exploration is the ability to explore broad regions of the search space, whereas exploitation is the capability of concentrating the search around a promising area to refine a candidate solution. The trade-off between exploration and exploitation greatly affects the convergence accuracy and speed of M-HS algorithms [3]. To speed up convergence and alleviate premature convergence, particles should perform extensive exploration in the early stages and execute refined exploitation in the latter stages. However, how to achieve a fine balance between these two remains an unsolved challenge [4].

Over the past decades, a number of methods have been developed for balancing exploration and exploitation. Typical examples include control of the optimization parameters [5, 6] and neighborhood topology based approach [7–9]. Among these techniques, the neighborhood topology is one of the popular methods due to its effects on the dissemination of search information during the evolutionary process [10]. To this end, various types of neighborhood topologies have been proposed [11–14]. According to the scope of interaction among particles, these topologies can be roughly grouped into two categories, i.e. global neighborhood topology and local neighborhood topology. In global neighborhood topology, all particles are connected to each other and attracted to the global best particle of the whole population, i.e. ***gbest***. Its main merits include rapid convergence and the ability of exploitation around ***gbest*** [7]. However, the population is more likely to be confined at a local optimum, as the ***gbest*** found early in the search can be a poor leader [7]. On the contrary, in local neighborhood topology, each particle connects only to several other particles in its neighborhood and is attracted by the best position of the neighborhood, lbest [11]. This kind of topology enables particles to search diverse regions of the problem space and puts more emphasis on exploration [15]. Previous research has indicated that the local neighborhood topology is more suitable for complex problems [16]. Despite the exploration ability, the local neighborhood topology, however, slows the convergence speed down because of the delay of information spread among particles. In other words, both local and global neighborhood topologies have their cons and pros. In considering their supplementary role to each other, it has attracted increasing attention to the combination of both in an improved M-HS solution [17].

The gravitational search algorithm (GSA) is one of the recent M-HS algorithms inspired by the law of gravity [18]. In GSA, a particle is guided by the sum of gravitational force exerted on it by all the particles stored in K_{best} [18], where

Manuscript received October 18, 2016; revised December 14, 2016. This work was supported by Chinese Natural Science Foundation Projects (41471353, 41271349), Fundamental Research Funds for the Central Universities (15CX06001A). (*Corresponding author: Genyun Sun*)

A. Z. Zhang, G. Y. Sun, and Z. J. Wang are with the School of Geosciences, China University of Petroleum (East China), Qingdao Shandong, 266580, China and the Laboratory for Marine Mineral Resources Qingdao National Laboratory for Marine Science and Technology, Qingdao, 266071, China (genyunsun@163.com).

J. C. Ren is with Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, G1 1XQ, United Kingdom and the Guangzhou Key Laboratory of Digital Content Processing and Security Technologies, Guangzhou, 510665, China

X. D. Li is with the School of Computer Science and Information Technology, RMIT University, Melbourne, VIC3001, Australia

X. P. Jia is with the School of Engineering and Information Technology, The University of New South Wales at Canberra, Canberra, ACT2600, Australia

the movement of a particle is propelled in many different directions. That is, each particle can learn more from all the particles stored in K_{best} instead of only one best particle as in the canonical particle swarm optimization (PSO) model. This provides GSA a unique property, i.e. diverse search directions. In addition, the size of K_{best} is a function of time, with the initial value $K0 = N$ at the beginning and decreases with time. Therefore, by the lapse of time, the exploration fades out and the exploitation turns to fade in. By adjusting the size of K_{best}, the balance between exploration and exploitation can be approved for GSA [18].

However, in spite of the limited balance effects of the K_{best} model, GSA does have some weaknesses. On one hand, due to the repetitive calculation of gravitational force for all particles, GSA apparently suffers from high computational complexity, especially in the early stages where the size of K_{best} is large [19]. On the other hand, at the later stages, each particle can only learn from very few elite particles, which inevitably causes quick loss of search diversity and fast convergence. In this case, the population might be trapped in a local optimum. The major problem associated with GSA is that the K_{best} model is actually equivalent to a global neighborhood [20], in which all particles learn from the same elites all the time. This kind of learning strategy usually causes rapid information exchange between particles and finally results in premature convergence [3].

To address the aforementioned shortcomings of GSA, many GSA variants have been developed [3, 8, 21–25] in recent years. A few new operators have been introduced, including a disruption operator to explore and exploit the search space in [3], opposition-based learning to improve exploitation ability of GSA [21], and the application of the Black Hole theory to prevent premature convergence and improve the exploration and exploitation abilities of GSA [26]. Combining other state-of-the-art M-HS algorithms with GSA has also developed to enhance GSA. Combining other state-of-the-art M-HS algorithms with GSA is another way to enhance GSA. For example, Li et al. [8] integrated Differential Evolution (DE) with GSA to overcome the premature convergence encountered in unconstrained optimization. Mirjalili et al. introduced the social thinking of PSO into GSA to accelerate convergence in the last iterations and improve the search ability [27–29]. In these algorithms, the global memories are utilized to direct the search path of particles. Especially in [29], researchers have attempted to adaptively balance exploration and exploitation to further improve the performance of GSA. Zhang et al. [24] presented a hybrid genetic algorithm (GA) and GSA (GAGSA) to overcome the problem of premature convergence. In improved GSA [23], both the chaotic perturbation operator and the memory of the position of each particle were utilized. The chaotic operator enhances the global convergence to escape from the local optima, and the memory strategy provides a faster convergence.

While these efforts ameliorate the performance of GSA, few studies have addressed the problems resulted from the K_{best} model where the global property of K_{best} remains unchanged in the evolutionary process. In [27–29], although ***gbest*** is utilized to speed the convergence in the later stages, simultaneously utilization of the ***gbest*** and K_{best} model further strength the global property of GSA. These GSA variants still emphasize exploitation over exploration as GSA does, thus resulting in a rapid loss of diversity and premature convergence. Specifically, the K_{best} model gives rise to the following problems: (1) High computational complexity, especially in the early stages where the size of K_{best} is large; (2) Exploitation bounded only by a few elite particles at the later stages with slow convergence and lack of recovery; and (3) Rapid loss of population diversity and potential premature convergence.

Since incorporating the local and global neighborhood topologies is an effective way for balancing exploitation-exploration [17], this paper presents a dynamic neighborhood learning-based GSA (DNLGSA) to improve the performance of GSA.The K_{best} model is replaced in the proposed method. Specifically, the novelties of DNLGSA are summarized below:

(1) A new learning strategy is presented which combines the local neighborhood with global topology. Through the new learning strategy, each particle can learn search information from i) all the particles in its neighborhood and ii) the historically best experience of the whole population (***gbest***). The local neighborhood is helpful to decrease the computational complexity and keep search diversity while the global model is beneficial to accelerate the convergence speed.

(2) A new mechanism for constructing local neighborhoods dynamically is proposed based on evolutionary state. Two convergence criteria, limit value and population diversity are designed to depict the evolutionary states. The former is used to judge if the ***gbest*** is trapped while the latter is applied to determine whether the local neighborhoods should be dynamically reformulated and to control the scale of mutation. This new operation is critical for preserving population diversity and escaping from local optimum.

The remainder of this paper is organized as follows. Section II briefly describes the frameworks of GSA. In Section III, a detailed introduction of the proposed DNLGSA is given. The experimental setting and simulation results are presented in Section IV. Finally, Section V concludes the proposed algorithm. To make the paper focus on the methodology, more details on the algorithm implementation and parameter selection are provided in the supplementary document.

II. Basic gravitational search algorithm (GSA)

The GSA is a stochastic optimization algorithm inspired by the Newton's law of gravity and mass interactions [18]. This algorithm provides an iterative method that simulates the mass interactions in a D-dimensional space following the Newtonian gravity and the laws of motion. In a basic GSA, a particle $\boldsymbol{x}_i = [x_i^1, x_i^2, ..., x_i^D]$ moves through the search space with the velocity $\boldsymbol{v}_i = [v_i^1, v_i^2, ..., v_i^D]$ which is determined by the gravitational forces exerted by its neighbors. The force between any two particles is directly proportional to their masses and inversely proportional to their distance. So each particle moves towards those particles that have heavier masses [18, 30]. The mass of particle i in generation t, denoted by $Mass_i(t)$, is calculated as follows:

$$nmfit_i(t) = \frac{fit_i^t - worst^t}{best^t - worst^t} \tag{1}$$

where

$$Mass_i(t) = \frac{nmfit_i(t)}{\sum_{j=1}^{N} nmfit_j(t)} \tag{2}$$

where fit_i^t represents the fitness value of particle i at time t. For a minimization problem, the $best(t)$ and $worst(t)$ are defined as follows:

$$best(t) = \min_{j \in [j=1,2,...,N]} fit_j(t) \tag{3}$$

$$worst(t) = \max_{j \in [j=1,2,...,N]} fit_j(t) \tag{4}$$

In an optimization problem, the force acted on the particle i from the particle j at a specific time t can be calculated as follows:

$$F_{ij}^d(t) = G(t) \frac{Mass_i(t) \times Mass_j(t)}{R_{ij}(t) + \varepsilon} (x_j^d(t) - x_i^d(t)) \tag{5}$$

where $G(t)$ is the gravitational constant in generation t; $Mass_i(t)$ and $Mass_j(t)$ are the gravitational mass of the particles i and j, respectively; $R_{ij}(t)$ is the distance between particles i and j; ε is a small positive constant; $x_i^d(t)$ and $x_j^d(t)$ represent the position of the particle i and j in the d-th dimension, respectively.

