Applied Soft Computing 67 (2018) 126–140

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting

Xuewen Xia [a,*], Ling Gui [b], Zhi-Hui Zhan [c,*]

[a] School of Software, East China Jiaotong University, Nanchang 330013, China
[b] School of Economics and Management, East China Jiaotong University, Nanchang 330013, China
[c] Guangdong Provincial Key Lab of Computational Intelligence and Cyberspace Information, School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 23 October 2017
Received in revised form 6 February 2018
Accepted 19 February 2018
Available online 2 March 2018

Keywords:
Particle swarm optimization
Dynamic sub-swarm number
Sub-swarm regrouping
Purposeful detecting
Local-searching

ABSTRACT

This paper proposes a multi-swarm particle swarm optimization (MSPSO) that consists of three novel strategies to balance the exploration and exploitation abilities. The new proposed MSPSO in this work is based on multiple swarms framework cooperating with the dynamic sub-swarm number strategy (DNS), sub-swarm regrouping strategy (SRS), and purposeful detecting strategy (PDS). Firstly, the DNS divides the entire population into many sub-swarms in the early stage and periodically reduces the number of sub-swarms (i.e., increase the size of each sub-swarm) along with the evolutionary process. This is helpful for balancing the exploration ability early and the exploitation ability late, respectively. Secondly, in each DNS period with special number of sub-swarms, the SRS is to regroup these sub-swarms based on the stagnancy information of the global best position. This is helpful for diffusing and sharing the search information among different sub-swarms to enhance the exploitation ability. Thirdly, the PDS is relying on some historical information of the search process to detect whether the population has been trapped into a potential local optimum, so as to help the population jump out of the current local optimum for better exploration ability. The comparisons among MSPSO and other 13 peer algorithms on the CEC2013 test suite and 4 real applications suggest that MSPSO is a very reliable and highly competitive optimization algorithm for solving different types of functions. Furthermore, the extensive experimental results illustrate the effectiveness and efficiency of the three proposed strategies used in MSPSO.

1. Introduction

Particle swarm optimization algorithm (PSO) is a widely known evolutionary algorithm proposed by Kennedy and Eberhart in 1995 [1,2]. During the optimization process, each particle adjusts its flight direction and step-size relying on the information extracted from the past experience of itself and its neighbors. Although the search pattern of each particle is quite simple, the search behavior of the entire population is very complex, and the population shows great intelligence owing to the cooperative behavior among particles. Due to the simplicity of implementation, PSO has been applied for many academic and real-world applications [3–5].

Extensive studies reveal that PSO's performance mainly depends upon its two characteristics [6,7]: exploration and exploitation. However, there is a contradiction between the two capabilities. In order to be successful, PSO needs to establish a good ratio between exploration and exploitation. A common belief is that PSO should start with exploration and then gradually change into exploitation. Hence, many time-varying strategies are proposed to regulate the parameters and neighbor topology involved in PSO.

For example, in the most ubiquitous update rules of parameters introduced in [8,9], three parameters involved in PSO are adjusted based on the iteration numbers aiming to meet different search requirements of different evolutionary stages. The fundament thought of these modifications is tuning particles' learning weights for their exemplars. Moreover, different neighbor topologies display various characteristics because the way and the speed of knowledge diffusion among the population are different. For instance, many studies indicate that PSO with a sparse neighbor topology yields favorable results on complicated multimodal problems, while PSO with a dense neighbor topology offers promising solutions on simple unimodal problems [10,11]. To take advantages of the merits of different neighbor topologies, many dynamic

* Corresponding authors.
E-mail addresses: xwxia@whu.edu.cn (X. Xia), zhanapollo@163.com (Z.-H. Zhan).

https://doi.org/10.1016/j.asoc.2018.02.042

neighbor topologies are proposed in accordance with particles' search performance [12–14]. The main motivations of the dynamic topologies are adjusting particles' neighbors, and then changing the speed of information diffusion among the population. In addition, to layout a more satisfactory performance for PSO, various adaptive methods utilizing the feedback of evolution processes are introduced to control the parameters [15,16] and adjust the neighbor topologies [17]. Furthermore, directing an evolution process toward exploration or exploitation is also possible by population resizing [18].

Inspired by the idea that information from very different types of trade-offs could be combined to yield other kinds of good trade-offs, we propose a novel multi-swarm PSO (MSPSO) in this paper. In the new proposed MSPSO, three novel strategies are introduced to balance the exploration and exploitation. The first strategy, named as dynamic sub-swarm number strategy (DNS), divides the entire population into many small-sized sub-swarms in the early stage and periodically reduces the number of sub-swarms (i.e., increase the size of each sub-swarm) along with the evolutionary process. In this case, more sub-swarms with smaller size in the early evolutionary stage urge the search toward more exploration. On the contrary, less sub-swarms with greater size in the later evolutionary stage push the search toward more exploitation. The second strategy is sub-swarm regrouping strategy (SRS), based on which the sub-swarms are regrouped in each DNS period based on the stagnancy information of the global best position. After the regroup operator, much superior information acquired by each sub-swarm can be shared by other sub-swarms, which is beneficial for exploitation. The last strategy is purposeful detecting strategy (PDS), in which much historical information of the search process is applied to help the population jump out of the current local optimum for better exploration ability.

The rest of this paper is organized as follows. Section 2 presents the framework of the canonical PSO and reviews some enhanced PSOs. The details of MSPSO are described in Section 3. The experimental verifications and comparisons using the CEC2013 test suite are presented in Section 4. Furthermore, the discussion of the results is also included in this section. Finally, conclusions are provided in Section 5.

2. Related works

2.1. Canonical PSO

In PSO, each particle i is associated with two vectors, i.e., a position vector $\mathbf{X}_i = [x_{i,a}, x_{i,b}, \ldots, x_{i,D}]$ and a velocity vector $\mathbf{V}_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,D}]$, where D represents the dimensions of a problem under study. The vector $\mathbf{X}_i$ is regarded as a candidate solution to the problem while $\mathbf{V}_i$ is treated as a searching direction and step size of the ith particle. During the evolutionary process, each particle adjusts its flight trajectory based on two vectors, named as personal historical best position $\mathbf{Pb}_i = [pb_{i,1}, pb_{i,1}, \ldots, pb_{i,D}]$ and the neighbor's best-so-far position $\mathbf{Nb}_i = [nb_{i,1}, nb_{i,1}, \ldots, nb_{i,D}]$, respectively. The update rules of $\mathbf{V}_i$ and $\mathbf{X}_i$ are defined as (1) and (2), respectively.

$$v_{i,j}(t+1) = w \cdot v_{i,j}(t) + c_1 \cdot r_{1,j} \cdot (pb_{i,j} - x_{i,j}(t)) + c_2 \cdot r_{2,j} \cdot (nb_{i,j} - x_{i,j}(t)) \quad (1)$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \quad (2)$$

where w represents an inertia weight determining how much the previous velocity is preserved; c_1 and c_2 are two acceleration coefficients deciding the relative learning weight for $\mathbf{Pb}_i$ and $\mathbf{Nb}_i$, respectively; $r_{1,j}$ and $r_{2,j}$ are two random numbers uniformly distributed in the interval $[0, 1]$; $x_{i,j}(t)$ and $v_{i,j}(t)$ represent the position and velocity on the jth dimension of the ith particle at generation t, respectively.

2.2. Variants of PSO

PSO has been popularly studied since it is proposed due to its effectiveness and simplicity, and numerous PSO variants have been developed. According to the different objectives to be dealt with, these improvements can generally be categorized into three cases, including parameter adjustments, topology adjustments, and hybridization strategies.

2.2.1. Parameter adjustments

Given that a larger w facilitates the exploration while a smaller one is beneficial for the exploitation, it seems natural to adopt a time-varying w to offset the contradiction between them. The most ubiquitous update rule of w, introduced by Shi and Eberhart [8] in 1998, is linearly decreasing from 0.9 to 0.4 over the optimization process, which is still applied in many PSOs now. Furthermore, motivated by the iteration-based w, Ratnaweera et al. [9] further advocated a PSO with time-varying acceleration coefficients (HPSO-TVAC). However, considering that the search process of PSO is nonlinear and complicated, many nonlinearly-varying strategies [19,20] introduced to tune the parameters.

Although these parameters adjustments offer relatively reliable performances, the common feature of them, i.e., iteration-based strategy, does not take enough advantages of evolutionary information of individuals. Thus, to take full use of the historical information of the evolutionary and layout a more satisfactory tuning method for the parameters, different adaptive strategies are proposed in recent years [15,16,21,22]. For example, Zhan et al. [15] proposed an adaptive PSO (APSO), in which w, c_1 and c_2 are relied on the evolutionary state estimation (ESE), which relies on the distribution of population and the fitness of particles rather than the iteration number. In [21], a new adaptive w based on Bayesian techniques is used to enhance PSO's exploitation capability. The common feature in these improvements is that particles regulate their own parameters according to their own characteristics or/and population's performance, and then adjust their search behaviors. The adaptive strategies cause different particles to play different roles on exploration and exploitation during the search process, and then to achieve a balance between exploration and exploitation.

2.2.2. Topology adjustments

Global version PSO (GPSO) and local version PSO (LPSO) are two common neighbor topologies. Some researches manifest that LPSO is conductive to exploration while GPSO is beneficial for exploitation [10]. Moreover, many different neighbor topologies are introduced to give particles more diversified learning models [23,24], and then improve the comprehensive performance of PSO. For instance, Zhan introduces the orthogonal test, which is a black box testing technique that has been successfully applied in [25], into PSO to help a particle construct a more favorable and efficient guidance exemplar, and then to achieve a reliable performance [26].

However, it is unrealistic to fix a proper neighbor topology for a specific problem beforehand since many real applications are black-box problems as well as the optimization process is a dynamic course. Thus, many dynamic topologies are proposed to balance the exploration and exploitation during the evolution process [27–30]. The main motivation of these type improvements is that different topologies are allocated to different particles or different evolutionary stages according to various characteristics of the particles and population, typically fitness and diversity. The heterogeneous topologies of individuals enable them to cope with

different fitness landscape resulting in a better balance between exploration and exploitation.

Furthermore, multi-swarm mechanisms also capture many researchers' attention during the last decade [20,27,31–35]. During the sub-swarms' independent evolution period, a particle only exchanges information with those particles who belong the same sub-swarm as itself, which is beneficial for exploration. In addition, to improve convergence speed and achieve a balance between exploration and exploitation, different PSOs introduce different exchange strategies among different sub-swarms. For instance, in [31], a food chain model is adopted to control the information exchange among different sub-swarms. While the fitness difference between two species is larger, much more information is exchanged between the species. In [34], the entire population is divided into an exploration-subpopulation and an exploitation-subpopulation. During the evolutionary process, the exploration-subpopulation does not take advantage of information from the exploitation-subpopulation aiming to retain the diversity. On the contrary, the exploitation-subpopulation extracts much useful information from the exploration-subpopulation to enhance its exploitation.

Although it has been testified by many researches that the multi-swarm mechanism is an efficient method for keeping population diversity, a static multi-swarm strategy may harmful for convergence speed [31,32] while a dynamical multi-swarm mechanism may introduce some new parameters and operators, which cause PSO more complicated [31] and time-consuming [27,34]. Thus, proposing a simple PSO variant with a higher efficiency is a challenging work in PSO community.

2.2.3. Hybridization strategies

In recent years, it has captured many researchers' attention that how to take full advantages of different algorithms and strategies through a proper hybrid mechanism. For instance, genetic operators [21,22,36,37] are very popular auxiliaries for balancing the exploration and exploitation. Generally, the selection and crossover operators are adopted to enhance the exploitation capability while the mutation operator injects diverse information into the population, and then enhance the exploration ability.

Another class of hybridization method is using some searching strategies [13,38–40] to improve the population diversity or/and speed up the convergence rate. For example, the Levy flight and detecting operator are adopted in [39–42] respectively to help the population get rid of local minimal and improve global search capability.

In addition, hybridizing PSO with other evolutionary algorithms (EAs) is another popular strategy in recent years. For instance, different hybrid algorithms based on PSO have been proposed in the last decade together with differential evolution (DE) [43,44], artificial bee colony (ABC) [45,46], and estimate distribute algorithm (EDA) [47] have been proposed in the last decade. No matter which hybrid mechanism is adopted in these PSO variants, the main idea is using different search behaviors of the cooperated algorithms to improve exploration capability as well as sharing helpful information of the algorithms to enhance exploitation capability.

