Copyright © 2022 Elsevier Ltd. All rights reserved. This is the accepted manuscript version of an article which has been published in final form at https://doi.org/10.1016/j.knosys.2022.108382, archived on this repository under a Creative Commons CC BY-NC-ND attribution licence.

A Particle Swarm Optimizer with Multi-level Population Sampling and Dynamic *p*-Learning Mechanisms for Large-scale Optimization

Mengmeng Sheng[a], Zidong Wang[b,c,*], Weibo Liu[c], Xi Wang[d], Shengyong Chen[e], Xiaohui Liu[c]

[a] *School of Computer Science and Technology, Zhejiang University of Science and Technology, Hangzhou, 300023, China*
[b] *College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao 266590, China.*
[c] *Department of Computer Science, Brunel University London, Uxbridge, Middlesex, UB8 3PH, United Kingdom*
[d] *Department of Computer Science, Hangzhou Normal University, Hangzhou, 311121, China*
[e] *School of Computer Science and Technology, Tianjin University of Technology, Tianjin, 300382, China*

Abstract

Large-scale optimization, which has received much attention in recent years, is inherently a challenging problem. This paper proposes a particle swarm optimizer with multi-level population sampling and dynamic *p*-learning mechanisms to address the problem. The multi-level sampling mechanism in the proposed method is developed for supporting a balanced evolutionary search. The mechanism works by partitioning the particles of swarm into multi-levels based on their fitness before each generation of evolution. A subset of swarm is then dynamically sampled from the particles at various levels for evolution such that encouraging exploration at the beginning of evolution while exploitation towards the end of evolution, thus appropriately searching the space. The dynamic *p*-learning mechanism, on the other hand, is introduced to allow efficient particle learning while preserving the swarm diversity during evolution. In this mechanism, each particle is devised to learn from one of the top $100p\%$ particles of the sub-swarm and the value of p associated with each particle is dynamically adjusted during evolution. By employing the above two mechanisms, the resulting method aims to appropriate search the solution space of large-scale global optimization problem for identifying the optimal or near-optimal solution. The performance of the proposed method has been evaluated on CEC'2010 and CEC'2013 benchmark suites for large-scale optimization and compared with related methods. Our results confirm the merits of the devised mechanisms in the proposed method. The results also show that our method can achieve a superior performance and outperform related methods.

Keywords: Particle swarm optimization, large-scale optimization, population sampling, particle learning strategy

[*]Corresponding author: Zidong Wang
Email address: zidong.wang@brunel.ac.uk

1. Introduction

Particle swarm optimization (PSO) [1], which simulates swarm behaviors of birds flocking, is a popular global optimization scheme. In PSO, each particle of swarm has a historical best position named *pBest*, which records the best position it has searched, and the best of these *pBests* is called *gBest*. Each particle is guided by its own *pBest* and the *gBest* to search for the optimal position in solution space. This scheme has shown to be efficient and successfully applied in various fields [2], [3], [4], [5], [6], [7], [8]. However, it could perform poorly when the optimization problem to be addressed involves a large number of local optima and has a high-dimensionality, generally referred to as large-scale global optimization (LSGO) [9]. This issue is mainly attributed to the premature convergence exerted by the *pBest* and *gBest* based particle learning in PSO as well as its limited capability to achieve a balanced evolutionary search [2], [10].

To improve the performance of PSO, many variants of the algorithm have been proposed [2], [3], [16], [17], [18]. Kennedy and Mendes [23] argued that the *gBest* based particle learning strategy as adopted in the traditional PSO could cause a rapid loss of swarm diversity, thus leading to premature convergence. Instead of *gBest*, the authors proposed to update each particle during evolution using the information of local best (*lbest*), which is defined as the best of *pBest* of the particle's neighborhoods determined by a given topology. Such a strategy allows the particle's learning to be influenced by its neighborhoods and can be used to preserve the swarm diversity to a certain extent. This scheme represents the first approach, which employs *lbest* rather than *gBest* for particle learning, to enhance the traditional PSO and is followed by several researchers to design their PSO variants [23], [24], [33]. The second approach tends to get rid of both *pBest* and *gBest* and introduce inter-particle learning strategies to implement PSO. For example, Cheng *et al.* [11] proposed a variant of PSO based on an inter-particle learning strategy, which works by first randomly selecting a pair of particles from the swarm. Then, the one with a higher fitness (i.e., the winner) will directly enter into the swarm for evolution, while the other one will learn from the winner before it can enter into the swarm. Yang *et al.* [14] introduced another inter-particle learning strategy, in which particles are first divided into multiple levels according to their fitness values. Each particle then learns from two predominant particles, which are from different higher levels. Comparing with *lbest* based strategies, inter-particle learning strategies are generally able to preserve more appropriate swarm diversity. However, they could lead to inefficient particle learning, as elite individuals are usually not considered for particle learning. Further, in existing particle learning strategies, fixed rules are typically employed to update all particles in the swarm. Since different particles possess different properties, it would be desirable to have a particle learning strategy with flexible rules, which can consider different properties of the particles. Additionally, the above PSO variants generally tend to enhance the traditional PSO by improving the particle learning strategy alone. As the particle learning strategy could have a limited capability to help PSO achieving a balanced evolutionary search, this may also restrict the performance of these methods.

To address above issues, here we first devise a multi-level population sampling mechanism to support a balanced evolutionary search. The mechanism tries to divide the particles of swarm into multi-levels based on their fitness before each generation of evolution. A subset of swarm is then dynamically sampled from the particles at various levels for evolution such that encouraging exploration at the beginning of evolution while exploitation towards the end of evolution, thus appropriately searching the space. Further, a dynamic p-learning mechanism is introduced to allow efficient particle learning while preserving the swarm diversity during evolution. In this mechanism, each particle is devised to learn from one of the top $100p\%$ particles of the sub-swarm and the value of p associated with each particle is dynamically adjusted during evolution. By incorporating these two mechanisms into PSO, a swarm optimizer with multi-level population sampling and dynamic p-learning mechanisms is thus proposed. We evaluate the proposed algorithm on CEC'2010 and CEC'2013 benchmark suites for LSGO and compare its performance with related methods. The results show that the proposed method is well suited to address LSGO and outperforms related methods. The results also confirm the significance of the devised mechanisms in our method.

The remainder of the paper is organized as follows. Following a brief review of related work in Section 2, we present the proposed method in Section 3. Subsequently, in Section 4, a series of experiments on CEC'2010 and CEC'2013 LSGO benchmark suites are conducted to evaluate the performance of proposed method and to compare with related methods. Finally, we conclude the paper with a summary in Section 5.

2. Related Work

2.1 Canonical PSO

Canonical PSO, proposed by Kennedy *et al.* [1], is a swarm-based stochastic optimization algorithm, which starts with N randomly initialized particles. For an D-dimensional optimization problem, each particle in the swarm maintains a velocity vector $V_i=\{v_{i,1}, v_{i,2}, ..., v_{i,D}\}$, a position vector $X_i=\{x_{i,1}, x_{i,2}, ..., x_{i,D}\}$ and a historical best position $pBest_i=\{pBest_{i,1}, pBest_{i,2}, ..., pBest_{i,D}\}$, where i=1, 2, ..., N. Among these *pBest*s, the best one is called *gBest*.

At each generation, each particle i updates the velocity and position according to its own $pBest_i$ and *gBest*. The updating rules are defined as:

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

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

where j=1, 2, ..., D represents the j^{th} dimension of the optimization problem, w is an inertia weight, c_1 and c_2 denote acceleration coefficients, r_1 and r_2 are random numbers uniformly distributed in [0, 1]. After updating the position of the particle, if $pBest_i$ is worse than the current position X_i, then it will be replaced by X_i. Due to its simplicity and efficiency, PSO has been widely applied to deal with optimization problems [3]. However, it may not perform well on optimization problems, which involve complex search spaces. This is mainly due to it suffers from premature convergence and has a limited capability to achieve a balanced evolutionary search [2], [10].

2.2 PSO Variants

To improve the performance of canonical PSO, many variants of the algorithm have been developed in literature [2], [3], [25], [26]. In [23], Kennedy and Mendes proposed a PSO variant, in which, rather than *gBest*, *lbest* is employed for particle learning. Specifically, in this method, the velocity of the particle is updated as:

$$v_{i,j} = w \cdot v_{i,j} + c_1 \cdot r_{1,j} \cdot (pBest_{i,j} - x_{i,j}) + c_2 \cdot r_{2,j} \cdot (lBest_{i,j} - x_{i,j}), \quad (3)$$

where $lBest_i$ is the best *pBest* of i^{th} particle's neighborhoods defined by a given topology. In [24], Liang *et al.* extended the above scheme by allowing each particle to learn from different *lBests* on different dimensions of the data. Specifically, the learning rule in this method is defined as:

$$v_{i,j} = w \cdot v_{i,j} + c \cdot r_j \cdot (pBest_{f_i(j),j} - x_{i,j}), \quad (4)$$

where $f_i(j)$ is the winner's *pBest* of two randomly selected particles. In [33], Liang *et al.* proposed a dynamic multi-swarm PSO, in which the particles are first randomly divided into multiple groups and each group then evolves using the *lbest* based learning strategy given in [23]. By employing *lbest* rather than *gBest* for particle learning, the above methods can avoid rapid loss of swarm diversity during evolution and have shown to be more effective than the canonical PSO. However, their performance is still limited on optimization problems with complex search spaces [2].

To improve the situation further, inter-particle learning strategies, in which neither *pBest* nor *gBest* is employed, have also been proposed to implement PSO. For example, in [11], Cheng *et al.* devised a competitive swarm optimizer (CSO), in which the updating of particles is driven by a pairwise random competition between particles. After each competition, the winner will directly pass to the swarm of next generation, while the loser X_l be updated according to the information from the winner using the following rules:

$$v_{l,j} = r_{1,j} \cdot v_{l,j} + r_{2,j} \cdot \left(x_{w,j} - x_{l,j}\right) + \varphi \cdot r_{3,j} \cdot \left(\bar{x}_j - x_{l,j}\right), \tag{5}$$

$$x_{l,j} = x_{l,j} + v_{l,j}, \tag{6}$$

where r_1, r_2 and r_3 are random numbers uniformly distributed within [0, 1], $\bar{x}$ is the mean position of all particles in the swarm, φ is a control parameter. In [12], a social learning PSO (SL-PSO) was developed, in which each particle X_i is set to learn from a randomly selected particle X_k in the swarm, which has a better fitness. In this method, the updating rule of velocity is defined as:

$$v_{l,j} = r_{1,j} \cdot v_{l,j} + r_{2,j} \cdot \left(x_{k,j} - x_{l,j}\right) + \varphi \cdot r_{3,j} \cdot \left(\bar{x}_j - x_{l,j}\right). \tag{7}$$

Yang *et al.* proposed a segment-based predominant learning swarm optimizer (SPLSO) [13] and a level-based learning swarm optimizer (LLSO) [14]. In SPLSO, the variables of particles are randomly divided into multiple segments and different segments learn from different predominant particles. While in LLSO, particles are divided into multiple levels according to their fitness values. Each particle learns from two predominant particles, which are from different higher levels.

The above methods are generally able to outperform the *lbest* based PSO variants as well as traditional cooperative coevolutionary algorithms (CCEAs) [13], [14] for addressing LSGO. However, in these methods, elite individuals are usually not considered during particle learning. This could lead to an inefficient particle learning, thus restricting the performance of the algorithm. Further, the particle learning strategies in existing PSO methods typically tend to update all particles using the same rules. As different particles possess different properties, a flexible learning rule, which can take into account such information, may be preferred in order to achieve a good performance. Additionally, to enhance the traditional PSO, the above methods generally focus on improving the particle learning strategy alone, which could have a limited capability to help PSO achieving a balanced evolutionary search. To address the above issues, this work first devises a multi-level population sampling mechanism, which is employed to encourage exploration at the beginning of evolution while exploitation towards the end of evolution, thus achieving a balanced evolutionary search. Further, we consider different properties of the particles and introduce a flexible particle learning mechanism to allow efficient particle learning while preserving the swarm diversity during evolution.

2.3 Other Related Work

The LSGO problem could be dealt using the divide-and-conquer strategy, which decomposes the problem into multiple sub-problems and solves them separately [29], [30], [31], [32]. The best partial solutions obtained for these sub-problems are then assembled together to form a full solution. This approach is generally referred to as the decomposition-based approach. Employing this strategy, PSO based methods have also been proposed in literature [15], [21], [27], [28]. These methods generally adopt the cooperative coevolutionary (CC) framework to evolve multiple swarms, each of which is used to encode one partial solution of the problem. For example, Bergh *et al.* [27] proposed a CC based PSO called CPSO-S_k for LSGO. In this method, sub-problems are formed by dividing variables of a given problem into k groups and each of which is dealt with a swarm. In [21], Li and Yao devised another CC based PSO named CCPSO2 for LSGO, in which the sizes of variables of sub-problems are determined dynamically during evolution. These methods are promising for LSGO problems. However, how to make swarms work cooperatively to deliver partial solutions, which can be used to form an optimal or near optimal full solution, is typically a difficult problem [22]. Further, it has been shown that CCEAs may not perform well on non-separable problems with more than 100 real-valued variables [19].

