Multi swarm optimization algorithm with adaptive connectivity degree

R. Vafashoar, M. R. Meybodi

Soft Computing Laboratory, Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran.

Tel: +98-21-64545120; Fax: +98-21-66495521; e-mail: (vafashoar, mmeybodi)@aut.ac.ir

Abstract

Particle swarm optimization algorithms are very sensitive to their population topologies. In all PSO variants, each particle adjusts it flying velocity according to a set of attractor particles. The cardinality of this set is a feature of neighborhood topology. In order to investigate this property exclusively, this paper defines the concept of connectivity degree for the particles and presents an approach for its adaptive adjustment. The presented approach is based on cellular learning automata (CLA). The entire population of the particles is divided into several swarms, and each swarm is resided in one cell of a CLA. Each cell of the CLA also contains a group of learning automata. These learning automata are responsible for adjusting the connectivity degrees of their related particles. This task is achieved through periods of learning. In each period, the learning automata realize suitable connectivity degrees for the particles based on their experienced knowledge. The empirical studies on a divers set of problems with different characteristics show that the proposed multi swarm optimization method is quite effective in solving optimization problems.

Keywords

Particle swarm optimization, Multi-swarm PSO, Global numerical optimization, Leaning automata (LA), Cellular learning automata.

1. Introduction

Particle swarm optimization (PSO) is a population-based stochastic search approach. It is inspired by the social behavior which exists in flocks of birds or schools of fish [1, 2]. A candidate solution to an optimization problem is called a particle in PSO terminology, and a group of these particles is called a swarm. The particles within a swarm utilize their individual and social experiences to update their flying velocity. The velocity of a particle guides its movements in the problem search space. PSO is simple, easy to implement, fast, and robust; owing to these attributes, it has attracted a great deal of interest during the recent decades. However, canonical PSO is highly prone to premature convergence and entrapment in local optima [3].

In the canonical PSO algorithm, each particle updates its velocity and position according to its own historical best position and the global best position experienced by the entire swarm. The use of this simple updating scheme is often blamed as a reason for premature convergence and entrapment in local optima. Accordingly, other updating rules and population topologies have been developed with the aim of enhancing the diversity of the population or balancing its explorative and exploitative behaviors [4-8]. Different problems may require specific updating rules or population topologies according to the properties of their fitness landscape [4-5]. Additionally, the use of different topologies and updating rules according to the state of the search process is also investigated in some literature [7-8].

In order to adjust the population topology in an intelligent manner, this paper introduces a CLA based multi-swarm optimization algorithm (CLAMS). Cellular learning automata (CLA) is a hybrid mathematical model for many decentralized problems [9-11]. This model can be considered as a network of learning automata which are interacting and cooperating on the basis of cellular automata rules. In the proposed PSO algorithm, the single swarm model is extended into an interacting multi-swarm model. Multi-swarm models maintain population diversity through the isolation of its sub-swarms. This diversity preserving mechanism is beneficial in the optimization of complex multimodal problems. The neighborhood size is one of the most important topological properties of PSO, which can greatly influence the exploration and exploitation characteristics of the algorithm [7]. In order to study this property in the introduced multi-swarm approach, the concept of connectivity degree is defined in this paper. We can balance the exploration and exploitation capabilities of the search procedure by appropriately tuning the defined connectivity degree. To adaptively control the connectivity degree of the particles based on the characteristics of their surrounding fitness landscape, this paper utilizes the CLA approach. In the introduced method, CLA learns an appropriate connectivity degree for each particle based on its individual and social performance. Through the empirical studies, it will be shown that: first, the selection of appropriate connectivity degrees can greatly enhance the performance of the proposed PSO algorithm;

second, CLA is quite effective in adaptively determining the connectivity degree of the particles. The proposed method also utilizes two exploration and exploitation mechanisms to further enhance its search procedure.
The rest of the paper is organized as follows: section 2 gives a brief overview on some preliminary concepts used in the paper; Sections 3 reviews some existing related works; section 4 gives the detailed description of the proposed approach; Experimental results and comparison to other algorithms are given in section 5; finally, section 6 contains conclusions.

2. Preliminaries

This section provides a brief overview of the concepts that are used in the paper. It starts with an introduction on particle swarm optimization; then, reviews the basic concepts of cellular learning automata.