3. MSPSO

The topology of population plays an important role in balancing the exploration and exploitation since it determines the speed of knowledge dissemination among the population. Inspired by the aforementioned researches of PSO, a novel MSPSO is proposed in this research. In MSPSO, the entire population is divided into many equal small-sized sub-swarms at the initial evolutionary stage. Each sub-swarm uses its own members to search for better area in parallel according to Eqs. (1) and (2). The LPSO topological structure is applied in each sub-swarm. This mechanism urges the search toward more exploration. Along with the evolutionary process, the DNS is carried out, by which the number of sub-swarms is decrease aiming to meet the exploitation requirement. Furthermore, to share each sub-swarm's helpful information, the simple strategy named SRS is conducted during the search process aiming to further improve exploitation. While the population has been trapped into a potential local optimum, the PDS is carried out to help the population jump out of the current local optimum for better exploration ability. The details of the new introduced strategies and the framework of MSPSO are presented as follows.

3.1. Dynamic sub-swarm number strategy (DNS)

At the beginning of evolutionary process, the entire population is divided into many smaller sub-swarms, typically each sub-swarm only contains two individuals. The number of the sub-swarms is reduced accordingly while the size of each sub-swarm is gradually increased along with the evolutionary process. At the final stage of evolutionary, the number of the sub-swarms is reduced to one, which means that all the sub-swarms are merged into one population.

Based on the DNS, many small-sized sub-swarms are conductive to keeping population diversity at the initial evolutionary stage since the information slowly diffuses within a smaller range, as well as many parallel evolved sub-swarms are beneficial for exploration. On the contrary, the gradually decreased number of sub-swarms makes the knowledge flow become more quickly and wider, which is beneficial for exploitation at the later evolutionary stage. Thus, the DNS pushes the search from exploration toward more exploitation along with the evolution process.

Note that, to rationally utilize the merits of DNS, there are two issues need to be carefully considered. One issue is *how to determine the number of sub-swarms*, and another issue is *when to adjust the number of sub-swarms*.

For the first issue, we define an ordered set of integers $\mathbb{N} = \{n_1, n_2, \ldots, n_{k-1}, n_k\}$ where $n_1 > n_2 > \ldots > n_{k-1} > n_k$. Each element in $\mathbb{N}$ denotes a candidate number of sub-swarms. In this research, to facilitate program, all the sub-swarms have the same size in each generation which means that the number of sub-swarms must be a factor of the population size. For example, if the population size is N=30, the number of sub-swarms (N_{sub}) is selected from $\mathbb{N} = \{15, 10, 6, 5, c, b, 1\}$, each element in which is a factor of N. Thus, the size of each sub-swarm (S_{sub}) at the initial stage is $S_{sub} = N/N_{sub} = 2$. In the final stage of evolution, N_{sub} and S_{sub} are 1 and 30 respectively, which means that all the sub-swarms have been amalgamated into single one swarm.

For the second issue, we adjust the number of sub-swarms in each C_{gen} fitness evaluations (FEs) (with $C_{gen} = MaxGens/\|\mathbb{N}\|$), where$\|\mathbb{N}\|$ is the number of elements in $\mathbb{N}$, C_{gen} denotes the adjustment period, $MaxGens$ is the predefined maximum generations of the evolution. In this case, the entire population at different evolutionary periods (i.e., in different numbers of sub-swarms cases) is assigned to the same generations except the final stage. The motivation of this scheme is to keep a smooth adjusting process for the number of sub-swarms. However, we believe our methods for the two issues may not be optimal choices since they do not consider different characteristics in different problems. Given it is a profoundly difficult task that precise extracting useful characteristics for a black-box problem during the optimizing process, we will study the issues in our future works.

Furthermore, we regard that the global best solution **GBEST** obtained by the entire population carries out a local-searching operator periodically can help MSPSO achieve a more accurate solution. In this research, **GBEST** is refined by BFGS Quasi-Newton method, which has been applied in many PSO variants [13,14], after

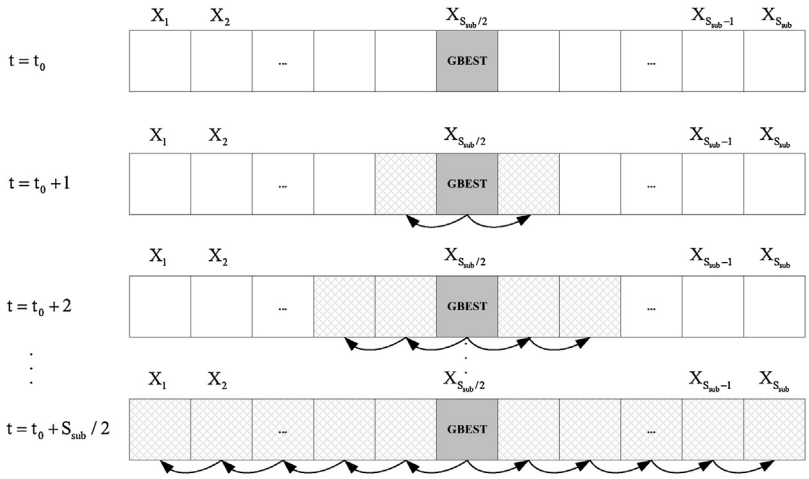

Fig. 1. The process of information diffusion among ring topological structure.

the number of sub-swarms has been adjusted. For each time that **GBEST** performs the local-searching operator, we assign [0.10 * *fes*] fitness evaluations to the operator, where *fes* is the number of fitness evaluations that the population has consumed. In this case, more fitness evaluations will be assigned to the local-searching operator in the later evolution stage since we regard that strengthening the local-searching ability is helpful for improving solutions accuracy if the population has found a promising region.

The details of DNS can be viewed in Algorithm 1, in which $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, and $\mathbf{Pb}_i^k$ are the position, velocity, and personal historical best solution of the *i*th particle in the *k*th sub-swarm, respectively.

Algorithm 1. DNS()

Input: *fes*, *t*, C_{gen}, *N*, N_{sub}, S_{sub}, *m*, $\mathbb{N}$, $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, $\mathbf{Pb}_i^k$ ($1 \le k \le N_{sub}$, $1 \le i \le S_{sub}$), **GBEST**;
01: **If** mod(*t*, C_{gen})==0 and $m < ||\mathbb{N}||$
02: $m = m+1$;
03: $N_{sub} = \{n_m | n_m \in \mathbb{N}\}$; $S_{sub} = N/N_{sub}$;
04: Random divide the whole population into N_{sub} sub-swarms, including $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, and $\mathbf{Pb}_i^k$;
05: Assign [0.1**fes*] fitness evaluations to **GBEST** to carry out BFGS Quasi-Newton method;
06: *fes*=*fes*+0.1**fes*;
07: **End If**
Output: *fes*, N_{sub}, S_{sub}, *m*, $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, $\mathbf{Pb}_i^k$ ($1 \le k \le N_{sub}$, $1 \le i \le S_{sub}$), **GBEST**.

3.2. Sub-swarms regrouping strategy (SRS)

In dynamic multi-swarm PSO (DMSPSO) [13,14], a randomized regrouping schedule endows particles with a mutable neighborhood structure. Moreover, the population of DMSPSO is randomly regrouped after every *R* generations, which is called regrouping period, and then restarts the search process in the new configuration of sub-swarms. However, it is very hard for us to choose a fixed *R* since different problems have their own various characteristics. For example, a smaller *R* may frequently disturb the correct searching direction of the population, on the contrary, a greater *R* cannot enable the population to be timely regrouped when it has stagnated.

In MSPSO, the consecutive generations-stagnancy of **GBEST**, denoted by $Stag_{best}$, is selected as a sub-swarms regrouping criteria in MSPSO. In this case, the entire population can be timely regrouped into N_{sub} sub-swarms while **GBEST** has stagnated more than a threshold, without having to wait for a predefined regrouping period. Since the neighbor topology of each sub-swarm is ring model, each particle, including the best one, in a larger sub-swarm may need more generations to thoroughly extract helpful information from other particles. In other words, the degree of information diffusion depends on the size of a sub-swarm. Thus, we set $Stag_{best} = \lfloor S_{sub}/2 \rfloor$ as the threshold. For example, if the number of particles in each sub-swarm is 10, the population will be regrouped if $Stag_{best}$ is more than 5. The process of information diffusion among the ring topological structure can be described in Fig. 1.

Based on the above mentioned, the details of SRS can be described as Algorithm 2.

Algorithm 2. SRS()

Input: $Stag_{best}$, N_{sub}, S_{sub}, $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, $\mathbf{Pb}_i^k$ ($1 \le k \le N_{sub}$, $1 \le i \le S_{sub}$);
01: **If** $Stag_{best} \ge S_{sub}/2$
02: Random regroup the whole population into N_{sub} sub-swarms;
03: $Stag_{best} = 0$;
04: **End If**
Output: $Stag_{best}$, $\mathbf{X}_i^k$, $\mathbf{V}_i^k$, $\mathbf{Pb}_i^k$ ($1 \le k \le N_{sub}$, $1 \le i \le S_{sub}$).

3.3. Purposeful detecting strategy (PDS)

Although DNS and SRS are included as techniques to balance the exploration–exploitation behavior, it is inevitable that the population may fall into a local optimum when optimizing complicated multimodal problems. Although the mutation strategy [37] applied in individuals increases the population diversity to some extent, the random disturbance cannot help **GBEST** purposefully explore promising regions. In recent years, extracting much useful knowledge from historical experience displays very favorable performance in different researches [48,49]. Thus, in this research, some historical information of individuals is applied to guide **GBEST** to carry out a purposeful detecting operator which has been applied in [41,42], and then to help the population jump out of the local optimum.

The first issue need to be considered in PDS is *how to select some useful information*. To easily collect information and carry out the detecting operator, each dimensional search space of a problem is divided into many small-sized segments, which can be described in Fig. 2.

In Fig. 2, lb^i and ub^i are the lower and upper boundaries of the *i*th dimension. Then the whole search space of the *i*th dimension $\mathbf{S}^i$ is divided into *Rn* equal segments, denoted as $s_1^i, s_2^i, \ldots, s_{Rn}^i$.

After the segmentation, the times that elites falling into a specific segment is an index to evaluate the segment. Specifically, more times that $\mathbf{Pb}_i$ falls into a segment indicates the segment has a

higher merit value. Thus, the merit value of each segment s_j^i is described as (3).

$$\mathcal{M}_j^i = \mathcal{M}_j^i + 1, \quad \text{if} \quad pb_{i,k} \quad \text{lies within} \quad s_j^i \tag{3}$$

The second issue is *when* **GBEST** *carries out the detecting operator*. Considering that the population may display different statistic characteristics for the distinct fitness landscape in different evolution stages, a periodic detecting operator is carried out in this work. Specifically, $\mathcal{M}_j^i$ $(1 \le j \le Rn)$ of the ith dimension is used to help **GBEST** find out a promising position in each S_{sub} generations. In this case, a smaller S_{sub} causes **GBEST** to carry out the detecting operator more frequently in the early stage in favorable for exploration. On the contrast, a larger S_{sub} enables the detecting operator to be conducted infrequently in the later evolution stage since the main task at the stage is exploitation.

The last issue is *how to carry out the detecting operator based on the merit values*. In this work, for a fixed value of i, Rn segments are divided into *superior sub-regions*, *inferior sub-regions*, and *moderate sub-regions*, which have the largest, the smallest, and the medium values of $\mathcal{M}_j^i$, respectively. When $Gbest_i$ denoting the ith dimension value of the **GBEST** lies within a superior sub-region of the ith dimension, **GBEST** will detect an inferior sub-region on ith dimension. And $Gbest_i$ will be replaced by the detected position only if the new position improves the performance of the original **GBEST**. In other two cases, i.e., $Gbest_i$ falling into an inferior sub-region or a moderate sub-region, **GBEST** will not carry out the detecting operator since we believe other particles can help it fly toward a promising region.

In order to avoid $Gbest_i$ detecting a same segment in different periods, which is harmful for exploration, a tabu strategy is applied in the detecting operator. For instance, if s_j^i has been detected by $Gbest_i$, a tabu flag $tabu_j^i$ for the segment is set to '1'. Under this condition, $Gbest_i$ does not detect s_j^i until $tabu_j^i$ is reset to '0'. While $Gbest_i$ has detected all segments, i.e., all the tabu flags $tabu_j^i$ are '1', these flags will be reset to '0', and then a new statistical information is recorded.

According to the aforementioned introduction, the PDS of **GBEST** is detailed as Algorithm 3.

Algorithm 3. PDS()

Input: fes, **GBEST**, $\mathcal{M}_j^i$, $tabu_j^i$ $(1 \le i \le D, 1 \le j \le Rn)$;
TmpGB = **GBEST**;
For i = 1 to D
  If $tmpgb_i \in \{s_j^i | \mathcal{M}_j^i$ is greater or equal to other $\mathcal{M}_k^i (1 \le k \le Rn)\}$
    /*$tmpgb_i$ is the ith value of **TmpGB***/
    $tmpgb_i$ is replaced by a random value within an inferior segment s_k^i where $tabu_k^i = 0$;
    Evaluate **TmpGB**; fes=fes+1;
    If TmpGB is better than **GBEST**
      GBEST=TmpGB;
    End If
    $tabu_k^i = 1$;
    If all $tabu_k^i$ $(1 \le k \le Rn)$ are equal to 1
      Set each $tabu_k^i$ to 0;
    End If
  End If
End For i
Output: fes, **GBEST**, $tabu_j^i$ $(1 \le i \le D, 1 \le j \le Rn)$.