Apart from PSO, many other evolutionary algorithms (EAs) have also been proposed to handle LSGO [37], [38]. For example, Molina *et al.* [34] proposed a memetic algorithm named MA-SW-Chains, which combines a steady-state genetic algorithm with a local search method, for LSGO. LaTorre *et al.* developed a multiple offspring sampling framework [35], [36], which hybridizes multiple EAs to handle LSGO. Yang *et al.* [19] developed a CC based differential evolution (DE) for LSGO, in which a random grouping scheme and adaptive weighting are introduced for problem decomposition. Zhang *et al.* [20] introduced a multilevel cooperative coevolution, which is incorporated into DE for LSGO. While, Omidvar *et al.* [29] devised a CC based DE with an automatic decomposition scheme, which tries to decompose the problem into sub-problems with minimum interdependences. In this work, the performance of several representative algorithms of above approach will be compared with our proposed method.

3. Proposed method

This section presents a PSO with multi-level population sampling and dynamic p-learning mechanisms (denoted as *mlsdpl*_PSO) for LSGO. In the proposed method, a multi-level sampling mechanism is developed and incorporated to dynamically sample a sub-swarm before each generation of evolution. The sampled subset subsequently evolves using the devised dynamic p-learning mechanism based swarm optimizer. The above process will

Step 1. Initialize a swarm *P* with *N* particles.

Step 2. Employ the proposed multi-level sampling mechanism (see Section 3.1) to sample a sub-swarm *SP* from *P*.

Step 3. Sort the particles of *SP* in ascending order according to their fitness values.

Step 4. For each particle *m* in *SP*, perform the dynamic *p*-learning mechanism (see Section 3.2) as follows:

  i. Randomly select a particle *e* from the top p_m (p_m is associated with the particle *m*) percent of particles in *SP*.

  ii. If *e* has a better fitness than *m*, then:

    a) Update the velocity of *m* according to equation (9).

    b) Update the position of *m* according to equation (10).

    c) If the updated *m* has a better fitness than the original *m*, then update p_m according to equation (13), otherwise update it according to equation (14).

Step 5. Go to Step 2 if the maximum number of function evaluations is not reached. Otherwise, terminate the evolution.

Step 6. Output the solution with the best fitness.

Algorithm 1. A PSO with multi-level population sampling and dynamic *p*-learning mechanisms for LSGO.

repeat until a termination condition is met. The procedure of the proposed algorithm is shown in Algorithm 1.

In the following sections, we shall describe the details of multi-level sampling and dynamic *p*-learning mechanisms in the proposed method.

3.1 Multi-level Sampling Mechanism

The multi-level sampling (MLS) mechanism is devised for supporting a balanced PSO evolution. In the devised mechanism, before each generation of evolution, particles of swarm are first partitioned into multi-levels based on their fitness. Then, a subset of swarm is dynamically selected from the particles at various levels such that encouraging exploration during the early phase of evolution while exploitation towards the end of evolution. The primary rationale behind this mechanism is that, during evolution, the exploitation and exploration behaviors of a swarm optimizer depend on the particles subjected to evolve. If most of the particles subjected to evolve possess relative low fitness values, then the swarm optimizer will be biased to explore the solution space. Otherwise, it will be biased for exploitation. In order to achieve a proper performance, generally, the task of evolutionary search should focalize more on exploration during the early phase of evolution, thus discovering promising areas of the space. While along with the progress of evolution, the evolutionary search should gradually switch to exploitation, therefore locating the optimal solution with high accuracy. According to the above rationale, dynamically sampling an appropriate subset of swarm for evolution at each generation is thus desirable in order to properly implement the evolutionary search.

Specifically, the devised MLS mechanism works as follow. At the beginning of each generation, the entire particles in the swarm are firstly sorted based on their fitness values in descending order. The sorted particles are then partitioned evenly to L levels, indexed 0 to L-1, such that a higher level (associated with a smaller index) will contain particles of higher fitness. Subsequently, we calculate the sampling probability pr_i for the particles at level i as:

$$pr_i = pr_{i,initial} + (pr_{i,final} - pr_{i,initial}) \cdot \frac{FEs}{FE\_Max}, \tag{8}$$

where FEs denotes the number of function evaluations consumed so far during evolution and FE_Max is a user-specified maximum number of function evaluations. The initial and final sampling probability $pr_{i,\mathrm{initial}}$ and $pr_{i,final}$ are defined as $pr_{i,\mathrm{initial}}=i/(L\text{-}1)$ and $pr_{i,final}=1\text{-}pr_{i,\mathrm{initial}}$, respectively. According to the calculated probabilities, a sub-swarm is finally sampled from particles at different levels. Based on the above procedure, during the early stage of evolution, particles from lower levels will have higher probabilities to be selected for evolution, therefore encouraging exploration aspect of evolution to identify potential regions of the space. While, during the later stage of evolution, particles from higher levels will have higher probabilities to be sampled, thus encouraging exploitation of the space to locate the optimum with high accuracy.

3.2 Dynamic p-Learning Mechanism

To appropriately search the space, it is also desirable to have a particle learning scheme, which can support efficient evolutionary search while preserving the swarm diversity during evolution. Here, we propose a dynamic p-learning mechanism (DPL) for this purpose. The DPL will be implemented on the sub-swarm selected by the MLS mechanism at each generation. In the proposed DPL, each particle is devised to learn from one of the top $100p\%$ particles of the sub-swarm and the value of p associated with each particle is dynamically adjusted during evolution.

Specifically, to update a particle m during particle learning, the proposed mechanism works as follows. Firstly, the p_m value associated with particle m is extracted to determine top $100p_m\%$ particles of the sub-swarm. Then, one particle e will be randomly selected from these top particles. If the selected particle e has a better fitness than m, then updating m according to the following rules:

$$v_{m,j} = r_{1,j} \cdot v_{m,j} + r_{2,j} \cdot \left(x_{e,j} - x_{m,j}\right) + \varphi \cdot r_{3,j} \cdot \left(\bar{x}_{weight,j} - x_{m,j}\right), \tag{9}$$

$$x_{m,j} = x_{m,j} + v_{m,j}, \tag{10}$$

where x_e is the position of particle e. It should be noted that, rather than the centroid of swarm, as typically used to define the updating rule, a weighted centroid has been adopted here to increase the efficiency of particle learning. Specifically, we define the term $\bar{x}_{weight}$ in equation (9), which denotes the weighted centroid of particles of the sub-swarm, as:

$$\bar{x}_{weight,j} = \sum_{i=1}^{|SP|} f_i \cdot x_{i,j} / \sum_{i=1}^{|SP|} f_i, \tag{11}$$

where f_i is the fitness of i^{th} particle and $|SP|$ denotes the number of individuals in the sub-warm SP.

Obviously, the setting of parameter p for each particle is critical for the performance of the proposed mechanism. A small value of p will lead the particle to learn from very top particles, thus resulting efficient particle learning and promoting exploitation of the evolutionary search. Increasing the value of p will allow it to learn from particles with relatively low fitness thus encouraging a diverse search and preserving the population diversity. To support an efficiently particle learning while preserving the swarm diversity during evolution, the following scheme has been introduced to dynamically control the value of p for each particle. Firstly, the parameter p_i associated with i^{th} particle in the swarm is initialized as:

$$p_i = \frac{(1-p_{min})(f_{max}-f_i)}{f_{max}-f_{min}} + p_{min}, \tag{12}$$

where f_i is the fitness of i^{th} particle, f_{max} and f_{min} denotes the maximum and minimum fitness of particles in the initial swarm, p_{min} is a user-specified minimum value of p and is set to be 0.05. As a result, a particle with a high fitness will be assigned with a small value of p to encourage it for exploitation during particle learning. Otherwise, a high value of p will be assigned to encourage it for a diverse search. Further, during the process of particle learning, the value p of each particle is set to learn from its paired particle. Specifically, for a particle m to be learned from the particle e, after updating its position, if m is improved, then its associated parameter p_m is updated as:

$$p_m = (p_m + p_e)/2, \tag{13}$$

Otherwise, it will be computed as:

$$p_m = 2 \cdot p_m - p_e, \tag{14}$$

where p_e is the parameter p associated with particle e. The resulting p_m will be truncated into [p_{min}, 1]. According to the above rules, if particle e has a guiding effect on m, then the value of p_m will be set close to value of p_e, which is typically smaller than p_m, to increase its probability for exploitation during particle learning. Otherwise, the value of p_m will be set far away from value of p_e to increase its probability for exploration during particle learning.

By employing the above scheme, particles with high fitness will tend to have a small value of p and could be used to efficiently exploit the search space during particle learning. While for particles with low fitness, they generally carry with a high value of p and could be used to promote a diverse search. By employing such a scheme, the resulting particle learning mechanism is thus able to support an efficient evolutionary search while preserving the swarm diversity during evolution.

4. Experiments

In this section, experiments have been carried out to evaluate the devised MLS and DPL mechanisms, and to compare our proposed method with related algorithms. All algorithms used in the experiments are coded using C++ and tested on a workstation with an Intel (R) Core™ i7-3630QM CPU at 2.40GHz and 8 GB RAM running Windows™ 7 operation system. Unless otherwise stated, 30 trials of each algorithm are performed and the average results of which are reported.

4.1 Data Sets and Parameter Settings

The CEC'2010 and CEC'2013 benchmark suits for LSGO, which contain 20 (denoted as F1 to F20) and 15 functions (denoted as M1 to M15), respectively, have been used for experiments. The characteristics of these two sets of functions can be found in [39] and [40], respectively. In our experiments, the dimension D of these functions is set to be 1000 and the maximum number of function evaluations (*FE_Max*) is set to be 3000×D. For the swarm size of our proposed method, it is configured as 2*(100+D/10.0). The level number L in the devised MLS mechanism and the control parameter φ in the particle updating rule are set to be 20 and D/100*0.01, respectively.

TABLE I. Comparing the Results Delivered By *mlsdpl*_PSO and its Variant *dpl*-PSO in Term of Mean Fitness of the Best Solutions over Thirty Trials.

Method	F1	F2	F3	F4	F5	F6	F7	F8	F9
*mlsdpl*_PSO	1.98e-19	**6.07e02**	**5.12e-13**	**1.51e11**	**3.58e06**	**5.22e-08**	1.09e01	**6.48e04**	**1.94e07**
*dpl*_PSO	**2.95e-20**	1.16e03	3.61e-02	3.32e11	1.05e07	8.22e-02	**3.15e-01**	2.17e07	3.40e07
	F10	F11	F12	F13	F14	F15	F16	F17	F18
*mlsdpl*_PSO	**6.60e02**	**9.66e-12**	**3.23e03**	**4.35e02**	**5.81e07**	**8.82e02**	**7.31e-12**	**5.17e04**	**1.18e03**
*dpl*_PSO	9.39e02	2.00e00	1.03e04	7.21e02	9.74e07	9.69e03	9.85e00	1.12e05	2.25e03
	F19	F20	M1	M2	M3	M4	M5	M6	M7
*mlsdpl*_PSO	**2.45e06**	**1.17e03**	6.92e-19	**6.91e02**	**2.16e01**	**1.64e09**	**6.17e05**	**1.06e06**	**8.01e04**
*dpl*_PSO	4.74e06	1.85e03	**3.17e-20**	1.30e03	**2.16e01**	5.25e09	7.12e05	**1.06e06**	7.41e05
	M8	M9	M10	M11	M12	M13	M14	M15	Null
*mlsdpl*_PSO	**4.45e13**	**9.16e07**	**9.14e07**	**9.28e11**	**1.16e03**	**1.77e07**	**2.13e07**	**6.17e06**	Null
*dpl*_PSO	9.21e13	1.56e08	9.33e07	9.30e11	1.89e03	5.39e08	1.94e08	1.13e07	Null
w/l/t	30/3/2								

TABL II. Comparing the Results Delivered By *dpl*-PSO, CSO and SL-PSO in Term of Mean Fitness of the Best Solutions over Thirty Trials.