To give a stochastic characteristic to the GSA, the total force acted on the particle i in the d-th dimension is set to be a randomly weighted sum of d-th components of the forces exerted from its neighbors stored in the elite archive, K_{best}, as shown in Eq. (6) as follows:

$$F_i^d(t) = \sum_{\substack{j \in K_{\text{best}}, \\ j \neq i}} rand \cdot G(t) \frac{Mass_i(t) \times Mass_j(t)}{R_{ij}(t) + \varepsilon} (x_j^d(t) - x_i^d(t))$$
$$where \;\; G(t) = G_0 \times exp(-\beta \times \frac{t}{T_{\max}}) \tag{6}$$

where $rand$ is a uniform random variable in the interval [0, 1], G_0 is the initial value of gravitational constant, β is the coefficient of decrease, and $T_{\max}$ is the maximum number of iterations.

Afterwards, the acceleration $a_i^d(t)$ of the particle i in the d-th dimension can be calculated as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{Mass_i(t)} \tag{7}$$

Hence, the particle i adjust its velocity and position according to Eq. (8) and Eq. (9) as follows:

$$v_i^d(t+1) = rand \cdot v_i^d(t) + a_i^d(t) \tag{8}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{9}$$

III. Dynamic neighborhood learning based-GSA (DNLGSA)

A. Dynamic neighborhood learning strategy

Let $\boldsymbol{X}$=[$\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N$] denote a GSA population, where $\boldsymbol{x}_i = [x_i^1, x_i^2, ..., x_i^D]$, i=1,2,...,N represents a particle in a D-dimensional search space. In the dynamic neighborhood learning strategy, we first randomly divide the whole population into M non-overlapping local neighborhoods, ***DN***={***DN***$_1$, ***DN***$_2$, ..., ***DN***$_M$} of an equal number of particles. Now, for a particle $\boldsymbol{x}_i$ which belongs to the j-th local neighborhood ***DN***$_j$, its neighbors consist of all the other particles in ***DN***$_j$, as illustrated in Fig. 1. Note that the M neighborhoods are dynamically generated based on two convergence criteria which

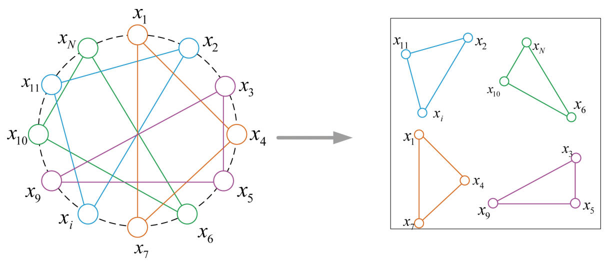


Fig. 1: Non-overlapping local neighborhoods in DNLGSA where N=12 and M=4.

are used to delineate the evolutionary stages (as described in Section III-B).

In a local neighborhood, the particles connect with each other through the gravitational forces. Consequently, for the particle i in the j-th local neighborhood at time t (denoted as ***DN***$_j(t)$), its velocity $Lv_i^d(t+1)$ is given as follow:

$$Lv_i^d(t+1) = \sum_{\substack{\boldsymbol{x}_p \in \boldsymbol{DN}_j(t), \\ \boldsymbol{x}_p \neq \boldsymbol{x}_i}} rand \cdot G(t) \frac{Mass_p(t)}{R_{ip}(t) + \varepsilon} (x_p^d(t) - x_i^d(t)) \tag{10}$$

where $rand$ is is a uniform random variable in the interval [0, 1].

Additionally, the particle i also learns from the best experience of the whole population has been found so far, denoted by ***gbest***= $[g^1, g^2, ..., g^D]$. Apparently, the ***gbest*** model is a global neighborhood topology [11]. The attraction of the ***gbest*** exerts on the particle i is defined as follow:

$$Gv_i^d(t+1) = rand \cdot (g^d(t) - x_i^d(t)) \tag{11}$$

where $Gv_i^d(t+1)$ is the new global velocity produced by the ***gbest***.

Now we combine the velocity created by the local and global neighborhood topologies using two acceleration coefficients, c_1 and c_2, to update the velocity of the particle i as follow:

$$v_i^d(t+1) = rand \cdot v_i^d(t) + c_1 \cdot Lv_i^d(t+1) + c_2 \cdot Gv_i^d(t+1) \tag{12}$$

where c_1 and c_2 are arbitrary nonnegative numbers. Clearly, if c_1 is bigger than c_2, the search process tends to perform exploration, while if c_2 is bigger than c_1, the search process tends to execute exploitation. Since there is no clear border between the exploration and exploitation phases, the adaptive adjustment method provides a practical option to offer a gradual transition between these two phases. Inspired by the adjustment method proposed in GGSA [29], we set $c_1 = 0.5 - 0.5t^{1/6}/T_{max}^{1/6}$ and $c_2 = 1.5t^{1/6}/T_{max}^{1/6}$.

Thereby the particle i updates its position according to Eq. (13) as follow:

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{13}$$

Remark 1. Different from conventional local neighborhood-based algorithms where a particle learns only from the best particle in the neighborhood, the new local neighborhood learning strategy inherits the merit of GSA that enables each particle to learn from all of its neighbors. This clearly preserves the diverse search property of GSA.

Remark 2. The K_{best} model in GSA is a kind of global neighborhood topology while the ***DN***$_j$ in DNLGSA is a dynamic local neighborhood topology which is compatible with the evolutionary states. This makes the algorithm can explore the feasible search space more thoroughly.

Remark 3. The guidance of ***gbest*** exhibits as a global neighborhood topology which helps speed up the convergence process. The combination of local and global neighborhood topologies is taken to boost the balance between exploration and exploitation.

*B. Dynamic neighborhood forming and **gbest** mutation based on convergence criteria*

Most of the conventional local neighborhood topologies, such as a ring [16] and a square [31], are static neighborhood topologies that remains unchanged during the evolutionary process [32, 33]. If particles are trapped in local optima, it is hard for them to escape due to no information exchange among them. To overcome this problem, in this paper, we propose a dynamic neighborhood generation scheme and a ***gbest*** mutation operation which are compatible with the evolutionary states, where the evolutionary states are depicted by two convergence criteria: limit value and population diversity.

1) Limit value: The ***gbest*** plays an important role in DNLGSA for it exerts directly attraction to other particles. If the ***gbest*** cannot improve self-solution after a certain number of sequential iterations, the other particles gradually get close to the ***gbest***. Especially in the early iterations, the ***gbest*** is likely to be a local optimum and leads to population stagnation. To estimate the evolutionary states, we set a counter *cnm* for the ***gbest*** as an indicator. Initially *cnm* is set to 0, and it is incremented by 1 if the ***gbest*** cannot improve self-solution at the end of current iteration. It is obvious that the ***gbest*** is more likely to be trapped as *cnm* increases. Here we set a limit value *gm* for the *cnm* and if *cnm* exceeds *gm*, we recognize the algorithm faces big risks of falling into stagnation. To prevent this trend, further operations should be carried out, including dynamic reformulation of local neighbors and mutation of ***gbest***, which will be decided based on the following population diversity (*PD*) indicator.

Generally, the value of *gm* should be neither too large nor too small. A large *gm* value tends to consume more computation resources due to excessive perturbation on ***gbest***, while a small value slows the convergence speed because particles will take a long time to search around the local optimum. In this study, *gm* = 5 is selected following the sensitivity analysis in Section IV-D.

2) Population diversity: As discussed above, to prevent the potential risks of stagnation when *cnm* exceeds *gm*, we need to calculate the *PD* to determine the operations of the next step. For a population consisting of M non-overlapping local neighborhoods, the geographical clustering center of each non-overlapping local neighborhood at time t can be calculated by:

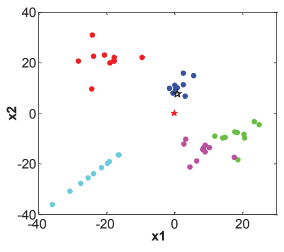
$$\boldsymbol{CDN}_i(t) = \frac{\sum_{\boldsymbol{x}_j \in \boldsymbol{DN}_i(t)} \boldsymbol{x}_j}{k} \tag{14}$$

where $\boldsymbol{DN}_i(t)$ is the i-th non-overlapping local neighborhood in time t which consists of k particles. Then we define the distribution diversity of the centers as below:

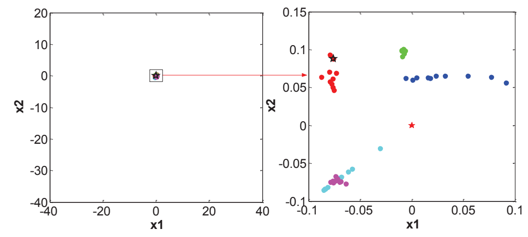
$$PD(t) = \frac{1}{M}\sum_{i=1}^{M}\sqrt{\sum_{d=1}^{D}(\boldsymbol{CDN}_i^d(t) - \overline{\boldsymbol{CDN}^d}(t))^2}$$
$$where\ \ \overline{\boldsymbol{CDN}^d}(t) = \frac{\sum_{i=1}^{M}\boldsymbol{CDN}_i^d(t)}{M} \tag{15}$$

where $\boldsymbol{CDN}_i^d(t)$ is the center position of the i-th non-overlapping neighborhood in the d-th dimension at time t, $\overline{\boldsymbol{CDN}^d}(t)$ is the average value over all the local neighborhoods at d-th dimension.