2.1. Particle swarm optimization

PSO [1-2] executes its search in a specified *D* dimensional space through the accumulation of velocity and position information. Each particle iteratively updates its current position according to three types of information: its best previously visited position, known as its personal best or pbest; its moving velocity; and the best position experienced by the entire swarm, known as global best or gbest. Three *D* dimensional vectors $x_i = [x_{i1}, x_{i2}, \ldots, x_{iD}]$, $p_i = [p_{i1}, p_{i2}, \ldots, p_{iD}]$, and $V_i = [v_{i1}, v_{i2}, \ldots, v_{iD}]$ are used to denote the current position of the i^{th} particle in the swarm, its personal best position, and its current flying velocity, respectively. Also, the global best position of the swarm is represented by a *D* dimensional vector like $p_g = [p_{g1}, p_{g2}, \ldots, p_{gD}]$. The position and velocity of the i^{th} particle for the next iteration are updated using the following equations:

$$\begin{aligned} v_{id}(k+1) &\leftarrow \omega v_{id}(k) + c_1 r_1 (p_{id} - x_{id}(k)) + c_2 r_2 (p_{gd} - x_{id}(k)) \\ x_{id}(k+1) &\leftarrow x_{id}(k) + v_{id}(k+1) \end{aligned} \tag{1}$$

where r_1 and r_2 are two random numbers uniformly distributed in the interval $u[0,1]$; ω is called inertia weight; c_1 and c_2 are called acceleration constants.
After acquiring a new position for a particle, its personal best position is also updated, which may also affect the global best position of the swarm.

2.2. Learning automaton

A variable structure learning automaton can be represented by a sextuple like $\{\Phi,\alpha,\beta,A,G,q\}$ [12], where Φ is a set of internal states; α is a set of outputs or actions; β is a set of inputs or environmental responses; A is a learning algorithm; $G(.)$: $\Phi \rightarrow \alpha$ is a function that maps the current state into the current output; and q is a probability vector that determines the selection probability of a state at each step. There is usually a one to one correspondence between Φ and α; as a result, the two terms can be used interchangeably.

The objective of a learning automaton is to identify the best action which maximizes the expected received payoff from the environment [13]. To this end, the learning automaton selects an action according to its action probability distribution. Then, the LA applies this action to the environment. The environment measures the favorability of the received action and responds the LA with a noisy reinforcement signal (response) β. The learning automaton uses this response to adjust its internal action probabilities via its learning algorithm. The environmental response to a selected action like a_i at step k is represented by $\beta\in\{0,1\}$, where 0 and 1 are used as pleasant and unpleasant responses, respectively. The internal action probability can be updated according to equation 2 when the response is pleasant and according to equation 3 when it is unpleasant.

$$q_j(k+1)=\begin{cases} q_j(k)+a(1-q_j(k)) & if\ \ i=j \\ q_j(k)(1-a) & if\ \ i\neq j \end{cases} \tag{2}$$

$$q_j(k+1)=\begin{cases} q_j(k)(1-b) & if\ \ i=j \\ \dfrac{b}{r-1}+q_j(k)(1-b) & if\ \ i\neq j \end{cases} \tag{3}$$

where a and b are called reward and penalty parameters, respectively, and r is cordiality of the action set. For $a = b$, the learning algorithm is called $L_{R\text{-}P}$; when $b << a$, it is called $L_{R\text{-}\varepsilon P}$; when b is zero, it is called $L_{R\text{-}I}$.

2.3. Cellular learning automaton

Integrating the learning capabilities of learning automata into a cellular automaton, cellular learning automata was introduced by Beigy and Meybodi [9-11, 14]. In this model, each cell of a cellular automaton is occupied with one or more learning automata. The learning automata residing in a cell define the state of that cell. Like cellular automata, a CLA rule controls the behavior of each cell. This rule determines the acceptability of the selected actions.

A cellular learning automata starts from some initial state (it is the internal state of every cell). At each time step, each learning automaton selects an action according to its probability vector and performs this action on the environment. Then, the local rule of the cellular learning automata determines a reinforcement signal to each learning automaton. Based on the received signal, each learning automaton updates its internal probability vector. This procedure continues until a desired result is obtained.