Method	F1	F2	F3	F4	F5	F6	F7	F8	F9
*dpl*_PSO	**2.95e-20**	**1.16e03**	3.61e-02	3.32e11	1.05e07	8.22e-02	**3.15e-01**	2.17e07	3.40e07
CSO	4.50e-12	7.42e03	**2.60e-09**	7.25e11	**2.86e06**	**8.21e-07**	2.01e04	3.87e07	7.03e07
SL-PSO	8.73e-18	1.93e03	1.88e00	**2.99e11**	3.17e07	2.08e01	6.49e04	**7.81e06**	**3.30e07**
	F10	F11	F12	F13	F14	F15	F16	F17	F18
*dpl*_PSO	**9.39e02**	**2.00e00**	**1.03e04**	7.21e02	9.74e07	**9.69e03**	9.85e00	1.12e05	2.25e03
CSO	9.60e03	4.02e-08	4.37e05	**6.29e02**	2.49e08	1.01e04	**5.89e-08**	2.20e06	**1.73e03**
SL-PSO	2.56e03	2.32e01	1.75e04	9.59e02	**8.41e07**	1.12e04	2.51e01	**9.00e04**	2.77e03
	F19	F20	M1	M2	M3	M4	M5	M6	M7
*dpl*_PSO	**4.74e06**	1.85e03	**3.17e-20**	**1.30e03**	**2.16e01**	5.25e09	7.12e05	**1.06e06**	**7.41e05**
CSO	1.01e07	**1.05e03**	7.71e-12	8.55e03	**2.16e01**	1.32e10	**5.91e05**	**1.06e06**	5.88e06
SL-PSO	5.10e06	1.85e03	1.09e-17	2.13e03	**2.16e01**	**4.35e09**	8.41e05	**1.06e06**	1.63e06
	M8	M9	M10	M11	M12	M13	M14	M15	Null
*dpl*_PSO	**9.21e13**	1.56e08	9.33e07	**9.30e11**	1.89e03	5.39e08	**1.94e08**	**1.13e07**	Null
CSO	2.60e14	**6.06e07**	9.40e07	**9.30e11**	**1.07e03**	6.67e08	3.62e09	7.87e07	Null
SL-PSO	1.03e14	8.25e07	**9.25e07**	9.33e11	1.78e03	**4.65e08**	3.28e08	5.86e07	Null
w/l/t (*dpl*_PSO vs. CSO)	21/11/3								
w/l/t (*dpl*_PSO vs. SL-PSO)	22/10/3								

4.2 Exploring the Proposed Mechanisms

We first assess the significance of MLS and DPL mechanisms in the proposed algorithm. For this purpose, two sets of experiments have been carried out. In the first set of experiments, we compare the performance of our proposed algorithm, *mlsdpl*_PSO, with its variant: *mlsdpl*_PSO without MLS mechanism (denoted as *dpl*-PSO). In the second set of experiments, we compare the performance of *dpl*-PSO, in which the DPL mechanism is used for particle learning, with two recently proposed methods, CSO [11] and SL-PSO [12], which employ different inter-particle learning mechanisms. The results of the two sets of experiments are shown in Tables I and II, respectively. The last rows of Tables I and II report a summary in terms of the number of *wins*, *losses* and *ties* on the test functions of the pairwise comparisons.

Comparing *mlsdpl*_PSO with *dpl*-PSO, the results show that the MLS mechanism could greatly improve the performance of *mlsdpl*_PSO. Specifically, the results in Table I show that, by incorporating the MLS mechanism, *mlsdpl*_PSO is able to locate better or comparable

TABLE III. Comparing the Results Delivered by Different Methods on CEC'2010 Test Suit in Terms of Mean Fitness and Standard Deviation Along with Two Tailed *t*-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	SPLSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
F1	Mean	1.98e-19	4.50e-12	8.73e-18	7.73e-20	**3.13e-22**	9.75e-20	1.88e04	2.96e00	8.65e-13	3.54e-07
	Std	1.39e-19	5.94e-13	3.30e-18	7.07e-21	8.03e-23	3.33e-19	4.66e04	6.68e00	2.97e-12	1.44e-07
	t-test	-	**4.15e01***	**1.41e01***	-4.75e00	-7.79e00	-1.53e00	**2.21e00***	**2.43e00***	1.60e00	**1.35e01***
F2	Mean	6.07e02	7.42e03	1.93e03	4.45e02	9.82e02	5.74e02	4.43e03	4.30e00	**2.89e00**	1.33e03
	Std	2.51e01	2.86e02	1.12e02	1.65e01	4.39e01	1.41e02	1.87e02	1.11e00	1.52e00	2.55e01
	t-test	-	**1.30e02***	**6.31e01***	-2.95e01	**4.06e01***	-1.26e00	**1.11e02***	-1.31e02	-1.32e02	**1.11e02***
F3	Mean	5.12e-13	2.60e-09	1.88e00	2.52e-13	**2.76e-14**	1.12e-12	1.66e01	4.51e-03	2.10e-07	1.10e00
	Std	8.24e-14	2.62e-10	3.30e-01	1.89e-14	2.38e-15	5.73e-13	3.02e-01	1.66e-03	1.12e-06	3.35e-01
	t-test	-	**5.43e01***	**3.12e01***	-1.68e01	-3.22e01	**5.75e00***	**3.01e02***	**1.49e01***	1.03e00	**1.80e01***
F4	Mean	**1.51e11**	7.25e11	2.99e11	4.30e11	4.40e11	2.74e11	5.22e12	1.70e12	1.71e13	2.59e13
	Std	2.99e10	1.23e11	7.16e10	8.31e10	1.10e11	7.24e10	1.89e12	1.04e12	5.47e12	8.14e12
	t-test	-	**2.48e01***	**1.04e01***	**1.73e01***	**1.39e01***	**8.60e00***	**1.47e01***	**8.15e00***	**1.70e01***	**1.73e01***
F5	Mean	3.58e06	**2.86e06**	3.17e07	6.30e06	1.22e07	3.42e07	1.55e08	4.14e08	4.99e08	2.69e08
	Std	1.28e06	1.79e06	6.21e06	1.76e06	3.43e06	6.69e06	2.15e07	1.38e08	1.07e08	6.84e07
	t-test	-	-1.79e00	**2.43e01***	**6.85e00***	**1.29e01***	**2.46e01***	**3.85e01***	**1.63e01***	**2.54e01***	**2.13e01***
F6	Mean	5.22e-08	8.21e-07	2.08e01	**9.45e-09**	5.20e-01	1.41e05	1.63e01	1.71e07	1.78e07	5.00e06
	Std	1.39e-08	2.68e-08	2.63e00	1.20e-09	7.46e-01	3.67e05	3.45e-01	5.20e06	4.37e06	1.03e06
	t-test	-	**1.39e02***	**4.33e01***	-1.68e01	**3.82e00***	**2.10e00***	**2.59e02***	**1.80e01***	**2.23e01***	**2.66e01***
F7	Mean	1.09e01	2.01e04	6.49e04	4.76e02	7.19e02	**1.04e01**	1.41e04	2.06e08	1.51e08	8.14e08
	Std	3.83e01	3.86e03	5.60e04	1.31e02	2.59e03	4.02e00	1.26e04	4.31e08	1.45e08	5.40e08
	t-test	-	**2.85e01***	**6.35e00***	**1.87e01***	1.50e00	-7.11e-02	**6.12e00***	**2.62e00***	**5.70e00***	**8.26e00***
F8	Mean	**6.48e04**	3.87e07	7.81e06	3.11e07	2.34e07	1.15e07	2.75e07	4.13e07	6.59e07	8.56e07
	Std	5.71e03	6.81e04	1.56e06	9.36e04	2.46e05	2.04e07	2.63e07	3.84e07	3.40e07	2.64e07
	t-test	-	**3.10e03***	**2.72e01***	**1.81e03***	**5.19e02***	**3.07e00***	**5.71e00***	**5.88e00***	**1.06e01**	**1.77e01***
F9	Mean	**1.94e07**	7.03e07	3.30e07	4.59e07	4.36e07	3.07e07	5.59e07	1.02e08	2.43e08	4.40e08
	Std	1.45e06	5.73e06	4.46e06	3.04e06	4.28e06	3.19e06	6.45e06	3.30e07	2.16e07	4.87e07
	t-test	-	**4.72e01***	**1.59e01***	**4.31e01***	**2.93e01***	**1.77e01***	**3.02e01***	**1.37e01***	**5.66e01***	**4.73e01***
F10	Mean	**6.60e02**	9.60e03	2.56e03	7.99e03	8.91e02	1.33e03	4.49e03	5.09e03	4.24e03	1.03e04
	Std	2.89e01	7.67e01	2.17e02	1.28e02	3.66e01	5.67e01	1.29e02	7.81e02	1.45e03	3.13e02
	t-test	-	**5.97e02***	**4.75e01***	**3.06e02***	**2.71e01***	**5.77e01***	**1.59e02***	**3.10e01***	**1.35e01***	**1.68e02***
F11	Mean	9.66e-12	4.02e-08	2.32e01	**3.04e-12**	5.80e00	8.66e00	1.02e01	1.98e02	1.98e02	2.59e01
	Std	7.30e-12	5.12e-09	2.10e00	2.89e-13	5.40e00	3.25e00	8.71e-01	2.12e00	1.12e00	1.73e00
	t-test	-	**4.30e01***	**6.05e01***	-4.96e00	**5.88e00***	**1.46e01***	**6.41e01***	**5.12e02***	**9.68e02***	**8.20e01***
F12	Mean	3.23e03	4.37e05	1.75e04	9.52e04	1.25e04	6.34e04	**2.84e03**	3.39e04	1.03e05	9.55e04
	Std	5.46e02	6.22e04	9.07e03	6.69e03	1.46e03	1.00e04	1.08e03	1.19e04	1.57e04	9.55e03
	t-test	-	**3.82e01***	**8.60e00***	**7.50e01***	**3.26e01***	**3.29e01***	-1.77e00	**1.41e01***	**3.48e01***	**5.28e01***
F13	Mean	**4.35e02**	6.29e02	9.59e02	5.48e02	7.35e02	9.89e02	6.27e03	1.34e03	4.22e03	5.96e03
	Std	6.40e01	2.32e02	3.74e02	1.69e02	1.93e02	4.52e02	3.65e03	1.72e02	4.70e03	4.16e03
	t-test	-	**4.42e00***	**7.56e00***	**3.42e00***	**8.08e00***	**6.65e00***	**8.75e00***	**2.70e01***	**4.41e00***	**7.27e00***
F14	Mean	**5.81e07**	2.49e08	8.41e07	1.60e08	1.24e08	1.70e08	3.42e08	3.06e08	5.70e08	9.78e08
	Std	2.40e06	1.53e07	6.31e06	8.50e06	7.38e06	1.29e07	2.42e07	1.19e08	5.50e07	7.52e07
	t-test	-	**6.75e01***	**2.11e01***	**6.32e01***	**4.65e01***	**4.67e01***	**6.39e01***	**1.14e01***	**5.09e01***	**6.70e01***
F15	Mean	8.82e02	1.01e04	1.12e04	9.91e03	**8.33e02**	2.66e03	5.86e03	1.08e04	8.90e03	1.23e04
	Std	5.97e01	5.23e01	8.65e01	6.70e01	4.31e01	1.50e02	1.05e02	1.35e03	2.07e03	8.24e02
	t-test	-	**6.36e02***	**5.38e02***	**5.51e02***	-3.64e00	**6.03e01***	**2.26e02***	**4.02e01***	**2.12e01***	**7.57e01***
F16	Mean	7.31e-12	5.89e-08	2.51e01	4.68e-12	4.25e00	5.94e01	**7.53e-13**	3.96e02	3.81e02	6.96e01
	Std	1.09e-11	5.61e-09	1.16e01	4.49e-13	2.41e00	1.37e01	6.25e-14	5.73e-01	5.76e01	6.43e00
	t-test	-	**5.75e01***	**1.19e01***	-1.32e00	**9.66e00***	**2.37e01***	-3.29e00	**3.79e03***	**3.62e01***	**5.93e01***
F17	Mean	5.17e04	2.20e06	9.00e04	6.84e05	9.05e04	1.07e05	**4.03e04**	1.25e05	3.49e05	3.11e05
	Std	4.32e03	1.55e05	1.58e04	3.63e04	3.53e03	6.75e04	2.29e03	5.25e04	3.11e04	2.24e04
	t-test	-	**7.59e01***	**1.28e01***	**9.47e01***	**3.81e01***	**4.48e00***	-1.28e01	**7.62e00***	**5.19e01***	**6.23e01***
F18	Mean	**1.18e03**	1.73e03	2.77e03	1.35e03	2.55e03	2.53e03	1.47e10	3.07e03	1.81e04	3.83e04
	Std	1.52e02	5.22e02	8.33e02	3.87e02	8.32e02	6.86e02	2.03e09	2.45e02	9.48e03	1.53e04
	t-test	-	**5.54e00***	**1.03e01***	**2.24e00***	**8.87e00***	**1.05e01***	**3.97e01***	**3.59e01***	**9.77e00***	**1.33e01***
F19	Mean	2.45e06	1.01e07	5.10e06	8.20e06	1.80e06	5.46e05	1.75e06	1.52e06	2.04e06	**1.14e06**
	Std	1.18e05	5.64e05	7.05e05	4.69e05	9.96e04	2.80e04	1.10e05	7.10e04	1.42e05	6.23e04
	t-test	-	**7.27e01***	**2.03e01***	**6.51e01***	-2.31e01	-8.60e01	-2.38e01	-3.70e01	-1.22e01	-5.38e01
F20	Mean	1.17e03	**1.05e03**	1.85e03	1.06e03	1.88e03	1.39e03	6.41e10	2.11e03	2.30e03	4.58e03
	Std	9.92e01	1.49e02	2.59e02	1.79e02	1.90e02	1.47e02	6.97e09	1.79e02	2.26e02	8.25e02
	t-test	-	-3.67e00	**1.34e01***	-2.94e00	**1.81e01***	**6.79e00***	**5.04e01***	**2.52e01***	**2.51e01***	**2.25e01***
w/l/t		-	18/1/1	20/0/0	13/6/1	15/4/1	16/1/3	16/3/1	18/2/0	16/2/2	19/1/0

solutions than *dpl*-PSO on all functions, except F1, F7 from CEC'2010 and M1 from CEC'2013. Comparing *dpl*-PSO with CSO and SL-PSO, the results reveal that *dpl*-PSO could