Accordingly, we can use the *PD* to delineate the distribution states of particles. When the ***gbest*** cannot improve self-solution after *gm* iterations, the *PD* has often two kinds of values, high or low, which represent two different evolutionary states of population as indicated in Fig. 2(a) and Fig. 2(b), respectively. Figure 2 is an experiment for STEP function (function f3 in [18]) with D=2, N=50 to reveal the two states. In Fig. 2, the sample points in 5 different colors represent 5 local neighborhoods and the red star represents the global optimum. As shown in Fig. 2(a), at early iterations i.e. t=19, in which *PD* has a high value, the particles scatter in a wider space and the ***gbest*** is far from the global optimum, it means the particles should pay more attention on exploration; while at later iterations i.e. t=211 as shown in Fig. 2(b), the population usually has low *PD* values and the particles are scattering in a limited space close to the global optimum, in this state the algorithm should concentrate on exploitation. Therefore, we can decide the next operations based on the *PD* results. Details of this figure were provided in Section S-I of the supplementary file.



(a) DNLGSA: *PD*=12.78, *t*=19.



(b) DNLGSA: *PD*=0.2584, *t*=211, where the figure to the right is the magnified version of the one to the left.

Figure 2: Population distribution of particles when the ***gbest*** has not been improved after sequential *gm* iterations.

*3) Dynamic reformulation of local neighborhoods and mutation for **gbest**:* Based on the above analysis, we can now construct the scheme to improve the performance of DNL-GSA. It includes two steps: (i) the tracking of the ***gbest*** and (ii) the reformulating of local neighborhoods and ***gbest*** mutation.

Step 1: The tracking of ***gbest***.

We first track the value of *cnm* at the end of current iteration. Once *cnm*>*gm*, it indicates that the ***gbest*** has not been improved after sequential *gm* iterations and we should calculate the *PD* to determine the operations of Step 2.

Step 2: The reformulating of local neighborhoods and ***gbest*** mutation.

In step 2, we first set a threshold *Th* for *PD* to distinguish the two evolutionary states and then execute corresponding operations:

① **Evolutionary state 1**: The value of *PD* is larger than *Th*. In this case, the particles should pay more attention to exploration to prevent the potential risks of stagnation as shown in Fig. 2(a). We here conduct the reformulation of the local neighborhoods because it realizes a timely information exchange among the non-overlapping local neighborhoods. Moreover, to improve the opportunities of escaping from the local optimum, the mutation of ***gbest*** with big jump steps is also introduced. In this paper, the scale of mutation is assigned proportional to the value of *PD* as follows:

$$newg^d(t) = g^d(t) + PD \cdot U(-1,1), \quad if \;\; PD > Th \tag{16}$$

where U(-1, 1) returns a uniformly distributed pseudorandom number in the interval [-1, 1]. The distribution interval of the pseudorandom number ensures diverse directions of mutation, thereby the ***gbest*** can fully explore the search space around itself in a sphere with a radius *r = PD*.

② **Evolutionary state 2**: The value of *PD* is not more than *Th*. This state usually means that the DNLGSA should focus on exploitation. To this end, we introduce a coefficient $RD \in (0,1)$ to the *PD* based mutation operation to adjust its step size as follow:

$$newg^d(t) = g^d(t) + RD \cdot PD \cdot U(-1,1), \quad if \;\; PD \leq Th \tag{17}$$

Apparently, with the coefficient *RD* the ***gbest*** can concentrate the search around the promising area. The sensitivity of parameter *RD* is conducted and 0.4 is selected (See Section S-II of the supplementary file for details).

Obviously, *Th* determines the time to reformulate the local neighborhoods and controls the scale of mutation and hence has significant effects on the balance of exploration and exploitation. The sensitively of *Th* is reported in Section S-III in the supplementary file. Accordingly, the thresholds *Th* is set to 0.5 in this study.

After steps 1-2, the *cnm* is set to zero. Moreover, to promote the evolution of the algorithm to the global optimum, we accept the new ***gbest*** if it is not worse than the previous one; otherwise, the previous ***gbest*** is kept. The whole implementation of the DNLGSA is summarized in **Algorithm** 1.

To analysis the characters of the proposed method, we conducted the convergence trajectories experiments as shown in Section S-IV in the supplementary file. Besides, a detailed analysis of the contribution of each of these components (dynamic neighborhoo ***gbest*** mutation, adaptive parameters) on various functions is given in Section S-V.

Algorithm 1 Dynamic neighborhood learning-based GSA (DNLGSA)

```
1: /*Initialization*/
2: for $i$=1 to $N$ do do
3:     Randomly initilize $\boldsymbol{v}_i$ and $\boldsymbol{x}_i$;
4:     Evaluate $fit(\boldsymbol{x}_i)$;
5:     $fit_g = min(\boldsymbol{fit})$, $\boldsymbol{g}$ is the global best particle;
6:     Randomly group the population to $M$ non-overlapping local neighborhoods;
7: end for
8: /*Main Loop*/
9: repeat
10:     Calculate $c_1$, $c_2$, and Mass;
11:     for $i$=1 to $N$ do do
12:         /*Particle Update*/
13:         Update $\boldsymbol{Lv}_i$ and $\boldsymbol{Gv}_i$ using Eqs. (10)-(11);
14:         Update $\boldsymbol{v}_i$ and $\boldsymbol{x}_i$ using Eqs. (12)-(13);
15:         Evaluate $fit(\boldsymbol{x}_i)$;
16:     end for
17:     $fit_{newg} = min(\boldsymbol{fit})$;
18:     if $fit_g < fit_{newg}$ then
19:         $fit_g = fit_g$;
20:         cnm=cnm+1;
21:         if cnm<gm then
22:             continue;
23:         else
24:             /*Dynamic reformulation of local neighborhoods and gbest mutation*/
25:             Calculate the PD of the population;
26:             if PD≤0.4 then
27:                 for d=1:D do
28:                     $newg^d = g^d + 0.4 \cdot PD \cdot U(-1,1)$;
29:                 end for
30:             else
31:                 Randomly regroup the non-overlapping local neighborhoods;
32:                 for d=1:D do
33:                     $newg^d = g^d + PD \cdot U(-1,1)$;
34:                 end for
35:             end if
36:             The newg exert direct attraction to all the particles.
37:             cnm=0;
38:         end if
39:         Evaluate the new particle $fit_{newg}$=newg;
40:         if $fit_{newg} < fit_g$ then
41:             $\boldsymbol{g} = \boldsymbol{newg}$ ;   /*The new gbest is adopt*/
42:         else
43:             $\boldsymbol{g} = \boldsymbol{g}$;   /*The previous gbest is kept*/
44:         end if
45:     else
46:         $fit_g = fit_{newg}$;
47:         cnm=cnm;
48:     end if
49: until Terminal Condition
```

C. Complexity Analysis of DNLGSA

The computational costs of the standard GSA involve the initialization (T_{ini}), evaluation (T_{eva}), calculation of the acceleration (T_{acc}) as well as velocity and position update (T_{upd}) for each particle. Assume D is the dimensionality of the search space and T_{max} is the maximum iterations for the algorithm with N particle. The computational complexity of GSA is $T_{GSA} = T_{ini} + (T_{eva} + T_{acc} + T_{upd}) \cdot T_{max} = 2 \cdot D \cdot N + (D \cdot N + D \cdot N^2 + 2 \cdot D \cdot N) \cdot T_{max} = 2 \cdot D \cdot N + 3D \cdot N \cdot T_{max} + D \cdot N^2 \cdot T_{max}$. As can be seen, the time complexity of the standard GSA is $O(D \cdot N^2 \cdot T_{max})$, which is proportional to $D \cdot N^2$.

In DNLGSA, the time complexity is determined by the computational costs of the modified GSA operation ($T_{mod-GSA}$) and the DNL operation (T_{DNL}). Similar to the standard GSA, the computational costs of the $T_{mod-GSA}$ involve the initialization (T_{ini}), evaluation (T_{eva}), calculation of acceleration ($T_{acc-DNLGSA}$) as well as velocity and position update (T_{upd}) for each particle. Note that the $T_{acc-DNLGSA}$ is different from the T_{acc} as particles are only directed by their N/Num local neighbors (*Num* is the number of local neighborhoods). Following Eq. (10), we can have $T_{acc-DNLGSA} = D \cdot (N/num)^2 \cdot Num$. Thereby, the costs of the modified GSA operation is $T_{mod-GSA} = T_{ini} + (T_{eva} + T_{acc-DNLGSA} + T_{upd}) \cdot T_{max} = 2 \cdot D \cdot N + 3D \cdot N \cdot T_{max} + D \cdot N^2 \cdot T_{max}/Num$.

The TDNL consists of the computational costs in calculating ***PD*** (T_{PD}), reformation of local neighborhoods (T_{ref}), and mutation for ***gbest*** (T_{mu}). For the worst-case, DNLGSA performs the calculation of ***PD*** every *gm* iterations, then $T_{PD} = T_{ref} = D{\cdot}N{\cdot}T_{max}/gm$ and $T_{mu} = D{\cdot}T_{max}/gm$, i.e. $T_{DNL} = D{\cdot}(N{+}1){\cdot}T_{max}/gm$. Therefore, the maximum time complexity of DNLGSA will be $T_{DNLGSA} = T_{mod-GSA} + T_{DNL} = 2 \cdot D \cdot N + 3D \cdot N \cdot T_{max} + D \cdot (N+1) \cdot T_{max}/gm + D \cdot N^2 \cdot T_{max}/Num$.