3. Related works on PSO

After its initial development by Kennedy et al., many enhanced variants of PSO have been developed to overcome its deficiencies such as premature convergence and stagnation. Several of these variants are focused on parameter adaption [15-16], primarily concerning the inertia weight [2] and the acceleration coefficients [1]. Hashemi and Meybodi investigated the use of learning automata for adaptive parameter selection in PSO [17]. They introduced two classes of learning automata based methods for adjusting the inertia weight and acceleration coefficients during the search process. In the first class, the parameters are adopted based on the individual performance of each particle and are utilized for the associated particle itself. In the second class, all individuals share the same parameter settings, and these settings are adapted based on the performance of the entire swarm. Adaptive PSO (APSO) utilizes population distribution and fitness information to estimate the evolutionary state of the swarm [18]; using this estimated state, APSO adjusts its inertia weight and acceleration coefficients. Xu proposed a method for parameter adaption based on the velocity information [19]. He used this information for tuning the inertia weight of the swarm.

Some other approaches improve PSO by introducing certain learning mechanisms into the algorithm [17]. Self-learning particle swarm optimizer (SLPSO) uses four different operators which are specifically designated for exploitation, exploration, convergence, and jumping out [4]. The decision of which operator to be utilized is made with the aid of reinforcement learning techniques. A very similar approach is practiced in ALPSO [5], where probability matching is used for the adaption of four developed updating rules. Piperagkas et al. incorporated reinforcement learning into the PSO algorithm for dealing with optimization problems in noisy environments [20]. A modified Bare bones PSO, which uses adaptive update distributions, was introduced by Vafashoar and Meybodi [21]. They have developed four updating strategies based on Gaussian and multivariate Gaussian distributions. The introduced model is integrated with a CLA, which adaptively selects an appropriate updating rule for each particle based on the characteristics of the fitness landscape. They also introduced a technique for the adaption of the covariance matrices of Multivariate Gaussian distributions.

The population topology can significantly affect the performance of a PSO algorithm. In fully informed PSO [6], a particle utilizes the information from all its neighbors to update its velocity. AHPSO [22] uses an adaptive neighborhood topology. Its particles are arranged into a tree structure with the better ones in the higher levels, and their positions are updated in the tree hierarchy based on their historical best fitness values. PSO with adaptive time-varying topology connectivity updates the topology of its particles according to their performance [23]. Simple topology adaption mechanisms like increasing or decreasing topology connectivity are also suggested in some literature [7-8]. Multi population is another mechanism that improves evolutionary algorithms by maintaining their population diversity. Although diversity may be low within each deme, but the natural distance among different demes ensures the overall population diversity [26]. Chen et al. introduced a multi swarm PSO in which each particle, in addition to its intra-swarm attractors, is affected by the best performer in its neighboring swarms [27]. Multi-swarm and multi-best algorithm randomly divides an initial population into multiple swarms [28]. Each swarm is evolved separately for a specified number of iterations. Then, all swarms are combined into a single one. The algorithm updates each particle based on “multi-gbest and multi-pbest” scheme. MCPSO is a multi-swarm PSO based on a master–slave model [29]. The slave swarms are evolved independently, while the master swarm is updated based on its own knowledge as well as the knowledge from the slave swarms. Zhang and Ding proposed a multi-swarm self-adaptive and cooperative particle swarm optimization (MSCPSO), which uses several strategies and updates its inertia weight based on a fitness adaptive policy [30]. In cooperative velocity updating model based particle swarm optimization (CVUPSO-R), several swarms are evolved based on their interactions [31]. The algorithm also utilizes mechanisms such as random individual generation to enhance its diversity and to speed up its search process. Another multi-swarm algorithm is presented in [32], which uses perturbed global best positions in the update rules for maintaining its population diversity. In SuPSO, particles are divided into three groups based on the different tasks they are assigned to perform. According to their designated tasks and groups, the particles are categorized as exploitation, divers, and support [33].