TABLE IV. Comparing the Results Delivered by Different Methods on CEC'2013 Test Suit in Terms of Mean Fitness and Standard Deviation Along with Two Tailed *t*-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	SPLSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
M1	Mean	6.92e-19	7.71e-12	1.09e-17	1.18e-19	**3.99e-22**	1.19e-20	6.42e03	4.11e01	8.60e-10	3.14e-06
	Std	1.22e-18	1.31e-12	2.50e-18	1.06e-20	1.32e-22	1.11e-20	1.81e04	3.14e01	4.38e-09	4.27e-06
	t-test	-	**3.22e01***	**2.01e01***	-2.58e00	-3.10e00	-3.05e00	1.94e00	**7.17e00***	1.08e00	**4.03e00***
M2	Mean	6.91e02	8.55e03	2.13e03	1.06e03	1.14e03	6.97e02	1.27e04	3.50e01	**3.82e00**	1.31e03
	Std	3.65e01	2.65e02	1.36e02	4.45e02	5.78e01	5.51e01	7.20e02	4.85e00	1.73e00	3.63e01
	t-test	-	**1.61e02***	**5.60e01***	**4.53e00***	**3.60e01***	4.97e-01	**9.12e01***	-9.76e01	-1.03e02	**6.59e01***
M3	Mean	2.16e01	2.16e01	2.16e01	2.16e01	2.16e01	2.03e01	2.14e01	**2.00e01**	2.00e01	2.02e01
	Std	8.64e-03	6.15e-03	1.45e-02	7.53e-03	4.07e-03	4.36e-02	1.45e-02	1.25e-04	2.76e-04	6.18e-03
	t-test	-	0.00e00	0.00e00	0.00e00	0.00e00	-1.60e02	-6.49e01	-1.01e03	-1.01e03	-7.22e02
M4	Mean	**1.64e09**	1.32e10	4.35e09	9.40e09	6.68e09	5.13e09	7.70e10	3.49e10	2.34e11	2.35e11
	Std	5.57e08	2.54e09	9.48e08	1.89e09	1.68e09	1.33e09	2.82e10	2.17e10	1.26e11	1.22e11
	t-test	-	**2.43e01***	**1.35e01***	**2.16e01***	**1.56e01***	**1.33e01***	**1.46e01***	**8.39e00***	**1.01e01***	**1.05e01***
M5	Mean	6.17e05	**5.91e05**	8.41e05	6.30e05	7.00e05	1.76e06	5.78e06	1.40e07	1.27e07	8.26e06
	Std	1.01e05	1.07e05	1.75e05	1.02e05	1.28e05	3.26e05	3.83e05	4.81e06	3.46e06	1.14e06
	t-test	-	-9.68e-01	**6.07e00***	4.96e-01	**2.79e00***	**1.83e01***	**7.14e01***	**1.52e01***	**1.91e01***	**3.66e01***
M6	Mean	1.06e06	1.06e06	1.06e06	1.06e06	1.06e06	**1.05e06**	1.06e06	**1.05e06**	1.05e06	1.06e06
	Std	8.59e02	1.10e03	1.48e03	8.05e02	8.28e02	7.00e03	1.07e03	5.24e03	4.13e03	1.84e03
	t-test	-	0.00e00	0.00e00	0.00e00	0.00e00	-7.77e00	0.00e00	-1.03e01	-1.30e01	0.00e00
M7	Mean	**8.01e04**	5.88e06	1.63e06	5.50e06	1.60e06	2.91e06	4.78e08	4.15e08	1.43e09	1.04e09
	Std	3.27e04	2.58e06	7.05e05	2.26e06	8.38e05	1.30e06	1.92e08	9.38e08	1.07e09	4.48e08
	t-test	-	**1.23e01***	**1.20e01***	**1.31e01***	**9.93e00***	**1.19e01***	**1.36e01***	**2.42e00***	**7.32e00***	**1.27e01***
M8	Mean	**4.45e13**	2.60e14	1.03e14	1.55e14	1.20e14	1.28e14	3.57e15	1.18e15	9.59e15	7.50e15
	Std	1.06e13	5.87e13	3.62e13	2.96e13	3.35e13	3.44e13	1.85e15	9.99e14	6.18e15	3.18e15
	t-test	-	**1.98e01***	**8.49e00***	**1.92e01***	**1.18e01***	**1.27e01***	**1.04e01***	**6.23e00***	**8.46e00***	**1.28e01***
M9	Mean	9.16e07	**6.06e07**	8.25e07	8.07e07	1.30e08	1.09e08	4.90e08	3.76e09	9.55e08	5.96e08
	Std	2.99e07	1.60e07	2.03e07	2.24e07	3.97e07	1.96e07	3.18e07	1.02e09	2.92e08	9.76e07
	t-test	-	-5.01e00	-1.38e00	-1.60e00	**4.23e00***	**2.67e00***	**5.00e01***	**1.97e01***	**1.61e01***	**2.71e01***
M10	Mean	**9.14e07**	9.40e07	9.25e07	9.39e07	9.40e07	9.34e07	9.45e07	9.30e07	9.27e07	9.29e07
	Std	1.39e06	1.51e05	1.67e06	2.26e05	2.11e05	3.55e05	2.46e05	7.01e05	6.07e05	6.16e05
	t-test	-	**1.02e01***	**2.77e00***	**9.72e00***	**1.01e01***	**7.64e00***	**1.20e01***	**5.63e00***	**4.69e00***	**5.40e00***
M11	Mean	9.28e11	9.30e11	9.33e11	9.27e11	9.30e11	**9.59e08**	4.83e10	9.37e11	2.28e11	1.28e11
	Std	9.63e09	1.03e10	1.46e10	9.48e09	9.50e09	1.68e09	4.33e10	1.53e10	1.53e11	7.15e10
	t-test	-	7.77e-01	1.57e00	-4.05e-01	8.10e-01	-5.19e02	-1.09e02	**2.73e00***	-2.50e01	-6.07e01
M12	Mean	1.16e03	1.07e03	1.78e03	**1.05e03**	1.79e03	1.33e03	1.71e11	2.10e03	2.49e03	4.35e03
	Std	7.18e01	7.78e01	1.74e02	5.37e01	1.39e02	1.00e02	2.24e10	1.78e02	7.51e02	7.83e02
	t-test	-	-4.66e00	**1.80e01***	-6.72e00	**2.21e01***	**7.56e00***	**4.18e01***	**2.68e01***	**9.66e00***	**2.22e01***
M13	Mean	**1.77e07**	6.67e08	4.65e08	1.20e09	3.35e08	1.04e09	2.05e10	4.02e09	1.06e10	9.35e09
	Std	7.29e06	2.45e08	2.35e08	4.99e08	1.71e08	3.28e08	5.53e09	2.31e09	3.73e09	2.78e09
	t-test	-	**1.45e01***	**1.04e01***	**1.30e01***	**1.02e01***	**1.71e01***	**2.03e01***	**9.49e00***	**1.55e01***	**1.84e01***
M14	Mean	**2.13e07**	3.62e09	3.28e08	8.31e09	1.72e08	6.53e09	1.92e10	9.10e10	2.21e11	1.42e11
	Std	4.26e06	1.44e09	5.17e08	6.67e09	1.38e08	5.70e09	1.44e10	8.53e10	8.54e10	5.86e10
	t-test	-	**1.37e01***	**3.25e00***	**6.81e00***	**5.98e00***	**6.25e00***	**7.29e00***	**5.84e00***	**1.42e01***	**1.33e01***
M15	Mean	6.17e06	7.87e07	5.86e07	4.13e07	**4.48e06**	8.48e06	9.90e06	4.75e06	1.61e07	1.16e07
	Std	5.14e05	6.50e06	6.11e06	3.11e06	3.32e05	2.25e06	2.30e06	5.07e06	1.90e06	1.26e06
	t-test	-	**6.09e01***	**4.68e01***	**6.10e01***	-1.51e01	**5.48e00***	**8.67e00***	-1.53e00	**2.76e01***	**2.19e01***
w/l/t		-	9/2/4	11/0/4	8/2/5	10/2/3	10/4/1	11/2/2	11/3/1	10/4/1	12/2/1

significantly outperform CSO and SL-PSO. For example, the results in Table II show that *dpl*-PSO is able to deliver better or comparable solutions than CSO and SL-PSO on 24 and 25, respectively, out of 35 functions to be tested. Since the only difference among the three algorithms to be compared is that they employ different particle learning strategies. These results thus indicate the significance of the proposed DPL.

4.3 Comparing with Related Algorithms

Then, we compare our proposed algorithm, *mlsdpl*_PSO, with related algorithms. The algorithms to be compared consist of recently proposed PSO variants for LSGO (including

CSO [11], SL-PSO [12], SPLSO [13] and LLSO [14]) and CCEAs for LSGO (including DECC-DG [29], CCPSO2 [21], MLCC [20] and DECC-G [19]) as well as the winner algorithm of CEC'2010 LSGO (i.e., MA-SW-Chains [34]). To facilitate a fair comparison, the same *FE_Max* value (i.e., 3000×D) is used for all methods. For the rest parameters of the methods to be compared, they are set according to the original papers with the best performance.

Tables III and IV show the comparison results of different methods on CEC'2010 and CEC'2013 benchmark suits, respectively. To statistically justify the comparisons between our proposed method and the related algorithms, two-tailed t-tests are performed at a significance level of $\alpha = 0.05$ and the results have also been reported in Tables III and IV. In addition, the number of *wins*, *losses* and *ties* on the test functions for each pairwise comparison between our algorithm and related methods have been summarized in the last rows of Tables III and IV. From the results, we can see that our proposed method could significantly outperform related algorithms to be compared. For example, the results show that, comparing to the PSO variants of CSO, SL-PSO, SPLSO and LLSO, *mlsdpl*_PSO can deliver better solutions on 27, 31, 21 and 25, respectively, out of 35 functions to be tested. Similar results can also be found by comparing our proposed method to CCEAs including DECC-DG, CCPSO2, MLCC and DECC-G that the *mlsdpl*_PSO is able to provide better solutions on most of the functions. While, comparing to MA-SW-Chains, which is a winner algorithm of CEC'2010 LSGO, our method gives better solutions on 26 out of 35 functions. Clearly, based the results, *mlsdpl*_PSO shows the best performance among the ten algorithms. The superiority is mainly due to the incorporation of MLS mechanism, which helps balance the exploitation and exploration of PSO evolution, as well as the DPL mechanism, which could be used to support efficient particle learning while preserving the swarm diversity during evolution. Equipped with these two mechanisms, the resulting *mlsdpl*_PSO could achieve a superior performance for addressing LSGO.

4.4 Scalability Evaluation and Comparison

To evaluate the scalability of our proposed algorithm, experiments have also been conducted on CEC'2010 functions with various dimensions including 200, 500, 800 as well as 2000 and the performance are compared with related methods. Same as previous experiments, a *FE_Max* value of 3000×D is used for all methods to make the comparison fair. The results are reported in Tables V, VI, VII and VIII for problem dimensions of 200, 500, 800 and 2000, respectively.