As we can see, DNLGSA has an $O(D \cdot N^2 \cdot T_{max}/Num)$ computational complexity. Note that $Num = N/\lceil 15\% \times N \rceil$, the computational complexity can be written as $O(D \cdot N \cdot \lceil 15\% \times N \rceil \cdot T_{max})$. When the value of N is small, the computational complexity is obviously lower than that of GSA as will be illustrated in Table III and Table V in Section IV-B1 and Section IV-B2.

IV. Experimental verification and comparisons

A. *Experimental setup*

To fully evaluate the performance of the proposed DNLGSA, 27 scalable benchmark functions with various features were tested in this study. They are classified in two test suites. The first test suite contains 13 (f1-f13) widely used functions, where f1-f7 are unimodal functions and f8-f13 are multimodal functions. The unimodal functions are usually utilized to investigate the convergence feature of algorithms while multimodal functions are more complex with numerous landscapes [34, 35]. Detailed description of these functions can be found in [18]. Optimization of these functions can reflect the exploration ability of an algorithm. The functions, f101-f114, form the second test suite. These functions are shifted and rotated functions from the 2015 IEEE Congress on Evolutionary Computation (CEC2015) test suite [36]. Among these 14 functions, f101-f102 are unimodal functions, f103-f105 are multimodal functions, f106-f108 are hybrid functions, and f109-f114 are composition functions as introduced in [36]. Performances of solving these shifted and rotated functions can reveal the capability of algorithms for solving more difficult problems. Experimental results of DNLGSA on these benchmark functions were compared with 5 established GSA variants: GSA [18], PSOGSA [27], GGSA [29], FGSA [37], and FLGSA [38].

TABLE I: Parameter settings.

Algorithm Parameter	settings
GSA	G_0=100, β=20, $k \in [N, 2]$
PSOGSA	G_0=100, β=20, c_1=0.5, c_2=1.5, $k \in [N, 2]$
GGSA	G_0=100, β=20, $c_1 = 2 - 2t^3/T_{\max}^3$, $c_2 = 2t^3/T_{\max}^3$, $k \in [N, 2]$
FGSA	$ED \in [0, 1]$, $CM \in [0, 1]$, $\beta \in [29, 31]$, $k \in [N, 2]$
FLGSA	$\beta \in [0, 150]$, $k \in [N, 2]$
DNLGSA	G_0=100, β=20, $c_1 = 0.5 - 0.5t^{1/6}/T_{\max}^{1/6}$, $c_2 = 1.5t^{1/6}/T_{\max}^{1/6}$, k=10, gm=5

The parameter settings for these 5 comparison algorithms are extracted from their corresponding literatures and listed in Table I. For a fair comparison, the population size (N) and the maximum fitness evaluation times (*Max_FEs*) of all algorithms are set to 50 and 1000 000, respectively. In the proposed DNLGSA, the initial value of gravitational constant (G_0) and its coefficient of decrease (β) are set to 100 and 20. The limit value and the number of neighbors of each particle are empirically set to *gm* = 5 and k=10, respectively.

To alleviate stochastic errors and obtain statistical results, each algorithm was repeated 30 times independently. We first assessed the searching accuracy of algorithms based on the best, mean, and standard deviation of fitness error (best, mean, and std). Note that the fitness error is the mean difference between the fitness value found by the algorithms and the actual global optimum. Then, searching reliability and convergence efficiency of DNLGSA were evaluated in terms of the minimum desired fitness evaluation times (*FEs*), the shortest desired CPU times (CPU), and successful rate (SR%). SR% reflects the reliability of an algorithm which stands for the percentage of the successful runs that acceptable solutions are found [16]. Moreover, the well-known nonparametric statistical hypothesis, Wilcoxon rank-sum test [39] is also utilized to evaluate whether the differences between the results are significant. In this paper, this test was conducted at a significance level of 5% (i.e. α = 0.05).

B. *Experimental results and comparison on Test Suite 1*

1) Results on unimodal functions: Experimental results from the unimodal functions (f1-f7) are reported in Tables II-III. The average (mean), standard deviation (std), best (best), and Wilcoxon test (p-val) obtained from each algorithm are summarized in Table II for comparison. The best results of each row are marked in bold. Moreover, at the bottom of Table II, we summarize the overall comparison results between DNLGSA and other algorithms with "$w/t/l$' and "#BME'. Here "$w/t/l$' gives the numbers that DNLGSA wins the particular peer in w functions, ties in t functions, and loses in l functions, and #BME denotes the number of best mean value achieved by each algorithm. The Wilcoxon test results (WTRs) are summarized as "+/=/-" to denote the number of functions that DNLGSA performs significantly better, comparable, or significantly worse than its counterparts, respectively. As can be seen, DNLGSA produces the best mean results with superior exploitation in optimizing 6 out of the 7 unimodal functions. All the 6 GSA variants could not handle the function f5 well. The statistical hypothesis results shown in Table II also reveal that DNLGSA significantly outperforms other peers for most functions tested. This may result from the cooperation of local and global neighborhoods and the mutation scheme as well as the population diversity of particles.

The search speed and reliability of the 6 algorithms are compared in Table III. Similar to Table III, the best results of each row are marked in bold. As we can see, with respect to the minimum desired FEs, DNLGSA shows the highest convergence efficiency. This may benefits from the direct attraction of the ***gbest***. Moreover, in terms of the CPU times, DNLGSA also stands out for fast convergence. This confirms that the proposed neighborhood learning strategy can reduce the computational complexity of GSA. It is also obvious that DNLGSA yields the highest average successful rate on all the tested unimodal functions.

TABLE II: Statistical results on functions f1-f7.

Function		GSA	PSOGSA	GGSA	FGSA	FLGSA	DNLGSA
f1	mean	6.33E-18	9.50E-20	1.51E-18	1.90E+00	2.11E-02	**1.53E-27**
	best	9.88E-18	3.02E+02	3.63E-18	2.34E+00	8.33E-02	**2.14E-27**
	std	3.51E-19	1.56E+00	3.49E-18	4.06E+00	7.21E-02	**5.11E-26**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f2	mean	1.32E-08	2.53E-09	4.22E-09	5.60E+00	5.67E-01	**1.66E-14**
	best	3.84E-08	1.50E+01	1.01E-08	7.99E+00	7.65E-01	**2.07E-14**
	std	1.74E-09	1.13E+01	3.93E-09	3.88E+00	6.27E-01	**5.06E-14**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f3	mean	1.25E+00	4.67E+05	6.06E-17	1.76E-08	1.37E-10	**1.83E-26**
	best	2.04E+00	2.01E+04	5.35E+04	1.57E-09	8.37E-11	**3.05E-26**
	std	4.39E-01	2.67E+03	5.27E+03	3.48E-08	5.56E-10	**6.96E-26**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f4	mean	1.32E-09	3.93E-10	8.67E-10	6.67E-01	7.71E-02	**2.98E-15**
	best	9.45E-10	1.64E-10	2.54E-10	3.66E-01	6.91E-02	**3.82E-16**
	std	8.04E-09	3.22E-10	5.29E-10	9.78E+00	5.83E-02	**8.39E-15**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f5	mean	3.41E+00	5.11E+01	**1.53E+01**	1.12E+03	3.16E+01	2.33E+01
	best	2.17E+01	**1.35E+01**	1.36E+01	2.63E+01	2.67E+01	2.02E+01
	std	**6.72E+00**	4.35E+00	5.41E+00	5.02E+02	1.95E+00	3.24E+01
	p-val	0.01+	0.72	0.00-	1.02	0.00+	
f6	mean	**0**	**0**	**0**	1	**0**	**0**
	best	**0**	**0**	**0**	1	**0**	**0**
	std	**0**	**0**	**0**	0	**0**	**0**
	p-val	#	#	#	0.00+	#	
f7	mean	7.57E-03	7.25E+01	1.61E+01	1.92E+01	8.51E-03	**1.09E-07**
	best	3.49E-02	8.19E+01	3.72E+01	2.14E+01	9.75E-03	**2.49E-07**
	std	4.55E-03	5.19E+00	5.52E+01	3.16E+01	6.08E-03	**6.38E-07**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
w/t/l		6/1/0	6/1/0	6/0/1	7/0/0	6/1/0	
#BME		1	1	2	0	1	**6**
+/=/-		6/1/0	6/1/0	6/0/1	6/1/0	6/1/0	

TABLE III: Search speed and reliability on functions f1-f7.