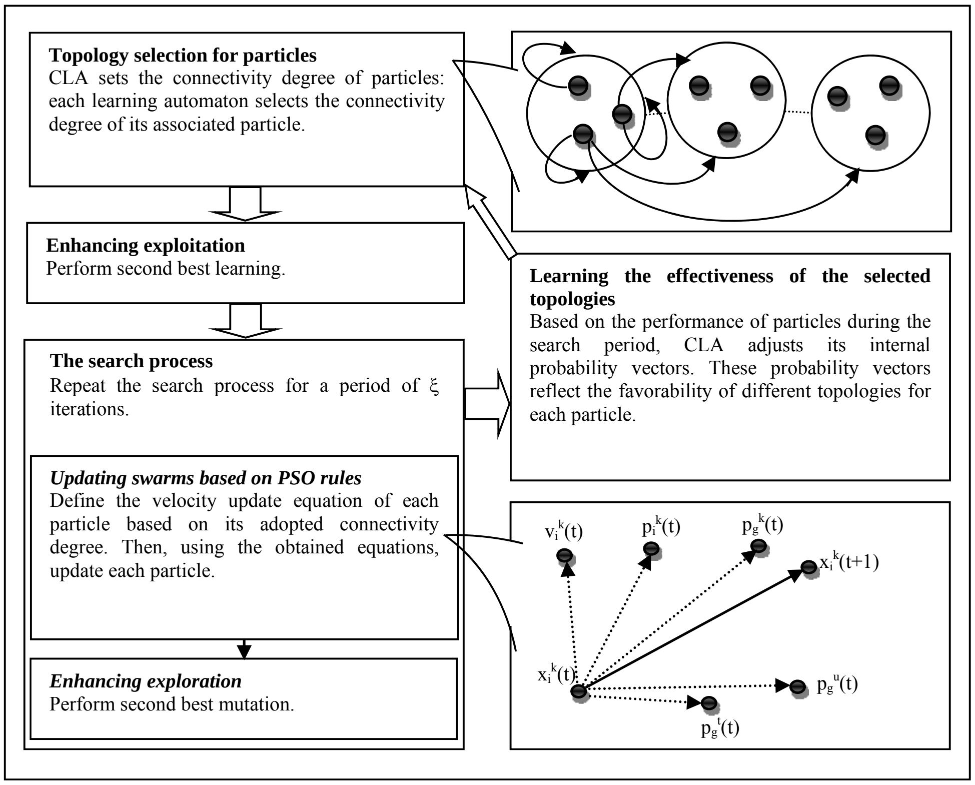
4. CLA based multi-swarm optimization algorithm (CLAMS)

The particles in CLAMS are divided into several swarms, and each particle is updated based on the information obtained from its parent swarm as well as the nearby ones. The global best position of each nearby swarm would be included in the update equation of a particular particle to influence its movement. In this way, particles can acquire information from their nearby niches. The swarms or particles that directly affect a single particle in an update step constitute its neighborhood. The size of this neighborhood controls the convergence speed and the exploration ability of the proposed approach. In the smallest neighborhood, a particle is updated in isolation from its nearby swarms. Hence, it can effectively exploit the promising areas covered by its parent swarm. On the other hand, the tendency of particles toward their nearby swarms grows as the size of the described neighborhood increases. As a result, the convergence speed would be increased since the particles are utilizing more similar information sources for adjusting their movements. Also, the particles that are trapped in poor local optima may be guided to more appropriate areas; however, they may lose track of the unexploited fine regions around their parent swarms. In order to proceed, we first define the following concept to formalize the aforementioned ideas.

Particle connectivity degree: the connectivity degree of a particle like X during time t is the number of particles that are connected to X at t. These connected particles influence the velocity of X during the time; therefore, X “flies” toward them.

Since the connectivity degree describes the neighborhood topology, the terms particle connectivity degree and particle topology will be used interchangeably. As described earlier, the neighborhood topology of particles have an intense impact on the search behavior of the proposed approach. Tuning the connectivity degree of a particle is not an easy task as it relies on the fitness landscape and the distribution of particles. Also, as the location of particles changes during the search process, different connectivity degrees would be required to achieve an efficient search. An approach based on cellular learning automata is developed in this paper for adjusting the connectivity degree of particles dynamically. CLA enables the particles to adapt their topology during the search process based on their previous experiences in the fitness landscape. Consequently, it balances intra-swarm and inter-swarm interactions.