TABLE V. Comparing the Results Delivered by Different Methods on CEC'2010 Test Suit with Dimension *D*=200 in Terms of Mean Fitness and Standard Deviation Along with Two Tailed *t*-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
	Mean	4.84e-27	6.45e-17	2.41e-21	**0.00e00**	6.73e-22	2.43e-21	2.44e02	3.88e-18	1.02e-12
F1	Std	2.00e-26	2.18e-17	2.07e-22	0.00e00	5.11e-22	1.73e-21	4.58e02	9.37e-18	4.32e-13
	t-test	-	**1.62e01***	**6.38e01***	-1.33e00	**7.21e00***	**7.69e00***	**2.92e00***	**2.27e00***	**1.29e01***
	Mean	1.05e02	9.70e02	3.29e02	1.00e02	7.56e00	2.11e02	1.86e00	**1.08e-11**	3.22e01
F2	Std	1.47e01	1.10e02	3.73e01	1.13e01	6.40e00	1.53e01	1.10e00	7.75e-12	5.29e00
	t-test	-	**4.27e01***	**3.06e01***	-1.48e00	-3.33e01	**2.74e01***	-3.83e01	-3.91e01	-2.55e01
	Mean	3.98e-14	1.89e-11	4.99e-14	**1.45e-14**	2.40e-14	2.38e00	1.12e-02	3.04e-13	1.37e-07
F3	Std	3.48e-15	2.75e-12	1.30e-15	6.49e-16	3.43e-15	3.29e-01	6.71e-03	7.68e-13	5.71e-08
	t-test	-	**3.76e01***	**1.49e01***	-3.91e01	-1.77e01	**3.96e01***	**9.14e00***	1.88e00	**1.31e01***
	Mean	**7.00e11**	1.99e12	1.46e12	1.72e12	1.57e12	4.31e13	4.05e12	5.66e12	9.76e12
F4	Std	1.55e11	3.67e11	4.04e11	4.08e11	3.53e11	9.55e12	1.62e12	1.65e12	3.31e12
	t-test	-	**1.77e01***	**9.62e00***	**1.28e01***	**1.24e01***	**2.43e01***	**1.13e01***	**1.64e01***	**1.50e01***
	Mean	**2.26e07**	3.03e08	3.23e08	2.39e07	1.09e08	2.50e08	4.36e08	3.06e08	2.80e08
F5	Std	4.89e06	8.87e06	1.37e07	4.46e07	3.78e07	2.40e07	1.33e08	1.31e08	3.78e07
	t-test	-	**1.52e02***	**1.13e02***	1.59e-01	**1.24e01***	**5.09e01***	**1.70e01***	**1.18e01***	**3.70e01***
	Mean	**7.44e-09**	1.67e-07	2.10e01	2.00e00	2.31e05	1.43e00	1.80e07	9.34e06	2.32e06
F6	Std	6.36e-10	6.55e-09	1.51e-01	5.29e00	4.98e05	2.37e-01	4.83e06	8.10e06	3.33e05
	t-test	-	**1.33e02***	**7.62e02***	**2.07e00***	**2.54e00***	**3.30e01***	**2.04e01***	**6.32e00***	**3.82e01***
	Mean	**1.62e04**	3.33e05	3.71e05	4.29e04	4.92e04	1.19e10	4.30e08	1.45e05	1.41e08
F7	Std	1.90e04	1.09e05	8.21e04	3.01e04	2.17e04	2.02e09	1.01e09	1.16e05	1.50e08
	t-test	-	**1.57e01***	**2.31e01***	**4.11e00***	**6.27e00***	**3.23e01***	**2.33e00***	**6.00e00***	**5.15e00***
	Mean	1.90e07	4.25e07	4.38e07	4.15e07	6.92e07	1.10e08	7.34e07	**1.18e07**	4.07e07
F8	Std	2.44e06	9.50e04	3.35e07	1.62e07	8.93e07	2.02e07	5.46e07	1.36e07	4.74e07
	t-test	-	**5.27e01***	**4.04e00***	**7.52e00***	**3.08e00***	**2.45e01***	**5.45e00***	-2.85e00	**2.50e00***
	Mean	**3.45e06**	8.49e06	6.71e06	6.74e06	5.86e06	1.71e07	1.59e07	1.69e07	3.83e07
F9	Std	5.50e05	1.57e06	1.48e06	1.20e06	1.04e06	4.76e06	2.78e06	4.31e06	1.03e07
	t-test	-	**1.66e01***	**1.13e01***	**1.37e01***	**1.12e01***	**1.56e01***	**2.41e01***	**1.70e01***	**1.85e01***
	Mean	1.21e02	1.52e03	1.64e03	**9.36e01**	1.50e02	6.41e02	8.18e02	4.99e02	1.29e03
F10	Std	1.47e01	4.16e01	2.45e02	8.40e00	2.34e01	2.38e01	1.84e02	1.29e02	5.96e01
	t-test	-	**1.74e02***	**3.39e01***	-8.86e00	**5.75e00***	**1.02e02***	**2.07e01***	**1.59e01***	**1.04e02***
	Mean	4.69e-14	8.17e-11	9.29e00	**2.25e-14**	1.20e00	2.27e-01	3.27e01	7.22e00	3.35e00
F11	Std	7.29e-15	1.06e-11	7.87e00	1.14e-15	1.09e00	4.21e-01	1.19e01	8.43e00	7.61e-01
	t-test	-	**4.22e01***	**6.47e00***	-1.81e01	**6.03e00***	**2.95e00***	**1.51e01***	**4.69e00***	**2.41e01***
	Mean	**7.20e01**	3.22e03	7.58e03	1.02e02	9.29e02	1.28e03	8.25e03	9.77e02	2.08e03
F12	Std	2.45e01	6.93e02	1.04e04	3.45e01	4.55e02	2.52e02	3.84e03	3.63e02	9.57e02
	t-test	-	**2.49e01***	**3.95e00***	**3.88e00***	**1.03e01***	**2.61e01***	**1.17e01***	**1.36e01***	**1.15e01***
	Mean	**7.51e01**	9.53e01	1.37e02	9.30e01	1.69e02	9.85e03	2.84e02	3.52e02	6.85e02
F13	Std	1.89e01	1.82e01	1.04e02	2.20e01	1.17e02	1.57e03	6.83e01	2.21e02	1.01e03
	t-test	-	**4.22e00***	**3.21e00***	**3.38e00***	**4.34e00***	**3.41e01***	**1.61e01***	**6.84e00***	**3.31e00***
	Mean	**1.20e07**	3.51e07	2.01e07	2.03e07	2.06e07	5.73e07	3.52e07	3.91e07	9.93e07
F14	Std	1.94e06	4.81e06	3.23e06	2.60e06	2.12e06	1.07e07	6.13e06	7.05e06	1.80e07
	t-test	-	**2.44e01***	**1.18e01***	**1.40e01***	**1.64e01***	**2.28e01***	**1.98e01***	**2.03e01***	**2.64e01***
	Mean	5.54e02	1.75e03	1.92e03	1.69e03	**3.37e02**	1.17e03	1.87e03	1.05e03	2.02e03
F15	Std	6.25e02	2.44e01	3.61e01	6.70e01	2.62e01	4.02e01	3.82e02	2.17e02	9.96e01
	t-test	-	**1.05e01***	**1.20e01***	**9.90e00***	-1.90e00	**5.39e00***	**9.84e00***	**4.11e00***	**1.27e01***
	Mean	6.03e-14	7.89e-11	1.97e00	**2.94e-14**	4.26e02	3.64e-11	7.65e01	1.58e01	1.01e01
F16	Std	5.50e-15	1.05e-11	3.37e00	1.53e-15	5.04e-01	7.93e-12	6.08e00	2.14e01	3.57e00
	t-test	-	**4.11e01***	**3.20e00***	-2.96e01	**4.63e03***	**2.51e01***	**6.89e01***	**4.04e00***	**1.55e01***
	Mean	1.11e04	1.14e05	5.31e04	**4.81e03**	2.86e04	7.87e03	2.07e04	1.23e04	1.35e04
F17	Std	1.78e03	1.70e04	1.12e04	8.69e02	8.02e03	1.04e03	5.90e03	2.07e03	2.41e03
	t-test	-	**3.30e01***	**2.03e01***	-1.74e01	**1.17e01***	-8.58e00	**8.53e00***	**2.41e00***	**4.39e00***
	Mean	**1.66e02**	1.94e02	3.46e02	1.87e02	3.28e02	6.44e02	6.09e02	4.90e02	1.68e03
F18	Std	2.44e01	2.48e01	1.45e02	3.00e01	1.64e02	8.92e01	1.55e02	2.09e02	1.64e03
	t-test	-	**4.41e00***	**6.71e00***	**2.97e00***	**5.35e00***	**2.83e01***	**1.55e01***	**8.43e00***	**5.06e00***
	Mean	7.59e04	3.58e05	3.43e05	4.11e04	**1.07e04**	9.50e04	9.67e04	4.83e04	3.58e04
F19	Std	1.25e04	2.82e04	5.34e04	5.09e03	1.99e03	9.49e03	1.92e04	5.12e03	6.63e03
	t-test	-	**5.01e01***	**2.67e01***	-1.41e01	-2.82e01	**6.67e00***	**4.97e00***	-1.12e01	-1.55e01
	Mean	**1.94e02**	2.39e02	3.62e02	2.82e02	3.44e02	3.57e02	5.23e02	3.52e02	6.37e02
F20	Std	4.09e01	1.58e02	2.32e02	1.98e02	1.20e02	6.32e01	1.11e02	8.24e01	1.99e02
	t-test	-	1.51e00	**3.91e00***	**2.38e00***	**6.48e00***	**1.19e01***	**1.52e01***	**9.41e00***	**1.19e01***
w/l/t		-	19/0/1	20/0/0	11/6/3	16/3/1	19/1/0	19/1/0	16/3/1	18/2/0

From the results, we can see that our proposed algorithm can outperform the related methods across all dimensions to be tested, except LLSO on 2000 dimensions. On the