3.4. Framework of MSPSO

By incorporating the aforementioned components into a multi-swarm PSO framework, the details of MSPSO can be described as Algorithm 4. Note that, although there are many new parameters in MSPSO, including C_{fes}, N_{sub}, S_{sub}, and Rn, the majority of the new introduced parameters except Rn depend on the entire population

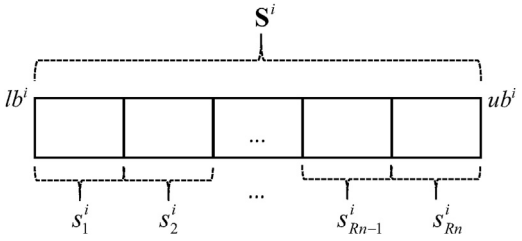

Fig. 2. Segmentation of the ith dimension.

size. Thus, from this perspective, we regard the new introduced strategies do not greatly raise the complexity of the canonical PSO. In addition, the adaptive multi-swarm mechanism provides a more comprehensive cooperating mechanism for all sub-swarms, and then helps MSPSO take full advantages of each sub-swarm.

Algorithm 4. MSPSO ()

Initialization: D, N, $MaxFEs$, $MaxGens$, fes=0, t=0, Rn=10, $Stag_{best}$ = 0;
$\mathbb{N} = \{n_1, n_2, \ldots, n_p\}$, m=1, $C_{fes} = MaxGens/|\mathbb{N}|$;
Divide each dimension search space into Rn same sized sub-regions;
$N_{sub} = n_m$, $S_{sub} = N/N_{sub}$, $\mathcal{M}_j^i = 0$, $tabu_k^i = 0$ $(1 \le i \le N, 1 \le j \le D, 1 \le k \le Rn)$;
Randomly initialize $\mathbf{X}_{ss}^{ns}$, $\mathbf{V}_{ss}^{ns}$, $\mathbf{Pb}_{ss}^{ns}$ $(1 \le ss \le S_{sub}, 1 \le ns \le N_{sub})$;
While not meet stop conditions
  t=t+1;
  For k= 1 to ns in parallel /* ns denotes the number of sub-swarms */
    For i= 1 to ss in parallel /* ss denotes the size of each sub-swarm */
      Update $\mathbf{V}_i^k$ and $\mathbf{X}_i^k$ according to (1) and (2), respectively;
      Update $\mathbf{Pb}_i^k$;
    End For i
  End For k
  Update MER according to (3);
  Update **GBEST** and $Stag_{best}$;
  DNS();
  SRS();
  PDS();
End While

4. Experimental verification and comparisons

4.1. Experimental setup

4.1.1. Benchmark functions

To verify how MSPSO performs in different environments, we carry out different sets of experiments on CEC2013 test suite. According to characteristics, 28 benchmark functions in CEC2013 test suite are divided into three groups:

- 5 unimodal functions (f_1–f_5);
- 15 basic multimodal functions (f_6–f_{20}); and
- 8 composition functions (f_{21}–f_{28});

In this study, acceptable errors are all set as 100 for all the test problems except f_1, f_3, and f_5. For the two easy problems, i.e., f_1 and f_5, a smaller acceptable error (10^{-6}) is selected while a larger acceptable error (10^{+7}) is chosen for the difficult problem f_3. It should be noted that the acceptable errors are not in favor of any peer algorithm. More detailed information of the functions can refer to the literature [50].

4.1.2. Peer algorithms

For a comparative analysis, experiments are conducted to compare MSPSO with 9 PSO variants proposed in the last decade and 4 outstanding EAs, including JADE, SaDE, CoDE, and CMA-ES. In the experiments, the dimensionality of the test problems and the maximum number of function evaluations ($MaxFEs$) are set to be

Table 1
Parameters settings of the 14 peer algorithms.

Algorithm	Parameters settings
MSPSO	$\chi = 0.7298$, $c_1 = c_2 = 1.49445$, $Rn = 10$, $N = 30$
DMSPSO [14]	$\chi = 0.7298$, $c_1 = c_2 = 1.49445$, $R = 10$, $L = 100$, $L\_FEs = 200$
[a]F-PSO [12]	$w = 0.9–0.4$, $Tmax = 3 * N^2$, $K = 4 * N$
OLPSO [26]	$w = 0.9–0.4$, $c = 2.0$, $G = 5$
SLPSO [28]	$w = 0.9 - 0.5 * gen/MaxGen$, $\eta = 1.496$, $\gamma = 0.01$
PSODDS [51]	$\chi = 0.7298$, $c_1 = c_2 = 2.05$
SL_PSO [30]	$M = 100$, $\alpha = 0.5$, $\beta = 0.01$
HCLPSO [34]	$w = 0.9–0.4$, $c = 1.49445$
SSS-APSO [20]	$w_{max} = 0.7$, $w_{min} = 0.2$, $\Delta w = 0.1$, $\phi_1 = \phi_1 = 2.0$, $p = 4$
SopPSO [41]	$w = 0.9–0.4$, $c_1 = 2.5–0.5$, $c_1 = 0.5–2.5$, $Rn = 10$, $MaxStag_{ind} = 13$, $MaxStag_{best} = 5$, $Cycle = 3$
JADE [52]	"current-to-*p*best/1", $\mu_{CR} = 0.5$, $\mu_F = 0.5$
SaDE [53]	"rand/1/bin", "rand-to-best/2/bin", "rand/2/bin", "current-to-rand/1" [$F \in N(0.5, 0.3)$, $C_r \in N(CR_m, 0.1)$]
CoDE [54]	"rand/1/bin", "rand/2/bin", "current-to-rand/1" [$F = 1.0$, $C_r = 0.1$], [$F = 1.0$, $C_r = 0.9$], [$F = 0.8$, $C_r = 0.2$]
[b]CMA-ES [55]	–

[a] To describe simply, Frankenstein's PSO is renamed as F-PSO in this research.
[b] There are too many parameters in CMA-ES. To describe simply, the parameters' settings can refer to the literature [55].

30 and 300,000, respectively. The parameters settings of all peer algorithms, which are the same as the corresponding papers, are listed in Table 1. All the experiments are conducted on the following system:

- OS: Windows 7
- CPU: Intel Core i5-4200, 2.30 GHz
- RAM: 4 GB
- Language: Matlab R2014a

4.1.3. Performance metrics

Results of 100 independent runs on the test functions are collected for statistical analysis. In this study, mean value (*Mean*), standard deviation (*Std.Dev.*), median value (*Median*), success ratio (*SR*), and success performance (*SP*) are 5 performance metrics for each algorithm. The *SR* is defined as (4).

$$SR = \text{No. of successful runs/total \# of runs.} \tag{4}$$

where a successful run is that an algorithm achieves the acceptable error within the maximum number of function evaluations.

The *SP* is the number of function evaluations (*fes*) for an algorithm to reach the acceptable error. As an evaluation index for convergence speed, the mean of *SP* defined in [56] is detailed as (5).

$$\text{mean}(SP) = ((1 - SR)/SR) * MaxFEs + \text{mean}\ (fes\ \text{for successful runs}) \tag{5}$$

Furthermore, to compare the performance of two algorithms at the statistical level, the Friedman-test and two-tailed *t*-test of freedom at a 0.05 level of significance are conducted between MSPSO and other peer algorithms.

4.2. Experimental results

In the experiments, the population sizes of all the peer algorithms for 30D problems are the same as that applied in the corresponding lectures. For all the test functions, each algorithm carries out 100 independent runs on them. Among the results listed in Tables 2–4, the best results of *Mean*, *Median*, and *SP* on each problem among all algorithms are shown in bold.

4.2.1. Unimodal functions

In the first set of experiments, 5 unimodal functions are studied. The comparison results are summarized in Table 2. For f_1 and f_5, all the algorithms show very promising performance, and there are 3 algorithms besides MSPSO achieve the global optima result on f_1 on the 100 runs. Furthermore, MSPSO also offers the best result on f_2, in terms of *Mean* value. However, it yields adverse result on f_3. On the contrary, CMA-ES attains significantly favorable performance on f_3 and f_4. When considering the *Median* results, DMSPSO manifests the best results since it offers the most accurate solutions on 3 out of the 5 unimodal functions which is slightly better than CMA-ES and SaDE.

There are only 4 algorithms, i.e., MSPSO, DMSPSO, CMA-ES, and SopPSO, satisfy the acceptable error at least one time in the independent 100 runs for each unimodal function. The values of *SR* demonstrate that CMS-ES has the best reliable performance on this class problems, followed by MSPSO, DEMSPSO, and SopPSO.

Although MSPSO shares a same characteristic with DMSPSO, i.e., multi-swarm structure, MSPSO performs weaker than DMSPSO on f_1 and f_5. The reason is that the larger population and the detecting operator in MSPSO waste too many evaluations since there is no local optima in the two simple problems. On the contrary, MSPSO offers more favorable performance than DMSPSO on f_2, f_3, and f_4, in terms of *Mean*(*SP*), since the 3 benchmark problems have a common feature, i.e., smooth local irregularities. The comparison results verify the effectiveness of the detecting operator on this type fitness landscape.

4.2.2. Multimodal functions

From the comparison results summarized in Table 3 among the 15 multimodal problems, it can be observed that MSPSO and JADE achieve the most pleasurable result since they both achieve the best results on 6 out of the 15 multimodal functions contributing to the *Mean* values, followed by SL_PSO, SLPSO, and DMSPSO.

Moreover, the *Mean*(*SP*) results manifest that MSPSO, CoDE, and JADE have very reliable performance on the multimodal functions since the 3 algorithms offer the promising results on all the multimodal functions except f_{15}, on which there is no algorithm reaches the acceptable error at least one run. For f_8, f_9, f_{16}, and f_{20}, the *Mean*(*SP*) values of most algorithms are less than 300. Thus, we regard majority of the algorithms have almost the same performance on these multimodal problems, in terms of *Mean*(*SP*). On the other problems, DMSPSO, MSPSO, SLPSO, and JADE offer more outstanding comprehensive performance.

4.2.3. Composition functions

The comparison results on the 8 composition functions are summarized in Table 4, from which we can see that MSPSO yields the most pleasurable performance on 3 out of the 8 composition functions, followed by DMSPSO which achieves the most favorable results on 2 composition problems. Thus, we can draw a conservative conclusion that the sub-swarm mechanism applied in MSPSO and DMSPSO has a positive effect on the composition problems. Comparing with the results between the unimodal functions and the multimodal functions, we find out that CMA-ES offers poor performance on the composition functions, in terms of *Mean* values. On the contrary, MSPSO manifests more reliable characteristics on this type functions. For f_{22}, f_{24}, f_{25}, f_{26}, and f_{28}, there is no significant difference between MSPSO and other outstanding algorithms, in terms of *Median* values. However, OLPSO, PSODDS, and MSPSO offer very favorable performance on f_{21}, f_{23}, and f_{27} respectively, contributing to *Median* value. Moreover, MSPSO attains the most promising results, in terms of *SR* and *Mean*(*SP*), since it is the only one that reaches the acceptable errors at least one run on the 3 composition problems.

Table 2
Comparison results on the 5 unimodal functions (f_1–f_5).

		DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
f_1	*Mean*	**0.00E+00**	1.48E−13	8.60E−09	5.23E−13	3.57E+01	1.11E−13	3.27E−13	5.91E−13	4.73E−13	**0.00E+00**	**0.00E+00**	1.11E−13	4.27E−13	**0.00E+00**
	Std.Dev	0.00E+00	1.09E−13	1.12E−09	1.43E−13	2.08E+02	1.14E−13	1.13E−13	1.39E−13	7.53E−14	0.00E+00	0.00E+00	1.14E−13	1.78E−13	0.00E+00
	Median	**0.00E+00**	2.27E−13	8.93E−09	4.55E−13	9.43E−09	0.00E+00	2.27E−13	6.82E−13	4.55E−13	**0.00E+00**	**0.00E+00**	0.00E+00	4.55E−13	**0.00E+00**
	SR	100	100	100	100	97	100	100	100	100	100	100	100	100	100
	Mean(SP)	**2581**	28,666	164,230	91,684	32,605	22,145	96,925	33,142	3592	14,510	24,450	90,228	6859	25,028
f_2	*Mean*	9.47E−02	3.58E+06	4.10E+07	3.01E+06	1.10E+06	5.45E+05	1.36E+06	2.06E+06	9.68E−02	7.39E+03	3.91E+05	1.07E+05	4.43E−03	**2.20E−03**
	Std.Dev	3.35E−01	2.22E+06	1.46E+07	2.09E+06	2.83E+06	2.73E+05	7.10E+05	2.00E+06	9.86E−02	5.96E+03	1.97E+05	5.29E+04	1.56E−13	4.78E−03
	Median	1.99E−03	3.36E+06	4.01E+07	2.23E+06	3.16E+05	4.57E+05	1.27E+06	1.30E+06	3.17E−02	6.05E+03	3.44E+05	1.00E+05	**4.56E−13**	9.20E−04
	SR	100	0	0	0	0	0	0	0	100	0	0	0	100	100
	Mean(SP)	296,250	–	–	–	–	–	–	–	167,400	–	–	–	**50,694**	80,618
f_3	*Mean*	9.53E+06	3.58E+06	3.32E+10	2.51E+09	4.76E+09	2.17E+07	3.61E+07	3.13E+09	1.07E+08	3.91E+05	2.18E+07	2.39E+06	**3.56E+02**	1.21E+07
	Std.Dev	3.35E−01	1.40E+09	1.48E+10	2.61E+09	5.80E+09	2.14E+07	4.54E+07	2.95E+09	1.09E+08	1.75E+06	3.25E+07	7.28E+06	2.54E+03	2.03E+07
	Median	**1.99E−03**	1.18E+07	3.12E+10	1.77E+09	2.86E+09	1.62E+07	2.31E+07	2.02E+09	4.46E+07	2.24E−01	7.39E+06	2.49E+04	**2.25E−02**	5.68E+06
	SR	68	47	0	1	3	38	28	0	16	99	56	93	100	69
	Mean(SP)	124,797	299,070	–	577,659	371,740	281,983	302,795	–	1,373,400	48,825	244,560	135,240	**28,039**	208,831
f_4	*Mean*	9.26E−01	2.31E+03	1.31E+05	4.73E+04	3.20E+03	5.95E+03	2.45E+03	2.46E+02	1.33E−02	8.01E+03	2.70E+03	1.13E+00	**4.07E−13**	1.79E−02
	Std.Dev	4.90E+00	8.17E+02	1.99E+04	1.07E+04	2.76E+03	2.13E+03	1.05E+03	8.08E+01	8.85E−03	1.49E+04	1.43E+03	3.92E+00	1.49E−13	3.20E−02
	Median	1.56E−02	2.18E+03	1.31E+05	4.59E+04	2.42E+03	5.74E+03	2.22E+03	1.89E+02	8.35E−03	3.04E−07	2.50E+03	2.63E−01	**4.55E−13**	8.40E−03
	SR	100	0	0	0	0	0	0	0	100	77	0	100	100	100
	Mean(SP)	162,754	–	–	–	–	–	–	–	30,882	156,093	–	250,626	61,683	**26,372**
f_5	*Mean*	7.96E−15	1.14E−13	9.32E−09	4.71E−11	3.54E+01	1.15E−13	4.07E−13	1.45E+02	1.60E−12	1.03E−13	**0.00E+00**	7.50E−14	8.19E−13	9.21E−14
	Std.Dev	2.92E−14	1.62E−14	6.51E−10	3.59E−10	5.55E+01	1.14E−14	9.72E−14	2.64E+02	9.71E−13	3.27E−14	0.00E+00	6.90E−14	1.67E−12	4.48E−14
	Median	**0.00E+00**	1.14E−13	9.51E−09	8.36E−12	5.46E−08	1.14E−13	3.98E−13	3.12E+01	1.14E−12	1.14E−13	**0.00E+00**	1.14E−13	4.55E−13	1.14E−13
	SR	100	100	100	100	64	100	100	57	100	100	100	100	100	100
	Mean(SP)	35,975	34,606	211,018	121,800	277,375	30,020	111,030	317,347	105,390	**21,195**	43,615	118,362	91,977	56,343

Table 3
Comparison results on the 15 multimodal functions (f_6–f_{20}).

		DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
f_6	*Mean*	1.59E−01	1.98E+01	4.76E+01	2.71E+01	5.91E+01	1.64E+01	1.62E+01	8.34E+01	4.47E−10	1.85E+00	2.95E+01	4.73E+00	1.62E+00	**7.29E−11**
	Std.Dev	7.85E−01	1.89E+01	6.53E+00	2.12E+01	3.15E+01	7.65E+00	2.96E+00	3.51E+01	6.59E−10	6.77E+00	2.76E+01	7.37E+00	6.31E+00	9.06E−11
	Median	4.48E−11	1.38E+01	4.87E+01	1.64E+01	5.45E+01	1.41E+01	1.57E+01	8.05E+01	4.68E−11	**1.14E−13**	1.52E+01	2.26E+00	2.27E−11	5.53E−11
	SR	100	100	100	100	90	100	100	93	100	100	100	100	100	100
	Mean(SP)	**3692**	13,372	71,426	15,752	41,760	5397	29,393	34,244	3671	4400	11,545	25,716	9215	5544
f_7	*Mean*	1.47E+01	3.74E+01	1.66E+02	9.81E+01	1.08E+02	4.81E+00	2.31E+01	4.74E+01	3.99E+01	**3.53E+00**	2.89E+01	9.78E+00	1.41E+01	1.28E+01
	Std.Dev	8.50E+00	1.89E+01	3.70E+01	2.19E+01	2.57E+01	3.52E+00	9.53E+00	3.04E+01	9.17E+00	4.42E+00	1.36E+01	7.29E+00	8.82E+00	4.31E+00
	Median	1.35E+01	3.23E+01	1.65E+02	9.53E+01	1.09E+02	3.63E+00	2.10E+01	3.41E+01	3.84E+01	**2.04E+00**	2.66E+01	7.71E+00	1.31E+01	1.21E+01
	SR	100	100	5	56	37	100	100	97	100	100	100	100	100	100
	Mean(SP)	7818	14,420	389,424	146,485	217,710	**4356**	12,288	11,263	38,786	5375	13,965	51,042	11,725	10,937
f_8	*Mean*	**2.09E+01**	**2.09E+01**	2.10E+01	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	2.10E+01	**2.09E+01**	**2.09E+01**	**2.09E+01**	2.15E+01	2.15E+01	**2.09E+01**
	Std.Dev	5.56E−02	5.06E−02	5.66E−02	4.88E−01	4.85E−02	4.85E−02	4.52E−02	4.80E−02	5.48E−02	6.82E−02	5.24E−02	1.25E−01	9.97E−02	4.50E−02
	Median	**2.09E+01**	2.10E+01	2.11E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.09E+01	**2.09E+01**	**2.09E+01**	2.10E+01	2.16E01	2.15E+01	**2.09E+01**
	SR	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	Mean(SP)	40	40	1700	496	1050	205	48	200	50	50	100	210	**14**	120
f_9	*Mean*	1.78E+01	1.59E+01	2.38E+01	3.02E+01	2.60E+01	**1.01E+01**	2.03E+01	1.26E+01	2.41E+01	2.65E+01	1.73E+01	1.36E+01	4.26E+01	1.18E+01
	Std.Dev	4.37E+00	3.25E+00	2.83E+00	2.26E+00	3.95E+00	2.52E+00	3.82E+00	3.05E+00	3.13E+00	1.76E+00	2.73E+00	2.67E+00	8.25E+00	3.29E+00
	Median	1.72E+01	1.56E+01	2.36E+01	3.03E+01	2.64E+01	**1.02E+01**	2.08E+01	1.26E+01	2.44E+01	2.68E+01	1.71E+01	1.36E+01	4.32E+01	1.14E+01
	SR	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	Mean(SP)	43	40	1360	472	1050	205	48	200	50	50	100	210	**14**	120
f_{10}	*Mean*	9.50E−02	2.46E−01	1.14E+02	4.22E−01	6.05E+01	2.62E−01	2.63E−01	8.86E+01	1.04E−01	4.11E−02	2.98E−01	3.47E−02	3.32E−02	**3.30E−02**
	Std.Dev	4.62E−02	8.64E−02	6.06E+01	2.17E−01	7.73E+01	1.28E−01	1.51E−01	8.95E+01	6.76E−02	2.06E−02	1.51E−01	2.07E−02	2.23E−02	2.83E−02
	Median	8.87E−02	2.34E−01	1.15E+02	3.91E−01	4.40E+01	2.36E−01	2.32E−01	7.77E+01	7.89E−02	3.94E−02	2.92E−01	2.96E−02	**2.46E−02**	2.96E−02
	SR	100	100	45	100	77	100	100	77	100	100	100	100	100	100
	Mean(SP)	**2685**	13,722	309,220	8432	86,240	6519	36,199	82,486	4730	3870	8215	23,808	4115	4033
f_{11}	*Mean*	3.92E+01	4.16E+01	1.05E+00	1.52E−13	7.20E+01	1.48E+01	1.10E−13	3.81E+01	2.54E−14	**0.00E+00**	2.49E−01	1.99E-02	9.39E+01	**0.00E+00**
	Std.Dev	1.12E+01	1.25E+01	3.21E+00	4.56E−14	2.44E+01	4.70E+00	3.16E−14	1.69E+01	2.71E−14	0.00E+00	6.06E-01	1.40E−01	2.67E+02	0.00E+00
	Median	3.78E+01	4.06E+01	9.10E−09	1.71E−13	7.00E+01	1.39E+01	1.14E−13	4.28E+01	1.68E−14	**0.00E+00**	**0.00E+00**	5.68E−14	4.88E+01	**0.00E+00**
	SR	100	100	100	100	88	100	100	100	100	100	100	100	97	100
	Mean(SP)	16,941	39,286	75,454	**1181**	46,530	27,980	47,347	10,153	1215	9605	13,535	74,316	13,837	1772
f_{12}	*Mean*	4.14E+01	1.70E+02	1.37E+02	1.09E+02	1.57E+02	1.62E+02	6.05E+01	7.56E+01	1.08E+02	**2.37E+01**	4.86E+01	3.71E+01	6.57E+02	6.24E+01
	Std.Dev	8.85E+00	1.08E+01	4.56E+01	3.23E+01	4.79E+01	9.09E+00	1.84E+01	2.66E+01	4.57E+01	4.80E+00	1.23E+01	1.21E+01	9.74E+02	2.02E+01
	Median	4.08E+01	1.70E+02	1.28E+02	1.06E+02	1.44E+02	1.62E+02	5.77E+01	5.64E+01	9.65E+01	**2.36E+01**	4.78E+01	3.48E+01	5.17E+01	5.67E+01
	SR	100	0	24	40	9	0	96	97	55	100	100	100	70	99
	Mean(SP)	**14,461**	–	396,280	314,356	281,550	–	88,091	74,844	501,150	21,680	53,670	82,560	96,286	63,650

Table 3 (*Continued*)

		DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
f_{13}	*Mean*	8.63E+01	1.71E+02	2.04E+02	1.74E+02	2.58E+02	1.60E+02	1.27E+02	1.10E+02	1.56E+02	**4.41E+01**	9.84E+01	7.79E+01	1.75E+03	1.24E+02
	Std.Dev	1.61E+01	1.09E+01	3.05E+01	3.22E+01	4.95E+01	9.67E+00	2.50E+01	2.09E+01	2.41E+01	1.21E+01	2.32E+01	2.49E+01	1.76E+03	2.68E+01
	Median	8.67E+01	1.71E+02	2.12E+02	1.76E+02	2.60E+02	1.61E+02	1.30E+02	1.16E+02	1.56E+02	**4.23E+01**	9.75E+01	7.52E+01	1.74E+03	1.32E+02
	SR	80	0	0	1	0	0	19	27	0	100	56	82	19	23
	Mean(SP)	95,039	–	–	577,419	–	–	338,405	275,221	–	**48,472**	211,217	181,690	249,678	343,630
f_{14}	*Mean*	2.36E+03	2.40E+03	3.85E+01	6.32E−02	2.00E+03	7.14E+02	1.04E+01	1.33E+03	5.89E−02	**2.60E−02**	9.40E−01	8.69E+02	5.39E+03	1.21E+00
	Std.Dev	4.58E+02	5.46E+02	5.68E+01	3.00E−02	6.18E+02	2.44E+02	3.23E+01	3.39E+02	2.54E−02	2.14E−02	1.30E+00	9.83E+02	7.90E+02	1.11E+00
	Median	2.41E+03	2.40E+03	7.28E+00	6.25E−02	1.92E+03	7.04E+02	2.65E−01	1.46E+03	6.25E−02	**2.08E−02**	1.46E−01	5.79E+02	5.36E+03	1.30E+00
	SR	0	0	86	100	0	0	92	0	100	100	100	26	0	100
	Mean(SP)	–	–	215,320	**7190**	–	–	199,970	–	5448	45,370	151,810	636,012	–	52,223
f_{15}	*Mean*	**2.94E+03**	6.31E+03	7.46E+03	4.28E+03	4.09E+03	4.74E+03	3.43E+03	3.21E+03	4.62E+03	3.26E+03	4.66E+03	3.47E+03	5.14E+03	4.07E+03
	Std.Dev	3.91E+02	3.59E+02	1.07E+03	5.38E+02	6.74E+02	2.38E+03	5.79E+02	1.09E+03	4.26E+02	2.74E+02	1.02E+03	5.12E+02	6.57E+02	6.72E+02
	Median	**2.97E+03**	6.32E+03	7.89E+03	4.25E+03	4.10E+03	6.20E+03	3.47E+03	2.96E+03	4.47E+03	3.25E+03	5.07E+03	3.52E+03	5.00E+03	3.98E+03
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{16}	*Mean*	1.08E+00	2.48E+00	3.34E+00	1.09E+00	7.03E−01	2.47E+00	1.35E+00	2.35E+00	1.33E+00	1.91E+00	2.22E+00	3.56E−01	**8.72E−02**	1.28E+00
	Std.Dev	2.37E−01	2.68E−01	4.50E−01	2.92E−01	3.63E−01	2.65E−01	2.56E−01	2.96E−01	2.69E−01	6.47E−01	2.92E−01	1.86E−01	6.20E−02	3.99E−01
	Median	1.08E+00	2.52E+00	3.39E+00	1.10E+00	6.26E−01	2.46E+00	1.35E+00	2.56E+00	1.37E+00	2.04E+00	2.23E+00	2.88E−01	**6.70E−02**	1.28E+00
	SR	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	Mean(SP)	43	40	1700	473	1050	205	48	120	50	50	100	210	**14**	120
f_{17}	*Mean*	6.94E+01	1.64E+02	3.07E+01	**3.01E+01**	1.01E+02	1.61E+02	3.06E+01	6.54E+01	3.04E+01	3.04E+01	3.05E+01	3.47E+01	4.07E+03	3.06E+01
	Std.Dev	9.71E+00	1.10E+01	1.93E−01	3.04E+00	1.89E+01	1.32E+01	9.62E−02	7.25E+00	8.56E−03	5.00E−14	1.36E−01	7.39E+00	9.61E+02	9.53E−02
	Median	6.90E+01	1.65E+02	3.07E+01	3.04E+01	1.01E+02	1.62E+02	3.06E+01	6.36E+01	3.04E+01	**3.04E+01**	**3.04E+01**	3.14E+01	4.19E+03	3.06E+01
	SR	99	0	100	100	47	0	100	100	100	100	100	100	2	100
	SP	48,261	–	104,150	3002	192,742	–	122,650	98,816	2629	13,025	23,345	123,726	303,933	**2509**
f_{18}	*Mean*	7.35E+01	1.97E+02	2.29E+02	1.44E+02	1.59E+02	1.92E+02	9.74E+01	2.10E+02	1.48E+02	7.71E+01	1.29E+02	**6.40E+01**	4.17E+03	1.26E+02
	Std.Dev	1.31E+01	1.03E+01	2.54E+01	2.89E+01	3.74E+01	9.93E+00	2.70E+01	1.09E+01	5.02E+01	6.70E+00	4.26E+01	1.06E+01	9.24E+02	3.10E+01
	Median	7.06E+01	1.98E+02	2.36E+02	1.37E+02	1.55E+02	1.93E+02	9.21E+01	2.03E+02	1.32E+02	7.70E+01	1.48E+02	**6.34E+01**	4.20E+03	1.18E+02
	SR	94	0	0	6	5	0	66	0	32	100	27	100	1	23
	Mean(SP)	**77,602**	–	–	397,329	310,367	–	300,050	–	827,812	83,850	479,775	219,576	313,352	337,555
f_{19}	*Mean*	3.07E+00	1.13E+01	2.54E+00	1.02E+00	8.08E+00	3.43E+00	1.42E+00	6.31E+00	8.30E−01	1.47E+00	4.07E+00	2.32E+00	3.53E+00	**8.58E−01**
	Std.Dev	7.92E−01	1.20E+00	4.65E−01	3.09E−01	7.02E+00	5.25E−01	2.45E−01	1.83E+02	2.26E−01	9.94E−02	8.50E−01	2.17E+00	9.43E−01	2.31E−01
	Median	3.05E+00	1.14E+01	2.57E+00	1.00E+00	5.11E+00	3.43E+00	1.44E+00	3.32E+00	8.41E−01	1.48E+00	4.09E+00	1.70E+00	3.33E+00	**8.45E−01**
	SR	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	Mean(SP)	3118	3156	42,232	**942**	2850	2714	13,713	3637	1030	2230	5015	15,042	2512	1677
f_{20}	*Mean*	1.38E+01	1.31E+01	1.44E+01	1.19E+01	1.42E+01	1.38E+01	1.10E+01	1.32E+01	**1.05E+01**	1.07E+01	1.07E+01	1.06E+01	1.27E+01	**1.05E+01**
	Std.Dev	1.92E+00	1.13E+00	5.89E−01	8.12E−01	1.01E+00	1.34E+00	7.39E−01	1.46E+00	9.97E−01	6.10E−01	7.38E−01	6.48E−01	8.98E−01	6.54E−01
	Median	1.50E+01	1.28E+01	1.45E+01	1.18E+01	1.45E+01	1.50E+01	1.10E+01	1.50E+01	**1.05E+01**	1.06E+01	1.08E+01	1.06E+01	1.27E+01	**1.05E+01**
	SR	100	100	100	100	100	100	100	100	100	100	100	100	100	100
	Mean(SP)	41	40	1700	140	1050	205	48	200	50	50	100	210	**14**	120

Table 4
Comparison results on the 8 composition functions (f_{21}–f_{28}).

		DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
f_{21}	*Mean*	3.39E+02	3.23E+02	**2.19E+02**	2.96E+02	3.00E+02	2.97E+02	2.28E+02	2.94E+02	3.27E+02	2.93E+02	3.15E+02	2.81E+02	3.11E+02	2.73E+02
	Std.Dev	9.48E+01	8.21E+01	5.13E+01	8.21E+01	9.53E+01	8.15E+01	4.18E+01	7.63E+01	7.65E+01	7.47E+01	8.83E+01	8.67E+01	9.22E+01	8.43E+01
	Median	3.00E+02	3.00E+02	**2.00E+02**	3.00E+02	3.00E+02	3.00E+02	2.01E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02	3.00E+02
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	8
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	**423,277**
f_{22}	*Mean*	2.00E+03	1.70E+03	3.00E+02	1.08E+02	2.32E+03	6.05E+02	1.12E+02	1.80E+03	1.09E+02	1.04E+02	1.24E+02	7.56E+02	7.18E+03	**1.01E+02**
	Std.Dev	5.92E+02	6.48E+02	1.83E+02	6.38E+01	6.37E+02	2.64E+02	1.60E+01	5.44E+02	6.43E+01	1.72E+01	5.32E+01	8.18E+02	8.73E+02	5.15E+01
	Median	2.04E+03	1.59E+03	2.79E+02	**1.00E+02**	2.35E+03	5.59E+02	1.15E+02	1.67E+03	1.11E+02	1.06E+02	1.15E+02	4.86E+02	7.26E+03	**1.08E+02**
	SR	0	0	2	17	0	0	7	7	15	3	0	0	0	17
	Mean(SP)	–	–	462,682	**256,648**	–	–	499,980	425,661	404,624	–	–	–	–	278,162
f_{23}	*Mean*	**3.32E+03**	6.87E+03	7.47E+03	4.93E+03	4.87E+03	3.93E+03	3.99E+03	3.45E+03	5.89E+03	3.64E+03	4.84E+03	3.60E+03	6.74E+03	4.28E+03
	Std.Dev	4.92E+02	5.22E+02	1.24E+03	6.53E+02	9.48E+02	2.49E+03	5.37E+02	1.26E+03	9.71E+02	3.87E+03	1.06E+03	6.36E+02	8.65E+02	7.96E+02
	Median	3.33E+03	6.86E+03	8.00E+03	4.94E+03	**4.90E+02**	2.83E+03	4.03E+03	3.39E+03	6.02E+03	3.61E+03	5.20E+03	3.58E+03	6.89E+03	4.42E+03
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{24}	*Mean*	2.20E+02	2.40E+02	2.73E+02	2.75E+02	2.72E+02	2.24E+02	2.30E+02	2.57E+02	2.65E+02	**2.13E+02**	2.27E+02	2.23E+02	6.74E+02	2.28E+02
	Std.Dev	1.12E+01	1.47E+01	5.44E+00	9.58E+00	1.03E+01	1.18E+01	8.21E+00	1.21E+01	9.50E+00	1.35E+01	6.80E+00	9.31E+00	5.86E+02	7.95E+00
	Median	2.18E+02	2.39E+02	2.73E+02	2.76E+02	2.73E+02	2.24E+02	2.30E+02	2.62E+02	2.67E+02	**2.09E+02**	2.27E+02	2.23E+02	2.40E+02	2.29E+02
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{25}	*Mean*	**2.53E+02**	2.63E+02	2.83E+02	2.93E+02	2.96E+02	**2.53E+02**	2.71E+02	2.92E+02	2.49E+02	2.75E+02	2.65E+02	2.56E+02	3.27E+02	2.60E+02
	Std.Dev	3.25E+01	2.33E+01	5.49E+00	7.96E+00	1.10E+01	5.93E+00	1.84E+01	1.67E+01	6.39E+01	9.76E+00	1.29E+01	7.37E+00	1.76E+02	6.92E+00
	Median	2.67E+02	2.71E+02	2.84E+02	2.95E+02	2.97E+02	**2.53E+02**	2.75E+02	2.91E+02	2.85E+02	2.77E+02	2.69E+02	2.57E+02	**2.53E+02**	2.61E+02
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{26}	*Mean*	2.14E+02	3.13E+02	2.28E+02	2.32E+02	2.41E+02	2.44E+02	**2.00E+02**	2.56E+02	**2.00E+02**	2.19E+02	2.03E+02	2.04E+02	5.10E+02	2.10E+02
	Std.Dev	4.05E+01	4.83E+01	5.35E+01	6.74E+01	7.00E+01	5.35E+01	2.69E−02	6.85E+01	4.06E−03	4.64E+01	1.80E+01	2.22E+01	4.84E+02	3.30E+01
	Median	**2.00E+02**	3.30E+02	2.06E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	3.30E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	3.32E+02	**2.00E+02**
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{27}	*Mean*	5.69E+02	6.56E+02	9.44E+02	1.07E+03	9.60E+02	4.84E+02	5.95E+02	7.66E+02	9.49E+02	6.99E+02	6.10E+02	6.13E+02	5.61E+02	**4.47E+02**
	Std.Dev	1.23E+02	1.27E+02	5.96E+02	7.91E+01	1.07E+02	1.21E+02	1.58E+02	1.29E+02	6.05E+01	2.21E+02	7.60E+01	1.05E+02	1.27E+02	3.94E+01
	Median	5.49E+02	6.59E+02	9.47E+02	1.08E+03	9.52E+02	5.16E+02	5.73E+02	7.43E+02	9.59E+02	7.46E+02	6.14E+02	6.37E+02	5.45E+02	**4.43E+02**
	SR	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mean(SP)	–	–	–	–	–	–	–	–	–	–	–	–	–	–
f_{28}	*Mean*	2.84E+02	3.12E+02	1.26E+03	5.78E+02	1.05E+03	3.11E+02	2.96E+02	1.05E+03	2.88E+02	3.00E+02	3.00E+02	3.00E+02	1.14E+03	**2.74E+02**
	Std.Dev	5.45E+01	2.67E+02	2.63E+02	5.67E+02	6.36E+02	1.05E+05	2.76E+01	4.72E+02	2.26E+01	0.00E+00	0.00E+00	0.00E+00	3.26E+03	6.76E+01
	Median	**3.00E+02**	**3.00E+02**	1.20E+03	**3.00E+02**	1.16E+03	**3.00E+02**	**3.00E+02**	1.09E+03	3.00E+02	**3.00E+00**	**3.00E+02**	**3.00E+02**	**3.00E+02**	**3.00E+02**
	SR	7	0	0	0	0	0	0	0	0	0	0	0	0	13
	Mean(SP)	435,723	–	–	–	–	–	–	–	–	–	–	–	–	**429,556**

Table 5
t-Results between MSPSO and other 13 peer algorithms.

	DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES
Sum(+)	2, 8, 3	3, 15, 6	5, 15, 8	4, 12, 8	5, 14, 8	5, 13, 5	4, 10, 4	5, 10, 6	4, 8, 4	3, 5, 3	3, 9, 6	4, 6, 3	3, 12, 8
Sum(−)	1, 5, 3	1, 0, 0	0, 0, 0	0, 2, 0	0, 0, 0	0, 2, 2	1, 2, 4	0, 1, 0	1, 4, 2	1, 7, 2	1, 3, 1	1, 6, 3	2, 1, 0
Sum(=)	2, 2, 2	1, 0, 2	0, 0, 0	1, 1, 0	0, 1, 0	0, 0, 1	0, 3, 0	0, 4, 2	0, 3, 2	1, 3, 3	1, 3, 1	0, 3, 2	0, 2, 0
CP	1, 3, 0	2, 15, 6	5, 15, 8	4, 10, 8	5, 14, 8	5, 11, 3	3, 8, 0	5, 9, 6	3, 4, 2	2, −2, 1	2, 6, 5	3, 0, 0	1, 11, 8

Table 6
Friedman-test results on the mean results.