Function		GSA	PSOGSA	GGSA	FGSA	FLGSA	DNLGSA
f1	FEs	211150	178000	228100	N/A	N/A	**38950**
	CPU	30.62	27.12	34.31	N/A	N/A	**2.18**
	SR%	**100**	40	**100**	0	0	**100**
f2	FEs	392000	325000	404600	N/A	N/A	**93900**
	CPU	57.50	52.06	60.88	N/A	N/A	**5.19**
	SR%	**100**	40	**100**	0	0	**100**
f3	FEs	N/A	N/A	24050	1500	1500	650
	CPU	N/A	N/A	4.49	0.32	0.26	0.04
	SR%	0	0	25	**100**	**100**	**100**
f4	FEs	273900	241200	302750	N/A	N/A	**39000**
	CPU	38.31	33.88	42.32	N/A	N/A	**1.96**
	SR%	**100**	**100**	**100**	0	0	**100**
f5	FEs	N/A	N/A	N/A	N/A	N/A	N/A
	CPU	N/A	N/A	N/A	N/A	N/A	N/A
	SR%	0	0	0	0	0	0
f6	FEs	82950	57600	119850	N/A	33250	**5550**
	CPU	11.92	8.28	17.45	N/A	5.70	**0.29**
	SR%	**100**	**100**	**100**	0	**100**	**100**
f7	FEs	192300	N/A	N/A	N/A	49750	**2650**
	CPU	28.94	N/A	N/A	N/A	9.65	**0.15**
	SR%	**50**	0	0	0	**100**	**100**
Avg-SR%		64.29	40	60.71	14.29	42.86	**85.71**

2) Results on multimodal functions: It is found that without strong global search ability the algorithms can be trapped in local optima when solving multimodal problems [40]. In this section, 6 multimodal functions (f8-f13) are utilized to validate the exploration ability of algorithms, and the results are given in Tables IV-V. As seen, DNLGSA can find acceptable solutions on all the multimodal functions except f8 and f13. Function f8 is a complex multimodal problem with a significant number of local optima and none of the tested algorithms can solve this problem [34]. Generally speaking, DNLGSA performs the best in 4 out of the 6 multi-modal functions in terms of solution accuracy. Although DNLGSA has difficulty in processing the function f13, it is the only one performs well on f9. The WRTs shown at the bottom of Table IV also reveal that DNLGSA produces significant superiority than the other 5 algorithms on functions f8-f11.The superiority may come from the dynamic adjustment of the neighborhood which diversifies the population and enhances the global search ability. In terms of the search speed and reliability shown in Table V, DNLGSA is the most suitable algorithm on the tested multimodal functions. This confirms the lower complexity of DNLGSA.

TABLE IV: Statistical results on functions f8-f13.

Function		GSA	PSOGSA	GGSA	FGSA	FLGSA	DNLGSA
f8	mean	1.51E+04	1.07E+04	8.44E+03	9.26E+03	9.82E+03	**6.22E+03**
	best	9.88E+03	6.22E+03	6.87E+03	9.11E+03	9.24E+03	**5.23E+03**
	std	**2.07E+03**	4.72E+03	6.37E+03	8.24E+03	1.35E+04	3.92E+03
	p-val	0.00+	0.01+	0.21	0.00+	0.00+	
f9	mean	2.35E+01	1.31E+02	1.98E+02	3.00E+02	2.78E+01	**0**
	best	1.39E+01	1.39E+01	1.01E+02	1.57E+02	1.75E+01	**0**
	std	6.40E+01	8.42E+01	5.07E+01	4.91E+02	2.17E+01	**0**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f10	mean	2.19E-09	2.86E-10	8.73E-10	2.44E+00	1.29E-01	**6.84E-14**
	best	1.07E-09	2.14E-10	4.14E-10	2.05E+00	2.11E-02	**5.66E-15**
	std	3.45E-09	6.94E-10	8.23E-10	6.08E+00	1.93E-01	**5.46E-14**
	p-val	0.00+	0.00+	0.00+	0.00+	0.00+	
f11	mean	**0**	4.18E-02	3.94E-02	1.18E-01	1.20E-03	**0**
	best	**0**	3.59E-02	2.41E-02	8.44E-02	7.99E-04	**0**
	std	**0**	4.28E-02	6.52E-02	1.35E-01	4.91E-03	**0**
	p-val	#	0.00+	0.00+	0.00+	0.00+	
f12	mean	5.02E-20	**6.20E-22**	6.44E-21	2.56E-02	3.47E-04	1.41E-05
	best	3.18E-21	2.45E-22	2.06E-21	3.15E-03	1.96E-04	**0**
	std	6.15E-20	7.54E-21	**4.66E-21**	5.73E-03	4.79E-04	3.25E-07
	p-val	0.00-	0.00-	0.00-	0.00+	0.00+	
f13	mean	3.37E-19	**1.36E-20**	7.56E-20	1.84E+00	7.36E-03	2.91E+00
	best	5.67E-20	**1.02E-20**	3.59E-20	1.24E+00	2.54E-03	1.53E-01
	std	6.09E-19	**2.38E-20**	7.20E-20	2.54E+00	3.68E-03	2.73E+00
	p-val	0.00-	0.00-	0.00-	0.37	0.00-	
w/t/l		3/1/2	4/0/2	4/0/2	5/0/1	6/0/0	
#BME		1	2	0	0	0	**4**
+/=/-		3/1/2	4/0/2	5/1/0	5/1/0	6/0/0	

TABLE V: Search speed and reliability on functions f8-f13.

Function		GSA	PSOGSA	GGSA	FGSA	FLGSA	DNLGSA
f8	FEs	N/A	N/A	N/A	N/A	N/A	N/A
	CPU	N/A	N/A	N/A	N/A	N/A	N/A
	SR%	0	0	0	0	0	0
f9	FEs	N/A	N/A	N/A	N/A	N/A	**58700**
	CPU	N/A	N/A	N/A	N/A	N/A	**3.38**
	SR%	0	0	0	0	0	**100**
f10	FEs	306050	267800	319100	N/A	N/A	**50100**
	CPU	44.61	39.96	47.62	N/A	N/A	**2.70**
	SR%	**100**	**100**	**100**	0	0	**100**
f11	FEs	133000	N/A	N/A	N/A	43950	**48200**
	CPU	20.03	N/A	N/A	N/A	8.00	**2.85**
	SR%	**100**	0	0	0	**100**	**100**
f12	FEs	97100	62200	114600	N/A	38050	**6700**
	CPU	16.81	10.76	20.00	N/A	7.33	**0.52**
	SR%	**100**	**100**	**100**	0	**100**	**100**
f13	FEs	161450	131950	186500	N/A	**48400**	N/A
	CPU	27.76	22.62	33.33	N/A	**9.49**	N/A
	SR%	**100**	**100**	**100**	0	**100**	0
Avg-SR%		**66.67**	50	50	0	50	**66.67**

C. Experimental results and comparison on Test Suite 2

In this section, we present more experiments which were conducted on the shifted and rotated functions selected from CEC2015 for further performance assessment of DNLGSA. The mean and standard deviation of fitness error produced by DNLGSA and the eight benchmarking algorithms are given in Table VI. Again the best value in each row is marked in bold. It can be observed that DNLGSA is a competitive algorithm on the second test suite as well. In the 14 tested functions, DNLGSA is ranked the first for 8 times, and the second for 3 times in Table VI. The corresponding WTRs are shown in Table VII, where the superiority of DNLGSA is verified on these 14 benchmark functions in comparison to 5 state-of-the-art algorithms.

TABLE VI: Statistical results on Test Suite 2 (mean error ± standard deviation).

Function	f101	f102	f103	f104	105	f106	f107
GSA	7.63E+08 ± 3.02E+09	1.24E+10 ± 7.92E+09	2.01E+01 ± 1.24E-01	2.59E+02 ± 4.96E+01	3.95E+03 ± 7.97E+02	4.88E+06 ± 4.23E+06	1.44E+02 ± 3.66E+02
PSOGSA	8.63E+06 ± 2.28E+06	8.58E+07 ± 1.22E+07	2.03E+01 ± 2.13E-02	6.31E+01 ± 6.56E+01	2.23E+03 ± 2.87E+02	4.14E+05 ± 9.47E+04	1.10E+01 ± 2.18E+00
GGSA	3.61E+05 ± **3.94E+05**	8.01E+07 ± 3.03E+08	**2.00E+01** ± **0.00E+00**	3.79E+01 ± 3.62E+01	2.83E+03 ± 3.96E+02	6.81E+05 ± 5.77E+05	6.12E+01 ± 3.26E+01
FGSA	5.23E+06 ± 2.12E+06	**4.97E-14** ± **1.72E-14**	**2.00E+01** ± 4.24E-07	8.09E+02 ± 1.64E+02	4.29E+03 ± 1.75E+02	**1.57E+03** ± **3.01E+03**	1.67E+01 ± 3.15E+01
FLGSA	**2.83E+05** ± 7.09E+06	9.37E+09 ± 3.56E+10	2.02E+01 ± 3.65E-02	5.04E+02 ± 1.14E+02	2.16E+03 ± **1.34E+02**	2.12E+05 ± 1.40E+05	6.33E+01 ± 3.42E-01
DNLGSA	1.86E+06 ± 2.64E+06	6.42E+07 ± 3.97E+07	**2.00E+01** ± **0.00E+00**	**1.53E+01** ± **3.29E+01**	**1.28E+03** ± 3.00E+03	4.07E+05 ± 3.09E+05	**1.00E+01** ± **0.00E+00**
rank-DNLGSA	3	2	1	1	1	2	1
Function	f108	f109	f110	f111	f112	f113	f114
GSA	1.73E+06 ± 6.28E+05	1.10E+02 ± 4.40E+02	4.09E+06 ± 6.77E+06	4.72E+02 ± 9.81E+03	1.52E+02 ± 6.39E+02	4.20E+00 ± 3.48E+00	7.49E+04 ± 9.26E+03
PSOGSA	5.42E+05 ± 7.39E+05	1.19E+02 ± 2.17E+03	2.83E+03 ± 7.01E+03	3.34E+02 ± 7.80E+02	1.06E+02 ± 7.83E+02	2.93E-02 ± 2.26E-02	3.55E+04 ± 7.58E+03
GGSA	6.53E+03 ± **1.22E+03**	**1.03E+02** ± **3.02E+01**	7.82E+04 ± 3.96E+04	2.37E+02 ± 4.72E+02	2.00E+02 ± 5.64E+02	**9.35E-03** ± **9.22E-03**	**8.06E+02** ± **1.00E+02**
FGSA	1.81E+06 ± 5.68E+07	6.22E+02 ± 5.23E+03	**1.05E+03** ± **4.70E+02**	4.03E+02 ± 9.34E+02	3.62E+02 ± 1.73E+02	4.15E+02 ± 9.43E+02	3.31E+04 ± 5.89E+03
FLGSA	6.63E+03 ± 5.66E+03	1.45E+02 ± 1.33E+02	2.14E+04 ± 1.45E+04	3.18E+02 ± 4.47E+02	1.88E+02 ± 1.53E+02	2.72E-01 ± 1.21E-02	3.31E+04 ± 4.19E+03
DNLGSA	**1.86E+03** ± 5.62E+03	**1.03E+02** ± 3.12E+01	1.05E+07 ± 2.74E+07	**1.30E+02** ± **3.66E+02**	**1.00E+02** ± **7.08E+01**	1.53E-01 ± 5.29E-01	3.12E+04 ± 6.04E+04
rank-DNLGSA	1	1	6	1	1	3	2

TABLE VII: WTRs between DNLGSA and other compared algorithms on Test Suite 2.