TABLE VI. Comparing the Results Delivered by Different Methods on CEC'2010 Test Suit with Dimension *D*=500 in Terms of Mean Fitness and Standard Deviation Along with Two Tailed *t*-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
	Mean	2.17e-21	2.52e-17	4.61e-19	**0.00e00**	2.79e-21	2.37e-07	7.24e00	5.98e-17	1.68e-17
F1	Std	2.44e-21	7.46e-18	2.99e-20	0.00e00	3.71e-21	3.33e-07	1.12e01	2.19e-16	9.42e-18
	t-test	-	**1.85e01***	**8.38e01***	-4.87e00	7.65e-01	**3.90e00***	**3.54e00***	1.50e00	**9.77e00***
	Mean	3.05e02	2.77e03	1.08e03	4.05e02	4.39e01	1.32e03	1.04e00	**3.17e-11**	8.03e-02
F2	Std	2.23e01	3.40e02	9.01e01	2.80e01	3.64e01	7.65e01	4.92e-01	2.22e-11	2.52e-01
	t-test	-	**3.96e01***	**4.57e01***	**1.53e01***	-3.35e01	**6.98e01***	-7.46e01	-7.49e01	-7.49e01
	Mean	1.18e-13	8.37e-12	1.06e00	**2.16e-14**	5.20e-14	1.03e01	2.08e-03	1.98e-12	9.00e-10
F3	Std	6.42e-15	1.34e-12	5.54e-01	6.49e-16	6.65e-15	8.63e-01	1.52e-03	6.50e-12	3.21e-10
	t-test	-	**3.37e01***	**1.05e01***	-8.18e01	-3.91e01	**6.54e01***	**7.50e00***	1.57e00	**1.54e01***
	Mean	**4.22e11**	1.73e12	1.13e12	1.29e12	1.10e12	1.83e13	4.40e12	1.08e13	2.58e13
F4	Std	8.22e10	3.68e11	2.94e11	3.64e11	2.30e11	3.91e12	2.09e12	3.94e12	6.54e12
	t-test	-	**1.90e01***	**1.27e01***	**1.27e01***	**1.52e01***	**2.50e01***	**1.04e01***	**1.44e01***	**2.13e01***
	Mean	**1.06e07**	2.88e08	1.54e08	1.71e07	4.57e07	1.93e08	5.00e08	4.30e08	2.45e08
F5	Std	2.98e06	8.87e06	1.34e08	2.92e06	7.60e06	2.49e07	1.59e08	1.04e08	8.05e07
	t-test	-	**1.62e02***	**5.86e00***	**8.53e00***	**2.36e01***	**3.98e01***	**1.69e01***	**2.21e01***	**1.59e01***
	Mean	**1.00e-08**	2.38e-07	2.13e01	1.74e00	3.42e05	9.07e00	1.88e07	1.84e07	5.18e06
F6	Std	2.65e-09	1.15e-08	1.19e-01	9.85e-01	5.49e05	8.73e-01	3.18e06	4.16e06	9.99e05
	t-test	-	**1.06e02***	**9.80e02***	**9.68e00***	**3.41e00***	**5.69e01***	**3.24e01***	**2.42e01***	**2.84e01***
	Mean	8.33e01	5.16e03	1.03e05	1.48e04	**4.11e01**	2.61e08	7.18e08	1.73e06	7.23e08
F7	Std	2.08e02	1.37e03	4.53e04	1.66e04	1.85e01	9.95e07	1.02e09	2.38e06	3.69e08
	t-test	-	**2.01e01***	**1.24e01***	**4.86e00***	-1.11e00	**1.44e01***	**3.86e00***	**3.98e00***	**1.07e01***
	Mean	**7.72e05**	3.89e07	2.59e07	2.66e07	2.15e07	4.97e07	4.80e07	5.41e07	9.03e07
F8	Std	6.81e05	1.03e05	1.22e07	2.59e05	2.79e07	2.10e07	4.34e07	2.40e07	1.53e08
	t-test	-	**3.03e02***	**1.13e01***	**1.94e02***	**4.07e00***	**1.28e01***	**5.96e00***	**1.22e01***	**3.20e00***
	Mean	**8.73e06**	2.55e07	1.76e07	1.55e07	1.28e07	2.21e07	4.38e07	5.57e07	2.00e08
F9	Std	1.49e06	3.53e06	3.66e06	2.26e06	1.33e06	3.88e06	1.47e07	8.36e06	2.94e07
	t-test	-	**2.40e01***	**1.23e01***	**1.37e01***	**1.12e01***	**1.76e01***	**1.30e01***	**3.03e01***	**3.56e01***
	Mean	**3.62e02**	4.56e03	4.25e03	3.79e02	4.88e02	1.85e03	2.37e03	2.10e03	3.47e03
F10	Std	2.14e01	5.29e01	1.55e03	2.27e01	6.31e01	6.03e01	4.48e02	5.26e02	4.88e02
	t-test	-	**4.03e02***	**1.37e01***	**2.98e00***	**1.04e01***	**1.27e02***	**2.45e01***	**1.81e01***	**3.49e01***
	Mean	**9.73e-13**	1.02e-10	2.23e01	1.29e00	4.58e00	3.66e00	9.89e01	9.81e01	1.43e01
F11	Std	4.26e-13	2.43e-11	5.33e00	3.85e00	1.59e00	5.08e-01	3.44e-01	2.85e00	8.11e-01
	t-test	-	**2.28e01***	**2.29e01v**	1.84e00	**1.58e01***	**3.95e01***	**1.57e03***	**1.89e02***	**9.66e01***
	Mean	2.01e03	7.10e04	5.62e04	**1.26e03**	1.71e04	1.26e03	1.84e04	1.19e04	1.87e04
F12	Std	2.78e02	1.25e04	2.87e04	2.64e02	4.73e03	2.47e02	3.45e03	2.40e03	3.41e03
	t-test	-	**3.02e01***	**1.03e01***	-1.07e01	**1.74e01***	-1.10e01	**2.59e01***	**2.24e01***	**2.67e01***
	Mean	**2.14e02**	3.17e02	6.61e02	2.92e02	4.69e02	4.62e05	7.14e02	1.62e03	3.12e03
F13	Std	4.71e01	1.79e02	1.22e03	1.05e02	2.27e02	4.92e04	1.29e02	2.09e03	4.98e03
	t-test	-	**3.05e00***	2.01e00	**3.71e00***	**6.02e00***	**5.14e01***	**1.99e01***	**3.68e00***	**3.20e00***
	Mean	**3.06e07**	1.00e08	4.92e07	4.41e07	6.97e07	1.71e08	1.24e08	1.51e08	4.19e08
F14	Std	2.65e06	1.06e07	5.53e06	4.35e06	5.44e06	1.70e07	4.69e07	1.88e07	4.52e07
	t-test	-	**3.48e01***	**1.66e01***	**1.45e01***	**3.54e01***	**4.47e01***	**1.09e01***	**3.47e01***	**4.70e01***
	Mean	**5.95e02**	4.91e03	5.42e03	4.78e03	1.11e03	2.94e03	5.06e03	4.18e03	4.42e03
F15	Std	2.42e02	4.55e01	5.37e01	5.84e01	7.39e01	4.90e01	6.37e02	9.95e02	1.51e03
	t-test	-	**9.60e01***	**1.07e02***	**9.21e01***	**1.11e01****	**5.20e01***	**3.59e01***	**1.92e01***	**1.37e01***
	Mean	1.24e-12	1.06e-10	2.30e01	6.71e-01	1.33e01	**1.28e-13**	1.98e02	1.73e02	2.54e01
F16	Std	2.44e-13	2.25e-11	1.53e01	9.83e-01	5.96e00	7.88e-15	2.10e00	5.63e01	1.76e00
	t-test	-	**2.55e01***	**8.23e00***	**3.74e00***	**1.22e01***	-2.49e01	**5.16e02***	**1.68e01***	**7.90e01***
	Mean	4.96e04	7.45e05	1.21e05	2.26e04	4.53e04	**2.15e04**	6.74e04	6.73e04	6.88e04
F17	Std	5.47e03	7.55e04	3.26e04	2.30e03	6.11e03	2.21e03	1.77e04	9.33e03	8.25e03
	t-test	-	**5.03e01***	**1.18e01***	-2.49e01	-2.87e00	-2.61e01	**5.26e00***	**8.96e00***	**1.06e01***
	Mean	**5.10e02**	8.45e02	1.41e03	7.78e02	1.14e03	1.20e06	1.42e03	2.57e03	1.59e04
F18	Std	7.68e01	4.32e02	1.20e03	2.78e02	5.14e02	4.63e05	1.42e02	3.17e03	9.70e03
	t-test	-	**4.18e00***	**4.10e00***	**5.09e00***	**6.64e00***	**1.42e01***	**3.09e01***	**3.56e00***	**8.69e00***
	Mean	8.05e05	2.89e06	1.68e06	5.85e05	**1.21e05**	5.56e05	4.77e05	3.97e05	2.19e05
F19	Std	7.07e04	2.28e05	2.58e05	5.00e04	1.10e04	3.23e04	4.39e04	3.41e04	2.08e04
	t-test	-	**4.78e01***	**1.79e01***	-1.39e01	-5.24e01	-1.75e01	-2.16e01	-2.85e01	-4.36e01
	Mean	**4.91e02**	5.49e02	7.38e02	7.63e02	7.11e02	1.60e10	1.09e03	9.53e02	1.63e03
F20	Std	4.00e01	8.90e01	1.19e02	1.42e02	8.47e01	2.92e09	1.55e02	1.32e02	1.00e03
	t-test	-	**3.26e00***	**1.08e01***	**1.01e01***	**1.29e01***	**3.00e01***	**2.05e01***	**1.83e01***	**6.23e00***
w/l/t		-	20/0/0	19/0/1	14/5/1	14/4/2	16/4/0	18/2/0	16/2/2	18/2/0

functions with a dimension of 2000, the results in Table VIII show that, comparing to the four CCEAs (i.e., DECC-DG, CCPSO2, MLCC and DECC-G), *mlsdpl*_PSO can provide better

TABLE VII. Comparing the Results Delivered by Different Methods on CEC'2010 Test Suit with Dimension *D*=800 in Terms of Mean Fitness and Standard Deviation Along with Two Tailed *t*-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
F1	Mean	8.45e-20	6.93e-12	3.71e-18	**2.36e-23**	8.61e-21	3.60e00	3.93e00	1.91e-11	8.28e-07
	Std	3.99e-20	1.06e-12	4.96e-19	1.43e-23	1.44e-20	5.68e00	7.27e00	8.61e-11	4.10e-07
	t-test	-	**3.58e01***	**3.99e01***	-1.16e01	-9.80e00	**3.47e00***	**2.96e00***	1.22e00	**1.11e01***
F2	Mean	4.96e02	5.95e03	1.69e03	6.96e02	2.69e02	3.08e03	**2.38e00**	2.65e00	1.15e03
	Std	2.17e01	1.99e02	1.28e02	3.66e01	8.74e01	1.46e02	7.74e-01	1.08e00	2.04e01
	t-test	-	**1.49e02***	**5.04e01***	**2.57e01***	-1.38e01	**9.59e01***	-1.25e02	-1.24e02	**1.20e02***
F3	Mean	3.03e-13	3.46e-09	1.59e00	**2.29e-14**	1.54e-13	1.51e01	3.17e-03	2.70e-07	9.78e-01
	Std	1.50e-13	2.59e-10	2.56e-01	2.53e-15	1.40e-13	4.37e-01	9.17e-04	1.36e-06	4.05e-01
	t-test	-	**7.32e01***	**3.40e01***	-1.02e01	-3.98e00	**1.89e02***	**1.89e01***	1.09e00	**1.32e01***
F4	Mean	**1.99e11**	9.73e11	4.29e11	7.12e11	4.06e11	2.64e12	2.02e12	1.28e13	2.24e13
	Std	5.23e10	1.79e11	1.89e11	1.71e11	8.77e10	8.93e11	1.01e12	3.64e12	8.98e12
	t-test	-	**2.27e01***	**6.42e00***	**1.57e01***	**1.11e01***	**1.49e01***	**9.86e00***	**1.90e01***	**1.35e01***
F5	Mean	5.71e06	**3.36e06**	5.96e07	1.28e07	3.58e07	1.60e08	4.06e08	4.17e08	1.88e08
	Std	1.83e06	1.56e06	7.84e07	3.10e06	6.49e06	1.83e07	1.45e08	9.27e07	4.58e07
	t-test	-	-5.35e00	**3.76e00***	**1.08e01***	**2.44e01***	**4.60e01***	**1.51e01***	**2.43e01***	**2.18e01***
F6	Mean	**2.79e-08**	8.26e-07	2.14e01	1.55e-01	2.55e05	1.44e01	1.84e07	1.90e07	3.74e06
	Std	2.01e-08	2.10e-08	1.46e-01	4.17e-01	5.93e05	5.14e-01	4.02e06	2.48e06	7.50e05
	t-test	-	**1.50e02***	**8.03e02***	2.04e00	**2.36e00***	**1.53e02***	**2.51e01***	**4.20e01***	**2.73e01***
F7	Mean	6.02e02	2.64e04	5.15e04	7.04e02	**4.44e00**	1.40e08	3.70e08	7.63e07	6.46e08
	Std	3.26e03	8.03e03	3.76e04	3.51e03	1.94e00	5.48e07	4.96e08	7.88e07	4.93e08
	t-test	-	**1.63e01***	**7.39e00***	1.17e-01	-1.00e00	**1.40e01***	**4.09e00***	**5.30e00***	**7.18e00***
F8	Mean	**1.94e06**	6.41e07	3.57e07	5.84e07	3.86e08	9.87e07	1.34e08	7.01e08	3.71e08
	Std	4.66e06	3.59e07	5.85e07	5.16e07	3.52e08	6.35e07	6.14e07	2.58e09	1.85e09
	t-test	-	**9.40e00***	**3.15e00***	**5.97e00***	**5.98e00***	**8.32e00***	**1.17e01***	1.48e00	1.09e00
F9	Mean	**1.51e07**	5.67e07	2.53e07	3.56e07	2.18e07	4.99e07	7.29e07	1.84e08	3.05e08
	Std	1.58e06	4.87e06	2.73e06	3.19e06	1.83e06	7.25e06	2.44e07	2.19e07	3.54e07
	t-test	-	**4.45e01***	**1.77e01***	**3.15e01***	**1.52e01***	**2.57e01***	**1.29e01***	**4.21e01***	**4.48e01***
F10	Mean	**5.65e02**	7.50e03	4.54e03	6.19e02	9.43e02	3.39e03	4.06e03	2.99e03	8.04e03
	Std	2.53e01	5.26e01	3.13e03	2.91e01	8.00e01	1.16e02	6.10e02	1.07e03	2.38e02
	t-test	-	**6.51e02***	**6.96e00***	**7.67e00***	**2.47e01***	**1.30e02***	**3.14e01***	**1.24e01***	**1.71e02***
F11	Mean	**4.18e-12**	4.59e-08	2.32e01	5.72e-01	7.64e00	7.73e00	1.58e02	1.50e02	1.92e01
	Std	2.83e-12	3.62e-09	2.27e00	7.00e-01	2.50e00	1.12e00	4.81e-01	3.19e01	1.47e00
	t-test	-	**6.94e01***	**5.60e01***	**4.48e00***	**1.67e01***	**3.78e01***	**1.80e03***	**2.58e01***	**7.15e01***
F12	Mean	3.03e03	2.51e05	3.08e04	8.97e03	4.46e04	**1.61e03**	3.07e04	7.06e04	6.87e04
	Std	5.44e02	3.01e04	1.47e04	1.01e03	1.02e04	1.79e02	8.46e03	1.16e04	9.01e03
	t-test	-	**4.51e01***	**1.03e01***	**2.84e01***	**2.23e01***	-1.36e01	**1.79e01***	**3.19e01***	**3.98e01***
F13	Mean	**4.04e02**	7.15e02	1.91e03	6.73e02	9.62e02	8.25e04	1.11e03	3.68e03	5.02e03
	Std	8.36e01	4.02e02	4.53e03	3.00e02	4.17e02	4.81e04	1.70e02	3.59e03	4.57e03
	t-test	-	**4.15e00***	1.82e00	**4.73e00***	**7.19e00***	**9.35e00***	**2.04e01***	**5.00e00***	**5.53e00***
F14	Mean	**4.35e07**	2.04e08	6.90e07	1.01e08	1.44e08	2.69e08	2.18e08	4.43e08	7.69e08
	Std	2.32e06	1.27e07	7.80e06	6.21e06	1.09e07	2.24e07	7.36e07	3.53e07	7.01e07
	t-test	-	**6.81e01***	**1.72e01***	**4.75e01***	**4.94e01***	**5.48e01***	**1.30e01***	**6.19e01***	**5.67e01***
F15	Mean	8.03e02	7.92e03	8.95e03	**6.08e02**	1.97e03	4.67e03	8.48e03	7.33e03	9.87e03
	Std	5.90e01	5.39e01	9.01e01	3.30e01	1.08e02	7.07e01	1.10e03	1.85e03	6.28e02
	t-test	-	**4.88e02***	**4.14e02***	-1.58e01	**5.19e01***	**2.30e02***	**3.82e01***	**1.93e01***	**7.87e01***
F16	Mean	**4.83e-12**	6.10e-08	2.45e01	8.40e-01	3.10e01	5.81e-12	3.17e02	2.93e02	5.32e01
	Std	2.85e-12	5.18e-09	1.13e01	1.13e00	1.34e01	6.70e-13	5.18e-01	7.01e01	5.77e00
	t-test	-	**6.45e01***	**1.19e01***	**4.07e00***	**1.27e01***	1.83e00	**3.35e03***	**2.29e01***	**5.05e01***
F17	Mean	5.68e04	1.67e06	9.63e04	6.79e04	6.64e04	**3.05e04**	1.07e05	2.58e05	2.16e05
	Std	5.51e03	1.00e05	2.00e04	3.93e03	7.78e03	2.61e03	3.77e04	2.76e04	1.81e04
	t-test	-	**8.82e01***	**1.04e01***	**8.98e00***	**5.52e00***	-2.36e01	**7.22e00***	**3.92e01***	**4.61e01***
F18	Mean	**9.46e02**	1.37e03	1.99e03	2.10e03	2.04e03	1.05e09	2.51e03	7.46e03	1.90e04
	Std	8.69e01	4.92e02	6.62e02	6.09e02	5.07e02	3.00e08	2.66e02	4.95e03	1.15e04
	t-test	-	**4.65e00***	**8.56e00***	**1.03e01***	**1.16e01***	**1.92e01***	**3.06e01***	**7.21e00***	**8.60e00***
F19	Mean	1.84e06	6.31e06	3.76e06	1.17e06	**3.29e05**	1.24e06	1.06e06	1.39e06	7.97e05
	Std	1.40e05	4.06e05	6.28e05	5.63e04	2.38e04	6.76e04	6.75e04	8.96e04	4.49e04
	t-test	-	**5.70e01***	**1.63e01***	-2.43e01	-5.83e01	-2.11e01	-2.75e01	-1.48e01	-3.89e01
F20	Mean	8.99e02	**8.46e02**	1.35e03	1.36e03	1.10e03	5.05e09	1.70e03	2.05e03	3.56e03
	Std	8.33e01	1.82e02	2.04e02	2.49e02	1.19e02	1.12e09	2.19e02	5.32e02	4.78e02
	t-test	-	-1.45e00	**1.12e01***	**9.62e00***	**7.58e00***	**2.47e01***	**1.87e01***	**1.17e01***	**3.00e01***
w/l/t		-	18/1/1	19/0/1	14/4/2	15/4/1	16/3/1	18/2/0	15/2/3	18/1/1