Average rank	Algorithm	Ranking	Unimodal functions		Multimodal functions		Composition functions	
			Algorithm	Ranking	Algorithm	Ranking	Algorithm	Ranking
1	MSPSO	3.70	DMSPSO	3.10	MSPSO	3.87	MSPSO	3.63
2	JADE	4.61	MSPSO	3.30	JADE	3.93	HCLPSO	4.81
3	CoDE	4.88	CMA-ES	4.20	CoDE	4.90	CoDE	5.06
4	DMSPSO	5.39	CoDE	4.50	SopPSO	5.57	DMSPSO	5.44
5	HCLPSO	6.09	JADE	5.30	HCLPSO	6.00	JADE	5.44
6	SopPSO	6.21	SADE	5.30	DMSPSO	6.13	SL_SPO	6.06
7	SADE	6.52	SopPSO	7.00	SADE	7.00	SADE	6.38
8	SL_PSO	7.64	SL_PSO	7.50	SLPSO	7.37	SopPSO	6.94
9	SLPSO	8.80	HCLPSO	8.40	SL_PSO	8.53	SSS-APSO	9.06
10	SSS-APSO	9.57	F-PSO	8.60	SSS-APSO	9.37	OLPSO	9.63
11	CMA-ES	9.59	SSS-APSO	11.00	CMA-ES	10.23	SLPSO	9.75
12	F-PSO	9.95	SLPSO	11.60	F-PSO	10.43	F-PSO	9.88
13	PSODDS	10.96	PSODDS	11.80	PSODDS	10.57	PSODDS	11.19
14	OLPSO	11.09	OLPSO	13.40	OLPSO	11.10	CMA-ES	11.75
Statistic	129.948		43.410		73.695		41.031	
p value	<0.001		<0.001		<0.001		<0.001	

4.2.4. Test of significance

1) *Comparison on t-test*: To investigate whether MSPSO is significantly better or worse than the other 13 peer algorithms on the 28 test functions, a two-tailed *t*-test is carried out in this section. The results of *t*-test are presented in Table 5 in which the symbols "*Sum*(+)", "*Sum*(−)" and "*Sum*(=)" denote the number that MSPSO are significantly better than, significantly worse than, and almost the same as the corresponding competitor algorithms, respectively. The comprehensive performance (*CP*) is equal to "*Sum*(+)" minus "*Sum*(−)". There are three integers in each cell of the table, which mean the number of *t*-test results on unimodal functions, multimodal functions, and composition functions, respectively. For example, three integers "2, 8, 3" in the second row and the second column denote the number of functions that MSPSO is significantly better than DMSPSO on the three types functions are 2, 8, and 3, respectively.

The values of *CP* in Table 5 manifest that MSPSO significantly outperforms 9 peer algorithms (i.e., F-PSO, OLPSO, SLPSO, PSODDS, SL_PSO, SSS-APSO, SopPSO SaDE, and CMA-ES) on all the three classes functions. Although sharing the same characteristic with DMSPSO, i.e., dynamical regrouping for sub-swarms, MSPSO attains more competitive results than DMSPSO on the unimodal functions and the multimodal functions as well as offers the same performance on the composition functions as DMSPSO. Moreover, the *t*-test results manifest that JADE is the only algorithm that dominates MSPSO on a class of problems. The comparison results demonstrate that MSPSO performs slightly weaker than JADE on the multimodal functions.

2) *Comparison on Friedman-test*: The Friedman-test results of *Mean* value and *Median* value are listed in Tables 6 and 7 respectively, in which algorithms are listed in ascending order based on their ranking values (the lower the better). Furthermore, we also separately carry out the Friedman-test of all peer algorithms on the three types functions in terms of *Mean* value and *Median* value, the results of which are also listed in Tables 6 and 7, respectively. The statistics and the corresponding *p* values are shown at the bottom of the tables.

It can be observed from Table 6 that MSPSO achieves the best comprehensive performance on all the 28 test functions, followed by JADE, CoDE, and DMSPSO. Furthermore, DMSPSO attains the most promising performance on the unimodal functions while DMSPSO achieves the best result on the multimodal and composition functions. In addition, as a variant of CLPSO, HCLPSO demonstrates more favorable performance on the multimodal functions and composition functions than that on the unimodal problems.

The Friedman-test results in Table 7 demonstrate that JADE offers the most favorable characteristic among all the test functions, in terms of median values, followed by MSPSO, CoDE, and DMSPSO. Note that MSPSO performs slightly weaker than JADE and SL_PSO on the multimodal and composition functions, respectively, although it offers the best performance on the unimodal functions.

4.3. Time usage

The experiment of time usage of all the peer algorithms is conducted in this section. And the results are demonstrated in Table 8. The comparison results of average CPU times presented at the bottom in Table 8 indicate that CMA-ES displays the most promising performance on all the test functions, followed by CoDE, JADE, and SL_PSO. Although MSPSO yields very promising performance, in terms of solutions accuracy, it displays a moderate performance on time usage. Note that 3 out of the 4 non-PSO algorithms shows more favorable performance. From the experimental results on the 5 unimodal functions we can see that MSPSO offers very unsatisfied results. The reason is that there is no local optima in these functions, which causes the detecting operator in MSPSO to wastes too many CPU time. On the contrary, for the 8 composition functions, MSPSO yields almost the same results as OLPSO, SLPSO, and SL_PSO. The phenomenon verifies that MSPSO is more suitable for the complicated functions.

Table 7
Friedman-test results on the median results.

Average rank	Algorithm	Ranking	Unimodal functions		Multimodal functions		Composition functions	
			Algorithm	Ranking	Algorithm	Ranking	Algorithm	Ranking
1	JADE	4.16	DMSPSO	3.40	JADE	3.47	SL_PSO	4.69
2	MSPSO	4.46	JADE	3.50	MSPSO	4.30	MSPSO	5.13
3	CoDE	5.02	MSPSO	3.90	CoDE	4.80	CoDE	5.63
4	DMSPSO	5.54	CMA-ES	4.40	SopPSO	5.63	DMSPSO	5.63
5	HCLPSO	6.59	CoDE	4.70	DMSPSO	5.87	HCLPSO	5.63
6	SADE	6.66	SADE	6.00	HCLPSO	6.20	JADE	5.88
7	SopPSO	6.84	SopPSO	7.40	SADE	6.93	SADE	6.56
8	SL_PSO	7.59	SL_PSO	7.50	SLPSO	7.90	CMA-ES	8.63
9	CMA-ES	8.18	F-PSO	8.10	SSS-APSO	8.87	SopPSO	8.75
10	SLPSO	8.80	HCLPSO	8.70	SL_PSO	9.17	SLPSO	8.88
11	SSS-APSO	9.54	SSS-APSO	11.20	CMA-ES	9.20	F-PSO	9.00
12	F-PSO	9.61	PSODDS	11.40	F-PSO	10.43	SSS-APSO	9.75
13	PSODDS	10.88	SLPSO	11.40	PSODDS	10.73	OLPSO	10.06
14	OLPSO	11.14	OLPSO	13.40	OLPSO	10.97	PSODDS	10.81
Statistic	115.841		42.922		68.746		32.094	
p value	<0.001		<0.001		<0.001		0.005	

Table 8
Comparison of time usage (s).

	DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
f_1	21.78	19.96	11.43	16.28	24.85	11.86	27.42	18.41	16.43	11.21	37.52	10.94	9.79	16.17
f_2	29.83	21.88	13.08	10.72	27.26	10.63	30.48	20.49	18.91	13.52	38.01	10.24	9.95	15.84
f_3	24.00	22.73	12.73	11.51	25.30	12.61	28.11	22.73	18.76	13.82	40.53	9.72	9.96	15.79
f_4	24.34	23.95	11.77	10.15	25.93	11.52	28.62	19.10	15.18	11.93	39.63	9.03	9.17	15.55
f_5	18.91	22.06	11.67	13.04	25.37	11.93	27.91	18.95	15.51	10.28	38.40	9.73	8.85	14.26
f_6	29.87	21.42	12.42	9.73	25.55	12.02	30.66	19.82	13.34	15.77	34.05	9.15	8.54	14.47
f_7	25.46	26.96	16.89	15.26	31.32	15.10	35.95	21.89	20.06	20.10	38.81	14.41	13.23	19.18
f_8	24.63	25.34	15.84	14.17	29.75	15.81	34.62	20.37	18.78	13.82	39.31	14.76	11.85	17.78
f_9	52.25	58.65	47.19	47.66	64.44	44.53	73.74	51.25	48.51	46.37	67.15	33.22	36.82	45.40
f_{10}	18.86	31.95	13.55	13.77	25.82	12.48	29.13	17.62	19.84	8.30	35.29	12.93	11.31	15.91
f_{11}	20.20	28.06	13.99	14.16	27.10	13.11	32.69	18.26	23.62	9.34	36.07	13.66	12.89	16.62
f_{12}	23.18	23.46	15.20	13.45	27.81	14.89	33.06	19.98	26.81	11.53	37.47	13.16	12.78	17.66
f_{13}	24.12	23.03	15.01	13.37	29.88	14.76	31.85	19.73	24.82	11.45	40.29	13.17	12.75	17.62
f_{14}	22.83	21.62	13.98	12.07	31.78	12.47	35.72	18.31	19.06	9.65	41.60	12.58	11.90	16.53
f_{15}	22.97	22.07	14.27	12.59	29.67	13.94	42.92	19.06	20.32	10.66	37.45	13.45	12.19	17.50
f_{16}	27.36	26.90	19.25	17.86	32.85	18.93	56.26	23.79	23.81	15.93	43.03	17.15	15.77	21.74
f_{17}	18.70	20.17	12.96	10.91	24.93	12.24	43.00	17.05	16.90	8.30	35.32	10.72	9.89	15.72
f_{18}	19.66	21.04	13.57	11.85	26.22	13.24	34.29	18.06	17.03	9.61	35.12	11.13	9.61	16.72
f_{19}	18.71	19.40	12.35	10.05	24.28	11.56	26.06	16.33	16.73	7.69	33.45	10.29	9.38	15.23
f_{20}	51.39	21.69	14.10	13.46	26.40	13.31	30.82	18.31	18.51	9.08	34.74	10.65	9.51	16.56
f_{21}	28.00	28.48	21.19	21.47	34.42	20.44	33.25	25.54	25.41	17.00	42.94	18.44	17.24	23.33
f_{22}	31.72	29.88	22.57	20.30	42.25	21.90	38.64	27.08	27.04	17.54	44.76	19.20	17.92	24.44
f_{23}	34.77	31.15	23.97	22.69	37.94	22.54	40.21	28.50	37.45	19.02	46.59	20.46	18.25	26.22
f_{24}	62.80	62.45	56.56	56.86	72.41	51.68	74.92	61.21	65.63	50.53	75.46	39.62	40.38	53.85
f_{25}	64.68	63.61	57.53	56.98	70.39	53.30	77.49	62.57	63.35	52.36	83.04	40.02	43.20	55.25
f_{26}	71.87	68.73	62.73	62.46	76.45	58.47	81.72	67.13	72.74	64.21	94.12	43.07	47.80	59.95
f_{27}	68.23	67.15	61.69	61.38	75.99	56.41	81.81	65.90	66.79	62.57	115.28	45.37	47.38	58.32
f_{28}	35.74	36.69	29.35	28.21	43.10	27.55	41.97	34.29	33.62	25.65	58.20	25.63	22.12	30.60
Avg.	32.75	31.80	23.10	22.23	37.12	21.76	42.26	28.28	28.75	20.62	47.99	18.28	17.87	24.79

4.4. Efficiency of involved strategies

To measure how the three components, i.e., DNS, SRS, and PDS, affect the performance of MSPSO, a series of experiments are also carried out in this research. In the experiment, the performances of MSPSO without DNS, SRS, and PDS are tested under the same runtime environment. The results are listed in Table 9, in terms of the mean and the standard deviation of the solutions. Furthermore, the two-tailed *t*-test is conducted to compare the performance between MSPSO and the other three algorithms. In Table 9, MSPSO-SP, MSPSO-DP, and MSPSO-DS indicate removing DNS, SRS, and PDS from MSPSO, respectively. Note that the size of each sub-swarm in MSPSO-SP is 3. The results of *t*-test are described as "+", "−" and "=", which denote that MSPSO is significantly better than, significantly worse than, and almost the same as the corresponding competitors, respectively.

It can be observed from Table 9 that MSPSO yields the best results on 12 out of the 28 benchmark functions in terms of the mean solution accuracy, followed by MSPSO-DS, MSPSO-SP and MSPSO-DP.