Algorithm	GSA	PSOGSA	GGSA	FGSA	FLGSA
Sig-Better	13	12	9	10	12
Sig-Worse	1	2	4	3	2

D. Sensitivity analysis of parameters

To analyze the impact of the parameters of DNLGSA, we performed parameter sensitivity analysis on the limit value (*gm*) and the size of neighborhood (k) in this section. Four functions, including two unimodal functions (f1, f2) and two multimodal functions (f10, f13) were utilized.

Firstly, we conduct DNLGSA with $k = [5\% \times N, 10\% \times N, ..., 50\% \times N]$. The mean fitness error values obtained from the aforementioned 4 functions are depicted in Fig. 3 for comparison. As shown, too small k can cause DNLGSA excessively emphasizes on local search in the early iterations and gets stuck in local optima. On the contrary, a large k can lead the particles to be attracted by jumbled information and thereby cannot perform properly exploitation in the latter iterations. In addition, too large values of k would lead to quick loss of population diversity and make DNLGSA suffer from premature convergence. It can be observed that the proper range of k value is $[15\% \times N, 30\% \times N]$, where $k = \lceil 15\% \times N \rceil$ is recommend in this paper for computational efficiency.

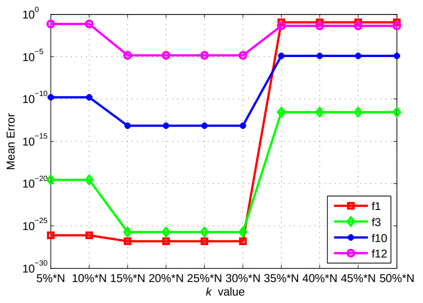


Fig. 3: Effects of the size of local neighborhood.

We also conducted the sensitivity analysis for selection of the limit value *gm*. The experiments were carried out with *gm*=$[1, 2, ..., 10]$. Fig. 4 plots the effects of different settings of *gm* on the performance of DNLGSA. It shows that in the range of *gm*=$[1, 2, 3, 4, 5]$, the higher value of *gm* is, the better performance can be obtained. As the parameter *gm* controls whether the dynamically reformulation of local neighborhoods and mutation for ***gbest*** are achieved, too large *gm* will lead to insufficient information exchange in the early evolutionary state and thereby weaken the performance of DNLGSA. Accordingly, *gm*=5 is recommended in this paper.

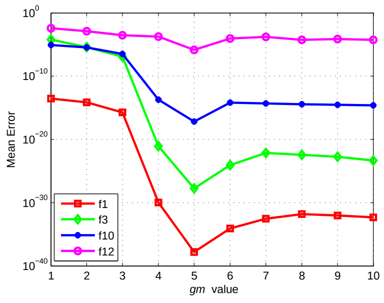


Fig. 4: Effects of the limit value *gm* on the performance of DNLGSA.

E. Scalability Analysis

Increase of scales often causes drastically degradation of searching accuracy. In this section, we present scalability analysis for the proposed DNLGSA to explore its feasibility to high dimensional problems. Experiments were carried out on 50-D and 100-D problems. Other parameters are kept the same as given in Section IV-A. Experimental results, i.e. the mean fitness error values of each function are listed in Table VIII.

As shown in Table VIII, except for function f13, DNLGSA performs the best on all the tested functions when D=50. Similar to the results shown in Table IV, the DNLGSA seems have difficulty for deal with this problem. When D=100, DNLGSA also shows superiority on functions f1-f11. Although the result from DNLGSA for f12 is slightly worse than that of GSA, it is noticeably better than other GSA variants. This confirms the validity of DNLGSA for solving up to 100 dimensional problems, especially for the multimodal problems f9-f11 with many local optima. Possible reasons contributing to its superiority include the evolutionary states based dynamic reformulation of local neighborhoods

TABLE VIII: Results of scalability analysis on 50-D and 100-D problems.

Function		GSA	PSOGSA	GGSA	FGSA	FLGSA	DNLGSA
f1	50D	2.52E-16	3.55E-10	3.14E-16	3.14E+00	5.15E-02	**2.31E-25**
	100D	9.70E-16	4.62E-10	6.44E-15	4.39E-01	1.92E+01	**5.24E-23**
f2	50D	7.51E-08	2.06E-08	6.39E-08	1.75E-05	2.70E+00	**6.66E-14**
	100D	9.95E-08	2.93E+00	1.18E-06	2.21E+00	1.92E+00	**7.18E-13**
f3	50D	2.31E+02	3.73E+04	9.97E-08	5.62E-06	2.07E-06	**1.10E-22**
	100D	3.90E+01	3.72E+04	2.66E-03	2.11E+02	1.81E-05	**2.63E-19**
f4	50D	7.48E-06	3.91E-07	5.95E-03	6.25E+00	8.29E+00	**7.92E-13**
	100D	2.53E-02	4.15E+00	5.56E-02	5.11E+01	3.41E+00	5.78E-08
f5	50D	4.13E+01	5.99E+01	4.91E+02	4.55E+01	2.81E+02	**2.53E+01**
	100D	5.26E+01	4.24E+03	4.23E+02	3.16E+03	1.74E+02	**3.05E+01**
f6	50D	**0**	1.00E+00	3.00E+00	1.00E+00	3.00E+00	**0**
	100D	1.00E+00	2.12E+04	3.49E+03	3.22E+03	3.24E+03	**0**
f7	50D	1.11E+00	5.25E+01	1.18E+02	4.24E+02	1.36E+00	**6.84E-05**
	100D	5.31E+00	5.92E+02	6.01E+01	5.23E+02	3.05E+00	**2.56E-04**
f8	50D	9.12E+04	3.65E+04	1.61E+03	8.73E+03	8.50E+03	**8.59E+01**
	100D	8.03E+04	1.69E+05	1.09E+04	8.63E+03	8.32E+03	**1.38E+03**
f9	50D	4.31E+01	1.12E+02	2.05E+02	1.04E+02	3.17E+01	**0**
	100D	4.27E+02	5.40E+02	6.23E+02	3.28E+02	5.02E+01	**0**
f10	50D	7.09E-09	3.06E-06	5.32E-09	4.21E+01	3.74E+00	**1.77E-12**
	100D	1.35E-08	3.59E-04	1.33E-03	2.49E+01	2.89E+00	**3.22E-11**
f11	50D	**0**	6.19E+01	3.25E+00	6.88E+00	5.99E+00	**0**
	100D	**0**	4.22E+01	4.58E+00	1.19E+01	2.04E+01	**0**
f12	50D	7.20E-02	5.61E+00	2.15E+00	1.56E+00	1.23E+00	**3.04E-03**
	100D	**1.79E-02**	2.68E+04	5.49E+00	3.57E+00	3.43E+00	4.22E-02
f13	50D	**1.89E-17**	2.57E-15	1.55E-13	5.45E-02	1.94E-01	6.84E+00
	100D	**6.23E-12**	1.98E-03	4.14E+01	3.85E+01	3.50E+01	6.02E+00

and effective mutation for ***gbest***. Current evolutionary states are identified by calculating two convergence criteria: limit value and population diversity, where the information can then be used to guide the learning behavior of particles to diversify the population, and thereby improve exploration ability of DNLGSA.

F. Comparisons with other M-HS algorithms

To further evaluate the proposed DNLGSA, we compared DNLGSA with 9 other M-HS algorithms: DE/BBO (Hybrid Differential Evolution with Biogeography-Based Optimization) [41], DSA (Differential Search Algorithm) [42], BSA (Backtracking Search Optimization Algorithm) [43], CS (Cuckoo Search Algorithm) [44], CoBiDE (Differential Evolution based on Covariance Matrix Learning and Bimodal Distribution Parameter Setting) [45], CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [46], PSO (Particle Swarm Optimization) [5], ABC (Artificial Bee Colony) [47], and (composite DE) CoDE [48]. Experiments were conducted over the 13 traditional problems and 14 CEC2015 functions with D=30 and N=50. For each function, we run DNLGSA and the 10 M-HS algorithms 30 times and the stopping criterion is *Max_FEs*=100000. Parameter settings of the benchmarking M-HS algorithms are given as follows.