solutions on at least 15 out of 20 functions. While, comparing to CSO, SL-PSO and LLSO, *mlsdpl*_PSO delivers better solutions on 11, 20 and 8, respectively, out of 20 functions. For

TABLE VIII. Comparing the Results Delivered by Different Methods on CEC'2010 Test Suit with Dimension D=2000 in Terms of Mean Fitness and Standard Deviation Along with Two Tailed t-Tests Between Our Proposed Method and Each of The Related Methods. The Symbol "*" Indicates That the Performance of Our Proposed Method Is Significantly Better Than the Method to be Compared with a Confidence Level of 95%.

Function	Index	*mlsdpl*_PSO	CSO	SL-PSO	LLSO	MA-SW-Chains	DECC-DG	CCPSO2	MLCC	DECC-G
F1	Mean	2.69e-14	2.63e-11	1.40e08	**1.75e-20**	1.01e-19	5.48e07	2.81e01	2.04e-15	2.39e-09
	Std	1.46e-13	2.53e-12	2.89e07	8.86e-22	8.20e-20	5.59e07	5.09e01	8.41e-15	4.62e-10
	t-test	-	**5.68e01***	**2.65e01***	-1.01e00	-1.01e00	**5.37e00***	**3.02e00***	-9.31e-01	**2.83e01***
F2	Mean	3.66e03	5.31e03	4.15e03	1.40e03	1.90e03	1.22e04	2.80e01	**7. 08e00**	2.30e03
	Std	9.82e01	8.14e02	1.75e02	5.17e01	3.20e02	3.19e02	8.99e00	3.32e00	5.66e01
	t-test	-	**1.10e01***	**1.34e01***	-1.12e02	-2.88e01	**1.40e02***	-2.02e02	-2.04e02	-6.57e01
F3	Mean	1.46e00	3.46e-09	6.34e00	**3.94e-14**	2.80e-08	1.90e01	1.78e-02	7.94e-02	1.23e00
	Std	1.30e-01	1.83e-10	3.24e-01	6.49e-16	6.45e-08	8.24e-02	7.77e-03	3.02e-01	1.11e-01
	t-test	-	-6.15e01	**7.66e01***	-6.15e01	-6.15e01	**6.24e02***	-6.07e01	-2.30e01	-7.37e00
F4	Mean	**1.18e11**	6.29e11	2.32e12	3.62e11	2.97e11	1.35e12	2.00e12	1.95e13	1.94e13
	Std	2.26e10	8.73e10	3.97e11	7.02e10	8.19e10	3.34e11	9.79e11	6.50e12	6.32e12
	t-test	-	**3.10e01***	**3.03e01***	**1.81e01***	**1.15e01***	**2.02e01***	**1.05e01***	**1.63e01***	**1.67e01***
F5	Mean	1.20e07	**4.62e06**	2.02e07	6.00e06	3.36e07	1.51e08	4.43e08	4.34e08	2.73e08
	Std	2.21e06	1.54e06	5.90e06	1.74e06	5.01e06	2.62e07	1.15e08	8.06e07	1.08e08
	t-test	-	-1.50e01	**7.13e00***	-1.17e01	**2.16e01***	**2.90e01***	**2.05e01***	**2.87e01***	**1.32e01***
F6	Mean	6.77e00	2.15e-06	1.99e01	**4.00e-09**	1.88e05	1.94e01	1.62e07	1.96e07	9.30e06
	Std	1.09e00	4.66e-08	2.33e-02	8.24e-14	5.10e05	1.41e-01	5.87e06	4.34e05	3.84e06
	t-test	-	-3.40e01	**6.60e01***	-3.40e01	2.02e00	**6.29e01***	**1.51e01***	**2.47e02***	**1.33e01***
F7	Mean	2.76e05	3.64e04	2.45e08	**1.17e01**	3.67e01	1.96e02	1.20e08	3.09e08	1.43e09
	Std	8.04e04	1.07e04	1.05e08	5.74e00	1.56e01	1.69e02	3.20e08	2.32e08	8.79e08
	t-test	-	-1.62e01	**1.28e01***	-1.88e01	-1.88e01	-1.88e01	2.05e00	**7.29e00***	**8.91e00***
F8	Mean	**1.15e06**	3.78e07	6.82e07	3.15e07	7.25e08	4.83e07	1.11e08	3.04e07	2.45e07
	Std	2.47e06	6.02e04	3.49e07	2.11e07	1.77e09	3.13e07	5.27e07	1.19e07	1.43e07
	t-test	-	**8.12e01***	**1.05e01***	**7.82e00***	**2.24e00***	**8.23e00***	**1.14e01***	**1.32e01***	**8.81e00***
F9	Mean	**3.52e07**	1.67e08	1.60e09	1.06e08	7.43e07	2.76e08	2.11e08	5.05e08	9.68e08
	Std	2.09e06	7.57e06	1.10e08	6.93e06	4.33e06	5.02e07	1.09e08	3.21e07	6.63e07
	t-test	-	**9.19e01***	**7.79e01***	**5.36e01***	**4.45e01***	**2.63e01***	**8.83e00***	**8.00e01***	**7.70e01***
F10	Mean	3.40e03	1.85e04	4.27e03	**1.16e03**	3.75e03	1.16e04	1.06e04	8.71e03	6.58e03
	Std	9.49e01	1.61e02	3.62e02	5.00e01	1.47e02	2.91e02	1.43e03	2.48e03	1.87e02
	t-test	-	**4.43e02***	**1.27e01***	-1.14e02	**1.10e01***	**1.47e02***	**2.75e01***	**1.17e01***	**8.31e01***
F11	Mean	2.08e01	1.19e-07	1.06e02	**5.72e-13**	1.81e01	1.78e01	3.97e02	3.96e02	6.81e01
	Std	2.50e00	7.22e-09	9.55e00	2.10e-14	5.58e00	3.34e-01	4.67e-01	8.02e-01	3.05e00
	t-test	-	-4.56e01	**4.73e01***	-4.56e01	-2.42e00	-6.51e00	**8.10e02***	**7.83e02***	**6.57e01***
F12	Mean	**2.62e04**	4.39e05	1.38e06	1.12e05	2.43e05	1.26e05	8.01e04	1.94e05	3.29e05
	Std	4.25e03	1.21e04	7.82e04	5.91e03	2.72e04	1.80e04	3.91e04	1.31e04	2.11e04
	t-test	-	**1.76e02***	**9.47e01***	**6.46e01***	**4.31e01***	**2.96e01***	**7.51e00***	**6.67e01***	**7.71e01***
F13	Mean	2.58e03	1.79e03	1.07e07	**1.48e03**	2.99e03	1.92e09	3.19e03	8.82e03	2.12e04
	Std	3.53e02	7.28e02	2.53e06	3.13e02	8.43e02	6.06e08	8.37e02	6.44e03	1.11e04
	t-test	-	-5.35e00	**2.32e01**	-1.28e01	**2.46e00***	**1.74e01***	**3.68e00***	**5.30e00***	**9.18e00***
F14	Mean	**1.06e08**	5.19e08	3.35e09	2.88e08	6.87e08	6.42e08	6.54e08	1.06e09	2.00e09
	Std	4.67e06	1.53e07	5.69e08	9.71e06	3.32e07	3.13e07	3.24e08	5.07e07	9.91e07
	t-test	-	**1.41e02***	**3.12e01***	**9.25e01***	**9.49e01***	**9.28e01***	**9.26e00***	**1.03e02***	**1.05e02***
F15	Mean	**2.80e03**	2.02e04	4.86e03	2.06e04	6.31e03	1.17e04	2.19e04	1.84e04	1.36e04
	Std	8.59e01	8.04e01	5.25e02	7.63e01	1.64e02	1.25e02	2.51e03	4.84e03	5.20e03
	t-test	-	**8.10e02***	**2.12e01***	**8.49e02***	**1.04e02***	**3.21e02***	**4.17e01***	**1.77e01***	**1.14e01***
F16	Mean	7.70e01	1.66e-07	3.17e02	8.51e-01	1.85e02	**9.70e-13**	7.93e02	7.88e02	1.58e02
	Std	1.13e01	8.99e-09	1.32e01	9.79e-01	2.59e01	4.50e-14	6.70e-01	2.14e01	1.33e01
	t-test	-	-3.73e01	**7.57e01***	-3.68e01	**2.09e01***	-3.73e01	**3.46e02***	**1.61e02***	**2.54e01***
F17	Mean	**7.35e04**	2.62e06	2.62e06	5.83e05	2.95e05	8.61e04	2.61e05	7.03e05	8.79e05
	Std	7.80e03	1.04e05	2.49e05	1.54e04	2.29e05	3.58e03	1.34e05	3.58e04	3.91e04
	t-test	-	**1.34e02***	**5.60e01***	**1.62e02***	**5.29e00***	**8.04e00***	**7.65e00***	**9.41e01***	**1.11e02***
F18	Mean	7.97e03	**5.22e03**	2.23e09	5.31e03	6.64e03	6.79e10	7.02e03	1.92e04	5.48e04
	Std	1.16e03	2.34e03	3.59e08	1.42e03	1.14e03	9.67e09	2.72e03	8.81e03	1.43e04
	t-test	-	-5.77e00	**3.40e01***	-7.95e00	-4.48e00	**3.85e01***	-1.76e00	**6.92e00***	**1.79e01***
F19	Mean	3.22e06	2.98e07	1.05e07	2.78e07	**2.02e06**	5.28e06	4.51e06	6.02e06	3.60e06
	Std	9.50e04	1.70e06	5.27e05	1.53e06	7.88e04	2.10e05	3.40e05	3.12e05	1.24e05
	t-test	-	**8.55e01***	**7.45e01***	**8.78e01***	-5.33e01	**4.90e01***	**2.00e01***	**4.70e01***	**1.33e01***
F20	Mean	5.44e03	**2.19e03**	2.54e09	2.79e03	2.73e03	1.65e11	4.50e03	4.94e03	8.36e03
	Std	3.43e02	2.51e02	5.03e08	2.45e02	1.89e02	1.37e10	5.25e02	4.86e02	6.49e02
	t-test	-	-4.19e01	**2.77e01***	-3.44e01	-3.79e01	**6.60e01***	-8.21e00	-4.60e00	**2.18e01***
w/l/t		-	11/9/0	20/0/0	8/11/1	11/7/2	17/3/0	15/3/2	16/3/1	18/2/0

the MA-SW-Chains, which has a relatively better performance than most of the rest methods to be compared, however, is still outperformed by our proposed algorithm on 11 functions.