Comparing the results yielded by MSPSO and MSPSO-DP we can see that the former algorithm dominates the latter one on 24 out of the 28 benchmark functions, in terms of the mean values. Moreover, the *t*-test results demonstrate that MSPSO is significantly better than MSPSO-DP on 21 functions while MSPSO offers almost the same performance as MSPSO-DP on 5 functions. From the comparison results we know that SRS causes some useful information shared among various sub-swarms plays a very positive performance for different type problems.

The comparison results between MSPSO and MSPSO-DS also manifest that MSPSO has more favorable performance since it outperforms MSPSO-DS on 15 out of the 28 functions. On the contrary,

Table 9
Performance of proposed modules.

	MSPSO	MSPSO-DP	MSPSO-DS	MSPSO-SP
f_1	**0.00E+00** ± 0.00E+00	6.37E−13 ± 1.64E−13(+)	**0.00E+00** ± 0.00E+00(=)	2.27E−14 ± 4.09E−14(+)
f_2	**2.20E−03** ± 4.78E−03	1.37E−01 ± 1.08E−01(+)	2.34E−01 ± 1.65E−01(+)	5.58E−01 ± 5.79E−01(+)
f_3	1.21E+07 ± 2.03E+07	1.20E+08 ± 1.34E+08(+)	**7.43E+06** ± 7.84E+06(−)	5.47E+07 ± 4.89E+07(+)
f_4	1.79E−02 ± 3.20E−02	9.28E−03 ± 4.64E−03(−)	**7.39E−03** ± 7.55E−03(−)	3.56E−02 ± 3.69E−02(+)
f_5	9.21E−14 ± 4.48E−14	6.14E−13 ± 1.91E−13(+)	1.25E−13 ± 2.05E−14(+)	**9.09E−14** ± 3.64E−14(=)
f_6	**7.20E−11** ± 9.06E−11	1.59E+00 ± 1.91E+00(+)	3.99E−01 ± 7.18E−01(+)	7.97E−01 ± 1.28E+00(+)
f_7	**1.28E+01** ± 4.31E+00	6.09E+01 ± 1.29E+01(+)	1.29E+01 ± 4.54E+00(=)	1.67E+01 ± 3.34E+00(+)
f_8	**2.09E+01** ± 4.50E−02	**2.09E+01** ± 3.08E−02(=)	**2.09E+01** ± 5.61E−02(=)	**2.09E+01** ± 5.39E−02(=)
f_9	**1.18E+01** ± 3.29E+00	3.05E+01 ± 1.40E+00(+)	1.67E+01 ± 3.41E+00(+)	1.83E+01 ± 4.11E+00(+)
f_{10}	3.30E−02 ± 2.83E−02	3.69E−02 ± 1.08E−02(=)	**2.71E−02** ± 1.28E−02(=)	2.88E−02 ± 1.35E−02(=)
f_{11}	**0.00E+00** ± 0.00E+00	2.50E−13 ± 1.18E−13(+)	3.44e+01 ± 6.49e+00(+)	1.14E−14 ± 1.82E−14(+)
f_{12}	6.24E+01 ± 2.02E+01	1.25E+02 ± 2.79E+01(+)	6.49E+01 ± 1.36E+01(=)	**6.10E+01** ± 1.90E+01(=)
f_{13}	1.24E+02 ± 2.68E+01	1.63E+02 ± 1.73E+01(+)	1.27E+02 ± 2.29E+01(=)	**9.95E+01** ± 1.75E+01(−)
f_{14}	1.21E+00 ± 1.11E+00	2.36E+00 ± 8.46E−01(+)	1.98E+03 ± 2.17E+02(+)	**4.65E−01** ± 5.47E−01(−)
f_{15}	4.07E+03 ± 6.72E+02	4.59E+03 ± 2.98E+02(+)	**3.71E+03** ± 3.94E+02(−)	4.10E+03 ± 5.11E+02(=)
f_{16}	1.28E+00 ± 3.99E−01	**1.24E+00** ± 2.74E−01(=)	1.45E+00 ± 3.25E−01(+)	1.27E+00 ± 2.39E−01(=)
f_{17}	3.06E+01 ± 9.53E−02	1.45E+02 ± 3.26E+01(+)	6.07E+01 ± 4.29E+00(+)	**3.05E+01** ± 1.85E−02(−)
f_{18}	1.26E+02 ± 3.10E+01	1.45E+02 ± 3.26E+01(+)	**1.13E+02** ± 1.15E+01(−)	1.24E+02 ± 1.96E+01(=)
f_{19}	8.58E−01 ± 2.31E−01	1.06E+00 ± 2.75E−01(+)	3.38E+00 ± 7.87E−01(+)	**7.86E−01** ± 2.93E−01(=)
f_{20}	**1.05E+01** ± 6.54E−01	1.47E+01 ± 3.58E−01(+)	1.06E+01 ± 3.74E−01(=)	1.08E+01 ± 5.15E−01(+)
f_{21}	2.73E+02 ± 8.43E+01	**2.40E+02** ± 4.80E+01(−)	3.23E+02 ± 7.21E+01(+)	3.47E+02 ± 7.67E+01(+)
f_{22}	**1.01E+02** ± 5.15E+01	1.10E+02 ± 3.06E+00(=)	2.02E+03 ± 4.93E+02(+)	1.20E+02 ± 6.46E+01(=)
f_{23}	4.28E+03 ± 7.96E+02	4.63E+03 ± 3.56E+02(+)	4.17E+03 ± 4.24E+02(=)	**3.33E+03** ± 6.86E+02(−)
f_{24}	2.28E+02 ± 7.95E+00	2.80E+02 ± 1.22E+01(+)	**2.24E+02** ± 5.72E+00(−)	2.38E+02 ± 1.21E+01(+)
f_{25}	**2.60E+02** ± 6.92E+00	3.06E+02 ± 5.99E+00(+)	2.64E+02 ± 6.93E+00(+)	2.68E+02 ± 6.38E+00(+)
f_{26}	**2.10E+02** ± 2.30E+01	2.12E+02 ± 2.56E+01(+)	2.36E+02 ± 5.07E+01(+)	2.51E+02 ± 6.16E+01(+)
f_{27}	**4.47E+02** ± 3.94E+01	1.07E+03 ± 1.95E+01(+)	5.66E+02 ± 1.13E+02(+)	5.91E+02 ± 8.17E+01(+)
f_{28}	2.74E+02 ± 6.76E+01	**2.71E+02** ± 6.81E+01(=)	5.22E+02 ± 3.56E+02(+)	2.80E+02 ± 3.60E+01(=)

Bold values signify the best results among all the algorithms.

Table 10
t-Results between *Rn* = 10 and other 4 values of *Rn* on three types problems.

	Rn = 3	*Rn* = 6	*Rn* = 15	*Rn* = 20
Sum(+)	0, 0, 0	0, 0, 4	0, 0, 0	0, 0, 0
Sum(−)	1, 1, 0	0, 1, 1	0, 1, 1	1, 0, 1
Sum(=)	4, 14, 8	5, 14, 3	5, 14, 7	4, 15, 7
CP	−1, −1, 0	0, −1,3	0, −1,−1	−1, 0, −1

MSPSO-DS dominates MSPSO only on 5 test functions. Specifically, MSPSO-DS is significantly better than MSPSO on 2 out of the 5 unimodal functions as well as obtains the same result as MSPSO on 1 unimodal problem. On the contrary, MSPSO offers more promising characteristics than MSPSO-DS on the 15 basic multimodal functions. The results verify that PDS is more conductive to multimodal problems than unimodal problems since there is no local optimal solution in unimodal problems to disturb the search direction of the population. Thus, the PDS may waste too many evaluations when optimizing unimodal functions. This conclusion is also confirmed by the comparison results between DMSPSO and MSPSO, which are listed in Tables 2–9.

From the experimental results of MSPSO and MSPSO-SP we can see that MSPSO dominates MSPSO-SP on 14 out of the 28 test functions, including 4 unimodal functions, 5 multimodal functions, and 5 composition functions. It can be observed from the comparison results on the 5 unimodal functions and 15 basic multimodal functions that MSPSO has more remarkable advantages than MSPSO-SP on the unimodal functions rather than multimodal problems. The results also verify that the multi-swarm technique does much better in optimizing multimodal functions while it sacrifices performance on unimodal functions.

The parameter *Rn* is a new introduced parameter in MSPSO representing the number of sub-regions on each dimension. To illustrate how *Rn* effects the performance of MSPSO, the experimental results of MSPSO with different *Rn* on the three types problems are listed in Table 10. Due to the space limitation, only the *t*-test results between *Rn*=10 applied in the above experiments and other 4 values of *Rn* are presented in Table 10.

The experiment on *Rn* indicates that the test functions are not very sensitive to *Rn*. There are several reasons behind the phenomenon, which are explained as follows. A larger *Rn* causes each sub-region to have a smaller space, which enables **GBEST** to take a finer detection. However, a larger *Rn* enables **GBEST** to carry out more times of detecting operators before it finds out a promising sub-region since there are more sub-regions needed to be detected. On the contrary, a smaller *Rn* enables **GBEST** to find out a proper but more coarse sub-region after fewer detecting operators.

From the comparison results aforementioned, several conclusions can be drawn for the new introduced strategies: (1) DNS improves the tradeoff between population's diversity and convergence speed to some extend; (2) SRS is conductive to valuable information diffusion among different sub-swarms, which is beneficial for both unimodal and multimodal functions optimization; and (3) PDS enhances MSPSO'S capability of escaping from local optimum, which is in favor of multimodal functions optimization, but it sacrifices the performance on unimodal functions since there is no local optimum needs to be detected.

4.5. Comparison on four real-world applications

In this part, 4 real-world applications are selected to testify MSPSO's practicability on engineering problems, including (1) F_1: parameter estimation for frequency modulated (FM) sound waves, (2) F_2: design of a gear train, (3) F_3: spread spectrum radar polly phase code design, and (4) F_4: Lennard-Jones potential problems. Due to the limitation of space, details of the 4 problems are not included in this paper. One can observe it from the corresponding literatures [28,57].

Parameters of each involved algorithm and experimental setup are the same as that in the above experiments except *MaxFEs* is set to 150,000 for each run. The experimental results listed in Table 11 include performance metrics, i.e., the mean and the best values of 30 runs. Furthermore, the rank values of the two performance metrics,

Table 11
Comparison results on 4 real applications.

		DMSPSO	F-PSO	OLPSO	SLPSO	PSODDS	SL_PSO	HCLPSO	SSS-APSO	SopPSO	JADE	SaDE	CoDE	CMA-ES	MSPSO
F_1	*Mean*	7.63E+00	1.21E+01	1.31E+01	9.66E+00	1.91E+01	1.05e+01	3.32E+00	1.14e+01	8.74E+00	1.87E−01	0.00E+00	1.02E+00	2.03E+01	7.25E+00
	Rank1	6	11	12	8	12	9	4	10	7	2	**1**	3	14	5
	Best	0.00E+00	8.42E+00	1.45E−20	4.74E−11	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.05E−13	0.00E+00	0.00E+00	8.42E+00	0.00E+00
	Rank2	**1**	14	10	12	**1**	**1**	**1**	**1**	**1**	11	**1**	**1**	13	**1**
F_2	*Mean*	0.00E+00	2.85E−14	2.85E−21	1.86E−29	0.00E+00	0.00E+00	1.49E−15	9.45E−15	5.18E−25	2.39E−17	7.67E−18	1.94E−17	5.91E-34	0.00E+00
	Rank	**1**	14	8	6	**1**	**1**	12	13	7	11	9	10	5	**1**
	Best	0.00E+00	7.51E−17	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.75E−21	1.32E−22	0.00E+00	0.00E+00	2.39E−29	0.00E+00	0.00E+00	0.00E+00
	Rank2	**1**	14	**1**	**1**	**1**	**1**	13	12	**1**	**1**	11	**1**	**1**	**1**
F_3	*Mean*	1.24E+00	1.59E+00	1.69E+00	1.26E+00	1.14E+00	8.01E-01	1.08E+00	9.37E-01	1.39E+00	1.17E+00	1.31E+00	6.73E-01	8.03E-01	1.21E+0−0
	Rank	9	13	14	10	6	2	5	4	12	7	11	**1**	3	8
	Best	1.01E+00	1.12E+00	1.21E+00	1.07E+00	7.89E-01	5.57E-01	7.59E-01	5.20E-01	9.92E-01	1.02E+00	1.11E+00	5.00E-01	5.36E-01	8.79E−01
	Rank2	9	13	14	11	6	4	5	2	8	10	12	**1**	3	7
F_4	*Mean*	−2.52E+01	−1.34E+01	−1.00E+01	−1.80E+01	−1.99E+01	−1.02E+01	−2.67E+01	−2.64E+01	−2.58E+01	−2.30E+01	−2.18E+01	−2.66E+01	−1.99E+01	−2.69E+01
	Rank	6	12	14	11	8	13	2	4	5	7	8	3	10	**1**
	Best	−2.84E+01	−1.47E+01	−1.42E+01	−2.73E+01	−2.72E+01	−2.75E+01	−2.84E+01	−2.84E+01	−2.84E+01	−2.53E+01	−2.84E+01	−2.84E+01	−2.84E+01	−2.84E+01
	Rank2	2	13	14	11	5	10	2	7	2	12	9	7	**1**	6
Avg.	*Rank*	4.375	13	10.875	8.75	5	5.125	5.5	6.625	5.375	7.625	7.75	3.375	6.25	3.75

Bold values signify the best results among all the algorithms.

named as *rank1* and *rank2* respectively, are also listed in Table 11. The average values of *rank1* and *rank2* are listed at the bottom of Table 11.