ABC: *limit*=D**N*, *number of employed bees*=size of colony/2 as in [47]; DE/BBO: F=rand (0.1, 1.0), CR=0.9, mutation scheme: DE/*rand*/1/bin as in [41]; BSA: *mixrate*=1.00 as in [43]; DSA: p_1=p_2=3**rand* as in [42]; PSO: c_1=c_2=2, ω=0.9-0.4 as in [5]; CS: α_{cs}=1, p_a=0.25 as in [44]; CoBiDE: p_b=0.4, p_s=0.5 as in [45]; CMA-ES: σ=0.25, $\mu = \lfloor (4 + \lfloor 2 \cdot log(N) \rfloor)/2 \rfloor$, $N = \lfloor (4 + \lfloor 3 \cdot log(D) \rfloor \rfloor$ as in [49]; CoDE: randomly combine 3 trial vector generation strategies (*rand*/1/bin, *rand*/2/bin, current-to-*rand*/1) with 3 control parameter settings ([F=1.0, C_r=0.1], [F=1.0, C_r=0.9], [F=0.8, C_r=0.2]) at each generation as in [48].

Because of the space limitation we only reported in Table IX the ranks and Wilcoxon Test Results obtained from the 10 algorithms, where the detailed convergence data is presented in Section S-VI in the Supplementary file. Ranking of the results obtained Pairwise WTRs are also provided at the bottom of the Table, where "+" and "-" respectively indicate DNLGSA performs significantly better or worse than its peer. Also "=" means that DNLGSA produces comparable performance to others. From Table IX we can conclude that for the 27 test functions, our proposed GNLGSA approach has produced the best results in 16 functions, followed by CMA-ES which has yielded the best results for 11 functions. For the WTRs, as can be seen, DNLGSA has yielded statistically better results than most of the other M-HS algorithms on 21 out of the 27 functions through it was defeated by all the other compared algorithms on f110. These superiorities demonstrate the great potential of our proposed approach. Besides, according to different test functions, other approaches perform quite diversely as analyzed below. For f6, all the algorithms have achieved the global optimum except PSO, yet PSO produced the best results on f110. Moreover, although the CMA-ES can obtain comparable or even better performance than DNLGSA on several functions, it is much worse on many other functions, especially on f7, f9, f10, and f104. These experimental results reveal that none of the meta-heuristic algorithm can obtain the best results on all optimization problems. This on one hand has shown the no free lunch mechanism [50]. On the other hand, it reveals the space for further improvement of the M-HS algorithms.

V. Conclusions

In this paper, a novel variant of GSA called DNLGSA is presented to provide better tradeoff between exploration and exploitation based on the evolutionary states. DNLGSA is characterized by a dynamic neighborhood-based learning strategy, in which the local neighborhood and global neighborhood topologies are combined. Thereby in this learning strategy, each particle learns search information from 1) all the particles in its dynamic neighborhood and 2) the historically best experience of the population (donated by ***gbest***). For adaptively forming local neighborhoods according to the evolutionary states, two convergence criteria for evaluating the evolutionary states are presented. Additionally, an adaptive mutation operator for the ***gbest*** is introduced on the basis of the two convergence criteria to alleviate the problem of premature convergence. Learning from the dynamic neighborhood enables the algorithm to more sufficiently explore the feasible search space while learning from the ***gbest*** offers the fast convergence towards the optimum. Compared to the K_{best} model, the small size of local neighborhood used in DNLGSA reduces the computational complexity, a major problem in the baseline GSA. In summary, the synergy of the two components is utilized in DNLGSA to achieve adaptive balance between exploration and exploitation as well as to speed up the convergence process.

For performance assessment of DNLGSA, 27 benchmark functions with different features were tested in this paper. In comparison to 5 GSA variants and 9 state-of-the-art M-HS algorithms, the experimental results have demonstrated

TABLE IX: Comparisons between DNLGSA and other M-HS algorithms.

Function	DE/BBO rank (WTR)	DSA rank (WTR)	BSA rank (WTR)	CS rank (WTR)	ABC rank (WTR)	PSO rank (WTR)	CoBiDE rank (WTR)	CMA-ES rank (WTR)	CoDE rank (WTR)	DNLGSA rank (WTR)
f1	3(+)	5(+)	4(+)	6(+)	9(+)	10(+)	7(+)	1(-)	8(+)	2
f2	3(+)	4(+)	5(+)	6(+)	8(+)	10(+)	7(+)	2(+)	9(+)	1
f3	1(-)	9(+)	6(-)	5(-)	2(-)	10(+)	3(-)	8(-)	4(-)	7
f4	8(+)	6(+)	5(+)	3(+)	10(+)	9(+)	4(+)	2(+)	7(+)	1
f5	8(+)	7(+)	6(+)	2(+)	10(+)	9(+)	3(+)	1(-)	5(+)	4
f6	1(=)	1(=)	1(=)	1(=)	1(=)	2(+)	1(=)	1(=)	1(=)	1
f7	7(+)	5(+)	3(+)	2(+)	9(+)	8(+)	4(+)	10(+)	6(+)	1
f8	7(+)	7(+)	6(+)	4(+)	9(+)	3(+)	8(+)	2(+)	5(+)	1
f9	3(+)	2(+)	4(+)	7(+)	10(+)	6(+)	5(+)	9(+)	8(+)	1
f10	2(+)	4(+)	3(+)	7(+)	8(+)	9(+)	5(+)	10(+)	6(+)	1
f11	2(+)	3(+)	4(+)	6(+)	8(+)	9(+)	5(+)	1(=)	7(+)	1
f12	2(-)	3(-)	4(-)	8(+)	10(+)	9(+)	5(-)	1(-)	7(-)	6
f13	2(-)	4(-)	3(-)	6(-)	10(+)	9(+)	5(-)	1(-)	7(-)	8
f101	7(+)	8(+)	5(+)	9(+)	10(+)	6(+)	2(=)	1(-)	4(+)	3
f102	2(-)	6(=)	4(-)	9(+)	10(+)	3(-)	5(-)	1(-)	8(=)	7
f103	6(+)	2(+)	3(+)	7(+)	8(+)	5(+)	4(+)	1(=)	4(+)	1
f104	6(+)	4(+)	2(+)	8(+)	10(+)	3(+)	5(+)	9(+)	7(+)	1
f105	8(+)	3(+)	2(+)	9(+)	10(+)	5(+)	7(+)	4(+)	6(+)	1
f106	6(+)	8(+)	7(+)	9(+)	10(+)	5(=)	2(-)	1(-)	3(-)	4
f107	2(+)	6(+)	3(+)	8(+)	9(+)	5(+)	7(+)	4(+)	6(+)	1
f108	5(+)	7(+)	4(+)	8(+)	9(+)	6(+)	2(+)	3(+)	3(+)	1
f109	2(+)	3(+)	2(=)	6(+)	5(+)	2(=)	2(=)	1(=)	4(+)	1
f110	2(-)	3(-)	5(-)	8(-)	9(-)	1(-)	7(-)	4(-)	6(-)	10
f111	3(+)	4(+)	9(+)	9(+)	10(+)	2(+)	8(+)	5(+)	7(+)	1
f112	2(+)	4(+)	7(+)	7(+)	8(+)	5(+)	6(+)	2(+)	5(+)	1
f113	3(-)	2(-)	4(+)	8(+)	10(+)	7(+)	1(-)	9(+)	5(-)	6
f114	4(+)	7(+)	9(+)	9(+)	10(+)	8(+)	1(-)	3(=)	5(+)	2
Rank_sum	107	127	120	177	232	166	121	97	153	75
Rank_Final	3	6	4	9	10	8	5	2	7	**1**
Sig_Better	20	21	21	23	24	23	16	13	19	
Sig_Worse	6	4	4	3	2	2	8	9	6	

significant superiority of DNLGSA in most cases. DNLGSA generally shows more rapid convergence ability, lower computational complexity, higher convergence accuracy, and better flexibility. In the future work, we will expand the applicability of DNLGSA to a diverse class of optimization problems, such as discrete, mixed, and multi-objective search spaces.

References

[1] S. Mirjalili, A. Lewis, and A. S. Sadiq, "Autonomous particles groups for particle swarm optimization," *Arabian Journal for Science and Engineering*, vol. 39, no. 6, pp. 4683–4697, 2014.

[2] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3-4, pp. 267–289, 2010.

[3] S. Sarafrazi, H. Nezamabadi-Pour, and S. Saryazdi, "Disruption: a new operator in gravitational search algorithm," *Scientia Iranica*, vol. 18, no. 3, pp. 539–548, 2011.

[4] X.-S. Yang, S. Deb, and X. He, "Eagle strategy with flower algorithm," in *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*. IEEE, 2013, pp. 1213–1217.

[5] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. IEEE, 1998, pp. 69–73.

[6] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 1, pp. 58–73, 2002.

[7] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3. IEEE, 1999.

[8] X. Li, M. Yin, and Z. Ma, "Hybrid differential evolution and gravitation search algorithm for unconstrained optimization," *Int. J. Phys. Sci*, vol. 6, no. 25, pp. 5961–5981, 2011.

[9] Y. Wang and L. Li, "Heterogeneous redundancy allocation for series-parallel multi-state systems using hybrid particle swarm optimization and local search," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 42, no. 2, pp. 464–474, 2012.

[10] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *Cybernetics, IEEE Transactions on*, vol. 45, no. 2, pp. 191–204, 2015.