Based the above results, we can conclude that our method possesses a good scalability in tackling problems with different dimensions or complexities and outperform related methods.

5. Conclusions

This paper reported a PSO incorporated with multi-level population sampling and dynamic *p*-learning mechanisms for LSGO. The MLS mechanism in the proposed method is devised to encourage exploration at the beginning of evolution while exploitation towards the end of evolution, thus appropriately searching the space. The dynamic *p*-learning scheme is designed to support efficient particle learning while preserving the swarm diversity during evolution. The performance of proposed algorithm has been evaluated via a series of experiments and compared with related methods. The results confirm the significance of the proposed mechanisms. By incorporating these mechanisms, our results reveal that the proposed algorithm is able to significantly outperform related methods for addressing LSGO.

For the future work, firstly, it will be interesting to design other schemes for partitioning the population, which can help the MLS to sample more appropriate subpopulations for evolution. This can be achieved, for example, by considering both the fitness of the individuals as well as their distances to the best individual in the population for partition purpose. Secondly, dynamically setting the parameter of L and φ in MLS could also be considered to improve the proposed MLS further. Finally, employing the proposed method to address optimization problems, especially for those involving complex search spaces such as network state estimation [41], [42], fault detection of dynamical systems [43], multi-senser filtering fusion [44] and data clustering [45], [46] could also be investigated.

References

[1] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proc. IEEE International Conference on Neural Network, vol. 4, pp. 1942-1948, 1995.

[2] M. Elbes, S. Alzubi and T. Kanan, "A survey on particle swarm optimization with emphasis on engineering and network applications," Evolutionary Intelligence, vol. 12, no. 2, pp. 113-129, 2019.

[3] A. Banks, J. Vincent and C. Anyakoha, “A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications,” Natural Computation, vol. 7, no. 1, pp. 109-124, 2008.

[4] Y. Gong, J. Zhang, H. Chung, W.N. Chen, Z.H. Zhan, Y. Li and Y.H. Shi, “An efficient resource allocation scheme using particle swarm optimization,” IEEE Transactions Evolutionary Computation, vol. 16, no. 6, pp. 801-816, 2012.

[5] W. Liu. Z. Wang, X. Liu, N. Zeng and D. Bell, "A Novel Particle Swarm Optimization Approach for Patient Clustering from Emergency Departments," IEEE Transactions on Evolutionary Computation, vol. 23, no. 4, pp. 632-644, 2019.

[6] N. Zeng, H. Qiu, Z. Wang, W. Liu, H. Zhang and Y. Li “A new switching-delayed-PSO-based optimized SVM algorithm for diagnosis of Alzheimer’s disease,” Neurocomputing, vol. 320, pp. 195-202, 2018.

[7] J.T. Tsai, P.Y. Chou and J.H. Chou, "Color Filter Polishing Optimization Using ANFIS With Sliding-Level Particle Swarm Optimizer," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 3, pp. 1193-1207, 2020.

[8] X. Song, Y. Zhang, Y. Guo, X. Sun and Y. Wang, "Variable-size Cooperative Coevolutionary Particle Swarm Optimization for Feature Selection on High-dimensional Data," IEEE Transactions on Evolutionary Computation, vol. 24, no. 5, pp. 882-895, 2020.

[9] Y. Yang and J.O. Pedersen, “A comparative study on feature selection in text categorization,” in Proc. ICML, pp. 412-420, 1997.

[10] W. Chen, J. Zhang, Y. Lin and E. Chen, “Particle swarm optimization with an aging leader and challengers,” IEEE Transactions Evolutionary Computation, vol. 17, no. 2, pp. 241-258, 2013.

[11] R. Cheng and Y. Jin, “A Competitive Swarm Optimizer for Large Scale Optimization,” IEEE Transactions on Cybernetics, vol. 45, no. 2, pp. 191-204, 2015.

[12] R. Cheng and Y. Jin, “A social learning particle swarm optimization algorithm for scalable optimization,” Information Science, vol. 291, no. 6, pp. 43-60, 2015.

[13] Q. Yang, W. Chen, T. Gu, H. Zhang, J.D. Deng, Y. Li and J. Zhang, “Segment-Based Predominant Learning Swarm Optimizer for Large-Scale Optimization,” IEEE Transactions on Cybernetics, vol. 47, no.9, pp. 2896-2910, 2016.

[14] Q. Yang, W. Chen, J.D. Deng, Y. Li, T. Gu and J. Zhang, “A Level-based Learning Swarm Optimizer for Large Scale Optimization,” IEEE Transactions Evolutionary Computations, vol. 22, no. 4, pp. 578-594, 2018.

[15] W. Guo, C. Si, Y. Xue, Y. Mao, L. Wang and Q. Wu, “A Grouping Particle Swarm Optimizer with Personal-Best-Position Guidance for Large Scale Optimization,” IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 15, no. 6, pp. 1904-1915, 2018.

[16] B. Tran, B. Xue and M. Zhang, "A New Representation in PSO for Discretization-Based Feature Selection," IEEE Transactions on Cybernetics, vol. 48, no. 6, pp. 1733-1746, 2018.

[17] Y. Cao, H. Zhang, W. Li, M. Zhou, Y. Zhang and W.A. Chaovalitwongse, "Comprehensive Learning Particle Swarm Optimization Algorithm with Local Search for Multimodal Functions," IEEE Transactions on Evolutionary Computation, vol. 23, no. 4, pp. 718-731, 2019.

[18] M.R. Bonyadi, "A Theoretical Guideline for Designing an Effective Adaptive Particle Swarm," IEEE Transactions on Evolutionary Computation, vol. 24, no. 1, pp. 57-68, 2020.

[19] Z. Yang, K. Tang and X. Yao, “Large scale evolutionary optimization using cooperative coevolution,” Information Science, vol. 178, no. 15, pp. 2986-2999, 2008.

[20] Z. Yang, K. Tang and X. Yao, “Multilevel Cooperative Coevolution for Large Scale Optimization,” IEEE conference on evolutionary computation, pp. 1163-1670, 2008.

[21] X. Li and X. Yao, “Cooperatively Coevolving Particle Swarms for Large Scale Optimization,” IEEE Transactions on Evolutionary Computation, vol. 16, no. 2, pp. 210-224, 2012.

[22] J.R. Jian, Z.H. Zhan and J. Zhang, "Large-scale evolutionary optimization: a survey and experimental comparative study," International Journal of Machine Learning and Cybernetics, vol. 11, no. 3, pp. 729-745, 2020.

[23] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in Proc. IEEE Congress on Evolutionary Computation, pp. 1671-1676, 2002.

[24] J.J. Liang, A.K. Qin, P.N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," IEEE Transactions Evolutionary Computation, vol. 10, no. 3, pp. 281-295, 2006.

[25] Z. Wang, Z. Zhan, S. Kwong, H. Jin and J. Zhang, "Adaptive Granularity Learning Distributed Particle Swarm Optimization for Large-Scale Optimization", IEEE Transactions on Cybernetics, vol. 51. no. 3, pp. 1175-1188, 2021.

[26] R. Lan, Y. Zhu, H. Lu, Z. Liu and X. Luo, "A Two-Phase Learning-Based Swarm Optimizer for Large-Scale Optimization," IEEE Transactions on Cybernetics, vol. 51, no. 12, pp. 6284-6293, 2020.

[27] F.V.D. Bergh and A.P. Engelbrecht, "A cooperative approach to particle swarm optimization," IEEE Trans. Evolutionary Computation, vol. 8, no. 3, pp. 225-239, 2004.

[28] X. Zhang, K. Du, Z. Zhan. S. Kwong, T. Gu and J. Zhang, "Cooperative Coevolutionary Bare-Bones Particle Swarm Optimization with Function Independent Decomposition for Large-Scale Supply Chain Network Design with Uncertainties," IEEE Transactions on Cybernetics, vol. 50, no. 10, 2020.

[29] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative Co-Evolution with Differential Grouping for Large Scale Optimization," IEEE Transactions on Evolutionary Computation, vol. 18, no. 3, 2014.

[30] Y. Sun, M. Kirley, and S.K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in Proceedings Genetic and Evolutionary Computation Conference, pp. 313-320, 2015.

[31] M.N. Omidvar, M. Yang, Y. Mei, X. Li and X. Yao, "DG2: A Faster and More Accurate Differential Grouping for Large-Scale Black-Box Optimization," IEEE Transactions on Evolutionary Computation, vol. 21, no. 6, pp. 929-942, 2017.

[32] Y. Sun, M. Kirley and S. K. Halgamuge, "A Recursive Decomposition Method for Large Scale Continuous Optimization," IEEE Transactions on Evolutionary Computation, vol. 22, no. 5, pp. 647-661, 2018.

[33] J.J. Liang and P.N. Suganthan, "Dynamic multi-swarm particle swarm optimizer with local search," in Proceedings IEEE Congress on Evolutionary Computations, pp. 522-528, 2005.

[34] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization," in Proceedings IEEE Congress on Evolutionary Computations, pp. 1-8, 2010.

[35] A. LaTorre, S. Muelas, and J. M. Pena, "Multiple Offspring Sampling in Large Scale Global Optimization," in Proceedings IEEE Congress on Evolutionary Computations, pp. 1-8, 2012.

[36] A. LaTorre, S. Muelas, and J. M. Pena, “Large scale global optimization: Experimental results with MOS-based hybrid algorithms,” in Proceedings IEEE Congress on Evolutionary Computations, pp. 2742-2749, 2013.

[37] S. Mahdavi, M.E. Shiri, and S. Rahnamayan, “Metaheuristics in large-scale global continues optimization: A survey,” Information Science, vol. 295, pp. 407-428, 2015.

[38] A. LaTorre, S. Muelas, and J.M. Pena, “A comprehensive comparison of large-scale global optimizers,” Information Science, vol. 316, pp. 517-549, 2015.

[39] K. Tang, X. Li, P.N. Suganthan, Z. Yang, and T. Weise, “Benchmark functions for the CEC 2010 special session and competition on large-scale global optimization,” Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, 2010.

[40] X. Li, K. Tang, M.N. Omidvar, Z. Yang, and K. Qin, “Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization,” Technical Report, Evolutionary Computation and Machine Learning Group, RMIT University, Australia, 2013.

[41] J. Hu, C. Jia, H. Liu, X. Yi and Y. Liu, "A survey on state estimation of complex dynamical networks," International Journal of Systems Science, vol. 52, no. 16, pp. 3351-3367, 2021.

[42] X.C. Jia, "Resource-efficient and secure distributed state estimation over wireless sensor networks: a survey," International Journal of Systems Science, vol. 52, no. 16, pp. 3368-3389, 2021.

[43] Y. Ju, X. Tian, H. Liu and L. Ma, "Fault detection of networked dynamical systems: a survey of trends and techniques," International Journal of Systems Science, vol. 52, no. 16, pp. 3390-3409, 2021.

[44] H. Geng, H. Liu, L. Ma and X. Yi, "Multi-sensor filtering fusion meets censored measurements under a constrained network environment: advances, challenges and prospects," International Journal of Systems Science, vol. 52, no. 16, pp. 3410-3436, 2021.

[45] W. Sheng, X. Wang, Z. Wang, Q. Li, Y. Zheng and S. Chen, “A differential evolution algorithm with adaptive niching and *k*-means operation for data clustering,” IEEE Transactions on Cybernetics, 2020, doi: 10.1109/TCYB.2020.3035887.

[46] X. Wang, Z. Wang, M. Sheng, Q. Li and W. Sheng, “An adaptive and opposite *k*-means operation based memetic algorithm for data clustering,” Neurocomputing, vol. 437, pp. 131-142, 2021.