From the experimental results we can see that, although majority of peer algorithms yield at least one time of the global optimal solution on F_1, SaDE displays more favorable performance since it obtains the global optimal solution on all the 30 runs. Moreover, DMSPSO, PSODDS, SL_PSO, and DMPSO also manifest very promising performance on F_2 for their stable and accurate solutions. For the third application, CoDE achieves the best results both on the mean and the best results, followed by SL_PSO and CMA-ES. The comparison results on F_4 indicate that MSPSO offers the most favorable performance, in terms of the mean value. Although CMA-ES obtains the best result on F_4, in terms of the value of *Rank2* followed by SopPSO, HCLPSO, DMSPSO, MSPSO, CoDE, and SaDE, their is no significant different among the best solutions of them. From the average rank values we can see that SaDE offers the most favorable comprehensive performance, followed by MSPSO, DMSPSO, and PSODDS.

5. Conclusion

This paper presents a variant of PSO named as a multi-swarm particle swarm optimization based on dynamical topology and purposeful detecting (MSPSO). In MSPSO, the entire population is divided into many small-sized sub-swarms. Furthermore, three modules, i.e., DNS, SRS and PDS, are introduced to improve the comprehensive performance of MSPSO. The advantages of each proposed strategy are experimentally verified by extensive experiments.

To testify MSPSO's comprehensive performance, various experiments have been conducted to compare it with other 13 widely accepted EAs, including 9 PSOs, 3 DEs, and 1 ES, on CEC2013 test suite. In addition, 4 real applications are also adopted to verify the performance of the peer algorithms. From the experimental results, the following conclusions can be drawn. Firstly, DNS can offset the contradiction between the exploration and exploitation of PSO to some extent. Secondly, SRS also has a positive effect on particles information sharing behavior, which can improve the convergence speed for both unimodal and multimodal functions. Lastly, PDS is favorable for multimodal functions optimization since it can help population escape from a potential local optimum.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (Nos. 61663009, 61762036, 61320106006, 61532006, 61772207, 61763010, 61602174, 61562028), the National Natural Science Foundation of Jiangxi Province (Nos. 20171BAB202012, 20171BAB202019, 20161BAB202064, 20161BAB212052), the Research Project of Jiangxi Provincial Department of Communication and Transportation (No. 2017D0038), and the Project for Pearl River New Star in Science and Technology (No. 201506010047).

References

[1] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, Proc. of Int. Symp. on Micro Machine & Human Science, SMMHS'95 (1995) 39–43.
[2] J. Kennedy, R.C. Eberhart, Particle swarm optimization, Proc. of IEEE Int. Conf. on Neural Network, CNN'95 (1995) 1942–1948.
[3] H. Soh, Y.S. Ong, Q.C. Nguyen, Q.H. Nguyen, M.S. Habibullah, T. Hung, J.L. Kuo, Discovering unique, low-energy pure water isomers: memetic exploration, optimization and landscape analysis, IEEE Trans. Evol. Comput. 14 (3) (2010) 419–437.
[4] M. Shen, Z.H. Zhan, W.N. Chen, Y.J. Gong, J. Zhang, Y. Li, Bi-velocity discrete particle swarm optimization and its application to multicast routing problem in communication networks, IEEE Trans. Ind. Electron. 61 (12) (2014) 7141–7151.

[5] Z.H. Zhan, X.F. Liu, Y.J. Gong, J. Zhang, H.S.H. Chung, Y. Li, Cloud computing resource scheduling and a survey of its evolutionary approaches, ACM Comput. Surv. 47 (4) (2015) 1–33 (Article 63).
[6] M. Clerc, J. Kennedy, The particle swarm – explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (2) (2002) 58–73.
[7] V. Kadirkamanathan, K. Selvarajah, P.J. Fleming, Stability analysis of the particle dynamics in particle swarm optimizer, IEEE Trans. Evol. Comput. 10 (3) (2006) 245–255.
[8] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, Proc. of Congr. on Evol. Comput., CEC'98 (1998) 69–73.
[9] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evol. Comput. 8 (3) (2004) 240–255.
[10] J. Kennedy, R. Mendes, Population structure and particle swarm performance, Proc. of Congr. on Evol. Comput., CEC'02 (2002) 1671–1676.
[11] Q. Liu, W. Wei, H. Yuan, Z.H. Zhan, Y. Li, Topology selection for particle swarm optimization, Inf. Sci. 363 (2016) 154–173.
[12] A. Marco, d.O. Montes, S. Thomas, B. Mauro, D. Marco, Frankenstein's PSO: a composite particle swarm optimization algorithm, IEEE Trans. Evol. Comput. 13 (5) (2009) 1120–1132.
[13] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, Proc. of IEEE Swarm Intell. Symp., SIS'05 (2005) 124–129.
[14] S.Z. Zhao, J.J. Liang, P.N. Suganthan, M.F. Tasgetiren, Dynamic multi-swarm particle optimizer with local search for large scale global optimization, Proc. of Congr. on Evol. Comput., CEC'08 (2008) 3845–3852.
[15] Z.H. Zhan, J. Zhang, Y. Li, H.S. Chung, Adaptive particle swarm optimization, IEEE Trans. Syst. Man Cybern. B: Cybern. 39 (6) (2009) 1362–1381.
[16] M.R. Tanweer, S. Suresh, N. Sundararajan, Self regulating particle swarm optimization algorithm, Inf. Sci. 294 (10) (2015) 182–202.
[17] W. Lim, N.A.M. Isa, Particle swarm optimization with adaptive time-varying topology connectivity, Appl. Soft Comput. 24 (2014) 623–642.
[18] M. Črepinšek, S.H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: a survey, ACM Comput. Surv. 45 (3) (2013) 1–33.
[19] Y. Shi, R.C. Eberhart, Fuzzy adaptive particle swarm optimization, Proc. of Congr. on Evol. Comput., CEC'01 (2001) 101–106.
[20] C. Pornsing, M.S. Sodhi, B.F. Lamond, Novel self-adaptive particle swarm optimization methods, Soft Comput. 20 (2016) 3579–3593.
[21] L.M. Zhang, Y.G. Tang, C.C. Hua, X.P. Guan, A new particle swarm optimization algorithm with adaptive inertia weight based on Bayesian techniques, Appl. Soft Comput. 28 (2015) 138–149.
[22] Y. Juang, S. Tung, H. Chiu, Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions, Inf. Sci. 181 (20) (2011) 4539–4549.
[23] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evol. Comput. 8 (3) (2004) 204–210.
[24] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baska, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.
[25] L. Martino, V. Elvira, D. Luengo, J. Corander, F. Louzada, Orthogonal parallel MCMC methods for sampling and optimization, Digit. Signal Process. 58 (2016) 64–84.
[26] Z.H. Zhan, Y. Li, Y.H. Shi, Orthogonal learning particle swarm optimization, IEEE Trans. Evol. Comput. 15 (6) (2011) 832–847.
[27] S. Mo, J.C. Zeng, W.B. Xu, Attractive and repulsive fully informed particle swarm optimization based on the modified fitness model, Soft Comput. 20 (3) (2016) 863–884.
[28] C. Li, S. Yang, T.T. Nguyen, A self-learning particle swarm optimizer for global optimization problems, IEEE Trans. Syst. Man Cybern. B: Cybern. 42 (3) (2012) 627–646.
[29] Y.H. Li, Z.H. Zhan, S.J. Lin, J. Zhang, X.N. Luo, Competitive and cooperative particle swarm optimization with information sharing mechanism, Inf. Sci. 293 (3) (2015) 370–382.
[30] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inf. Sci. 291 (6) (2015) 43–60.
[31] J. Liu, D. Ma, T.B. Ma, W. Zhang, Ecosystem particle swarm optimization, Soft Comput. 21 (3) (2017) 1667–1691.
[32] N.J. Cheung, X.M. Ding, H.B. Shen, OptiFel: a convergent heterogeneous particle swarm optimization algorithm for Takagi–Sugeno fuzzy modeling, IEEE Trans. Fuzzy Syst. 22 (4) (2014) 919–933.
[33] G.J. Sun, R.Q. Zhao, Dynamic partition search algorithm for global optimization, Appl. Intell. 41 (4) (2014) 1108–1126.
[34] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm Evol. Comput. 24 (2015) 11–24.
[35] Z.H. Zhan, J. Li, J. Cao, J. Zhang, H. Chung, Y.H. Shi, Multiple populations for multiple objectives: a coevolutionary technique for solving multiobjective optimization problems, IEEE Trans. Cybern. 43 (2) (2013) 445–463.
[36] Y.J. Gong, J.J. Li, Y.C. Zhou, Y. Li, H.S.H. Chung, Y.H. Shi, J. Zhang, Genetic learning particle swarm optimization, IEEE Trans. Cybern. 46 (10) (2016) 2277–2290.
[37] H. Higashi, H. Iba, Particle swarm optimization with Gaussian mutation, Proc. of IEEE Swarm Intel. Symp., SIS'03 (2003) 72–79.
[38] B.Y. Qu, J.J. Liang, P.N. Suganthan, Niching particle swarm optimization with local search for multi-modal optimization, Inf. Sci. 197 (197) (2012) 131–143.
[39] H. Haklı, H. Uğuz, A novel particle swarm optimization algorithm with Levy flight, Appl. Soft Comput. 23 (2014) 333–345.
[40] X.W. Xia, J.N. Liu, Z.B. Hu, An improved particle swarm optimizer based on tabu detecting and local learning strategy in a shrunk search space, Appl. Soft Comput. 23 (5) (2014) 76–90.
[41] X.W. Xia, C.W. Xie, B. Wei, Z.B. Hu, B.J. Wang, C. Jin, Particle swarm optimization using multi-level adaptation and purposeful detection operators, Inf. Sci. 385 (2017) 174–195.
[42] X.W. Xia, B.J. Wang, C.W. Xie, Z.B. Hu, B. Wei, C. Jin, A sophisticated PSO based on multi-level adaptation and purposeful detection, Soft Comput. (2017), http://dx.doi.org/10.1007/s00500-017-2514-x.
[43] B. Xin, J. Chen, J. Zhang, H. Fang, Z.H. Peng, Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: a review and taxonomy, IEEE Trans. Syst. Man Cybern. C: Appl. Rev. 42 (5) (2012) 744–767.
[44] S. Sayah, A. Hamouda, A hybrid differential evolution algorithm based on particle swarm optimization for nonconvex economic dispatch problems, Appl. Soft Comput. 13 (4) (2013) 1608–1619.
[45] C.H. Xu, An efficient clustering method for mobile users based on hybrid PSO and ABC, Int. J. Innov. Comput. Appl. 6 (3) (2015) 163–170.
[46] M.S. Kiran, M. Gndz, A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems, Appl. Soft Comput. 13 (4) (2013) 2188–2203.
[47] E. Bengoetxea, EDA-PSO: a hybrid paradigm combining estimation of distribution algorithms and particle swarm optimization, Proc. of International Conference on Swarm Intelligence, vol. 6234 (2010) 416–423.
[48] C.J. Geyer, Markov Chain Monte Carlo maximum likelihood, Comput. Sci. Stat. 91 (8) (1992) 133–169.
[49] P.J. Green, K. Latuszynski, M. Pereyra, C.P. Robert, Bayesian computation: a summary of the current state, and samples backwards and forwards, Stat. Comput. 25 (4) (2015) 835–862.
[50] J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, Tech. Rep., Nanyang Technological Univ., Singapore, 2013.
[51] X. Jin, Y.Q. Liang, D.P. Tian, F.Z. Zhuang, Particle swarm optimization using dimension selection methods, Appl. Math. Comput. 219 (10) (2013) 5185–5197.
[52] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[53] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[54] Y. Wang, Z.Z. Cai, Q.F. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evol. Comput. 15 (1) (2011) 55–66.
[55] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001) 159–195.
[56] A. Auger, N. Hansen, Performance evaluation of an advanced local search evolutionary algorithm, Proc. of Congr. on Evol. Comput., CEC'05 (2005) 1777–1784.
[57] D. Swagatam, P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2011 competition on testing evolutionary algorithm on real world optimization problems, Tech. Rep., Nanyang Technological Univ., Singapore, 2010.