[11] J. Kennedy, R. Eberhart *et al.*, "Particle swarm optimization," in *Proceedings of IEEE international conference on neural networks*, vol. 4, no. 2. Perth, Australia, 1995, pp. 1942–1948.

[12] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3. IEEE, 1999.

[13] G. Toscano-Pulido, A. J. Reyes-Medina, and J. G. Ramírez-Torres, "A statistical study of the effects of neighborhood topologies in particle swarm optimization," in *Computational Intelligence*. Springer, 2011, pp. 179–192.

[14] A. E. M. Zavala, "A comparison study of PSO neighborhoods," in *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*. Springer, 2013, pp. 251–265.

[15] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews*, vol. 36, no. 4, p. 515, 2006.

[16] W.-N. Chen, J. Zhang, Y. Lin, N. Chen, Z.-H. Zhan, H.-H. Chung, Y. Li, and Y.-H. Shi, "Particle swarm optimization with an aging leader and challengers," *Evolutionary Computation, IEEE Transactions on*, vol. 17, no. 2, pp. 241–258, 2013.

[17] Z. Beheshti and S. M. Shamsuddin, "Non-parametric particle swarm optimization for global optimization," *Applied Soft Computing*, vol. 28, pp. 345–359, 2015.

[18] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[19] Z. Beheshti and S. M. H. Shamsuddin, "CAPSO: centripetal accelerated particle swarm optimization," *Information Sciences*, vol. 258, pp. 54–79, 2014.

[20] G. Sun, A. Zhang, Z. Wang, Y. Yao, J. Ma, and G. D. Couples, "Locally informed gravitational search algorithm," *Knowledge-Based Systems*, vol. 104, pp. 134–144, 2016.

[21] B. Shaw, V. Mukherjee, and S. Ghoshal, "A novel opposition-based gravitational search algorithm for combined economic and emission dispatch problems of power systems," *International Journal of Electrical Power & Energy Systems*, vol. 35, no. 1, pp. 21–33, 2012.

[22] B. Gu and F. Pan, "Modified gravitational search algorithm with particle memory ability and its application," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 11, pp. 4531–4544, 2013.

[23] S. Jiang, Y. Wang, and Z. Ji, "Convergence analysis and performance of an improved gravitational search algorithm," *Applied Soft Computing*, vol. 24, pp. 363–384, 2014.

[24] A. Zhang, G. Sun, Z. Wang, and Y. Yao, "A hybrid genetic algorithm and gravitational search algorithm for global optimization," *Neural Network World*, vol. 25, no. 1, p. 53, 2015.

[25] R.-E. Precup, R.-C. David, E. M. Petriu, M.-B. Radac, and S. Preitl, "Adaptive gsa-based optimal tuning of pi controlled servo systems with reduced process parametric sensitivity, robust stability and controller robustness," *Cybernetics, IEEE Transactions on*, vol. 44, no. 11, pp. 1997–2009, 2014.

[26] M. Doraghinejad and H. Nezamabadi-pour, "Black hole: A new operator for gravitational search algorithm," *International Journal of Computational Intelligence Systems*, vol. 7, no. 5, pp. 809–826, 2014.

[27] S. Mirjalili and S. Z. M. Hashim, "A new hybrid PSOGSA algorithm for function optimization," in *Computer and information application (ICCIA), 2010 international conference on*. IEEE, 2010, pp. 374–377.

[28] S. Mirjalili, S. Z. M. Hashim, and H. M. Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11 125–11 137, 2012.

[29] S. Mirjalili and A. Lewis, "Adaptive gbest-guided gravitational search algorithm," *Neural Computing and Applications*, vol. 25, no. 7-8, pp. 1569–1584, 2014.

[30] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Filter modeling using gravitational search algorithm," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 117–122, 2011.

[31] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," 2002.

[32] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder, and P. N. Suganthan, "A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization," *Information Sciences*, vol. 209, pp. 16–36, 2012.

[33] J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*. IEEE, 2005, pp. 124–129.

[34] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H. S.-H. Chung, Y.-H. Shi, and J. Zhang, "Genetic learning particle swarm optimization," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–1, 2015.

[35] W.-J. Yu, M. Shen, W.-n. Chen, Z.-h. Zhan, Y.-J. Gong, Y. Lin, O. Liu, and J. Zhang, "Differential evolution with two-level parameter adaptation," *Cybernetics, IEEE Transactions on*, vol. 44, no. 7, pp. 1080–1099, 2014.

[36] J. Liang, B. Qu, P. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization," *Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2014.

[37] F. Saeidi-Khabisi and E. Rashedi, "Fuzzy gravitational search algorithm," *Proceedings 2nd international econference on computer and knowledge engineering*, pp. 156–160, 2012.

[38] A. Sombra, F. Valdez, P. Melin, and O. Castillo, "A new gravitational search algorithm using fuzzy logic to parameter adaptation," in *2013 IEEE Congress on Evolutionary Computation*, June 2013, pp. 1068–1074.

[39] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[40] J. Sun, W. Xu, and W. Fang, "Enhancing global search ability of quantum-behaved particle swarm optimization by maintaining diversity of the swarm," in *International Conference on Rough Sets and Current Trends in Computing*. Springer, 2006, pp. 736–745.

[41] W. Gong, Z. Cai, and C. X. Ling, "DE/BBO: a hybrid

differential evolution with biogeography-based optimization for global numerical optimization," *Soft Computing*, vol. 15, no. 4, pp. 645–665, 2010.

[42] P. Civicioglu, "Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm," *Computers & Geosciences*, vol. 46, pp. 229–247, 2012.

[43] ——, "Backtracking search optimization algorithm for numerical optimization problems," *Applied Mathematics and Computation*, vol. 219, no. 15, pp. 8121–8144, 2013.

[44] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. IEEE, 2009, pp. 210–214.

[45] Y. Wang, H.-X. Li, T. Huang, and L. Li, "Differential evolution based on covariance matrix learning and bimodal distribution parameter setting," *Applied Soft Computing*, vol. 18, pp. 232–247, 2014.

[46] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.

[47] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of global optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[48] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, Feb 2011.

[49] P. Civicioglu, "Artificial cooperative search algorithm for numerical optimization problems," *Information Sciences*, vol. 229, pp. 58–76, 2013.

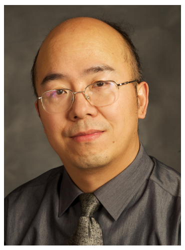
[50] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

Aizhu Zhang received the B.S. and M.Sc. degrees in Geomatics from China University of Petroleum (East China), Qingdao, China, in 2011 and 2014, respectively.

She is currently pursuing the Ph.D. degree with the School of Geosciences, China University of Petroleum (East China), Qingdao, China. Her research interests are in remote sensing image processing, pattern recognition, and intelligence optimization algorithms.

Genyun Sun received the B.S. degree from Wuhan University, Wuhan, China, in 2003 and Ph.D. degree in Institute of Remote Sensing Applications, Chinese Academy of Sciences in 2008.

He is currently an Associate Professor with China University of Petroleum, Qingdao, China. His research interests include remote sensing image processing, hyperspectral remote sensing, high resolution remote sensing, and intelligent optimization algorithm.



Jinchang Ren received his Ph.D. from Electronic Imaging and Media Communication from Bradford University, Bradford, U.K.

He is currently a Senior Lecturer with the Centre for excellence for Signal and Image Processing (CeSIP), also Deputy Director of the Strathclyde Hyperspectral Imaging Centre, University of Strathclyde, Glasgow, U.K. His research interests focus mainly on visual computing and multimedia signal processing, especially on semantic content extraction for video analysis and understanding and more recently hyperspectral imaging.

Xiaodong Li (M'03-SM'07) received his Ph.D. degree in information science from University of Otago, Dunedin, New Zealand.

He is a Professor with the School of Science (Computer Science and Software Engineering), RMIT University, Melbourne, Australia. His research interests include evolutionary computation, neural networks, data analytics, multiobjective optimization, multimodal optimization, and swarm intelligence. He serves as an Associate Editor of the IEEE Transactions on Evolutionary Computation, Swarm Intelligence (Springer), and International Journal of Swarm Intelligence Research. He is a founding member of IEEE CIS Task Force on Swarm Intelligence, a vice-chair of IEEE Task Force on Multi-modal Optimization, and a former chair of IEEE CIS Task Force on Large Scale Global Optimization.

Zhenjie Wang received the B.S. degree from Shandong University of Science and Technology, Taian, China, in 1991, M.Sc. degree from Wuhan University, Wuhan, China, in 2000 and Ph.D. degree in Institute of Geodesy and Geophysics, Chinese Academy of Sciences in 2004.

He is currently a Professor with China University of Petroleum, Qingdao, China. His research interests include GNSS data processing and hydrographic surveying and charting.

Xiuping Jia (M'93-SM'03) received the Ph.D. degree in electrical engineering from The University of New South Wales, Australia, in 1996.

Since 1988, she has been with the School of Information Technology and Electrical Engineering, The University of New South Wales, Canberra Campus, Australia, where she is currently a Senior Lecturer. She is also a Guest Professor with Harbin Engineering University, China, and an Adjunct Researcher at China National Engineering Rsearch Center for Informaiton Technology in Agriculture. She is the co-author of the remote sensing textbook titled Remote Sensing Digital Image Analysis [Springer-Verlag, 3rd (1999) and 4th eds. (2006)]. Her research interests include remote sensing and imaging spectrometry. She is an Editor of the Annals of GIS and an Associate Editor of the IEEE Transactions On Geoscience and Remote Sensing.