Each cell of the CLA embeds a swarm of particles and a group of learning automata. There is a one to one correspondence between the learning automata and the particles within each cell. Each cycle of CLAMS starts with topology selection for the particles. In this step, CLA sets the connectivity degree of the particles. Afterwards, each particle employs its selected connectivity degree during the search process. The role of a learning automaton is to understand the effectiveness of different topologies for its coupled particle. The learning automaton achieves this purpose by observing the performance of the particle during the search process and employing the gained experience of other nearby learning automata in the CLA. The searching process consists of ξ update steps, ξ will be referred to as learning period. Because of the stochastic nature of the PSO algorithm, it is probable that an appropriate selected topology leads to low fitness positions for a particle or vice versa. Therefore, CLAMS uses ξ search steps, rather than a single one, to obtain a better understanding of the selected topologies prior to their evaluation. After this search process, the effectiveness of the adopted topologies will be evaluated by the CLA. Then, the internal probability vectors of the learning automata will be updated accordingly. The block diagram of the proposed approach is illustrated in Figure 1, and its building blocks are thoroughly explained in the subsequent sections.



Topology selection for particles
CLA sets the connectivity degree of particles: each learning automaton selects the connectivity degree of its associated particle.

Enhancing exploitation
Perform second best learning.

The search process
Repeat the search process for a period of ξ iterations.

Updating swarms based on PSO rules
Define the velocity update equation of each particle based on its adopted connectivity degree. Then, using the obtained equations, update each particle.

Enhancing exploration
Perform second best mutation.

Learning the effectiveness of the selected topologies
Based on the performance of particles during the search period, CLA adjusts its internal probability vectors. These probability vectors reflect the favorability of different topologies for each particle.

Figure 1. The block diagram of the proposed method.

4.1. Topology selection for particles

In CLAMS, a CLA is responsible for selecting a promising topology for each particle. Each cell of the CLA is composed of two parts: a swarm of the population, which itself holds *N* particles; and a set of *N* learning automata, one learning automaton for each particle. Each learning automaton controls the connectivity degree of its associated particle. Figure 2 shows a schematic representation of these ideas.

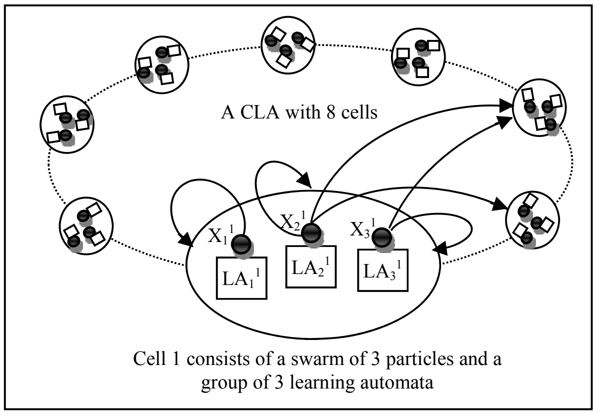

A CLA with 8 cells
X_1^1
X_2^1
X_3^1
LA_1^1
LA_2^1
LA_3^1
Cell 1 consists of a swarm of 3 particles and a group of 3 learning automata

Figure 2. A CLA with 8 cells: each cell embeds a swarm of particles and a learning automaton for each particle.

As discussed previously, each particle can "fly" toward the global best position of a nearby swarm. The number of nearby swarms, which will be denoted by γ, defines the maximum connectivity degree of a particle. Each learning automaton has a set of γ+1 actions, where each action corresponds to one of the γ+1 allowable connectivity degrees. It should be noted that a particles utilize only its intera-swarm information when its connectivity degree is set to its minimum value (i.e. 2). During the topology selection step, each learning automaton selects an action according to its internal probability vector. The selected action of a learning automaton defines the connectivity degree of its associated particle.

4.2. Enhancing exploitation via Second best learning

Maintaining an archive of promising visited locations would be beneficial as the algorithm can possess more eligible historical information about the fitness landscape. Based on this idea, the second best positions are employed in the proposed method. Each particle keeps track of the best visited locations in its personal best and second best positions. The second best position of a particle is a position that has been experienced by the particle itself and did not become the particle's pbest; additionally, it is the fittest position satisfying this condition. CLAMS uses the second best positions to enhance its exploration and exploitation abilities via two developed mechanisms: Second best mutation and Second best learning.

During the search process, which consists of ξ updating steps, particles move in the fitness landscape and their positions are altered accordingly. Consequently, they may forget some of their promising visited locations and reside in inappropriate areas of the search space. When a particle is positioned in an unpleasant location, it might require much effort to find its way back to a suitable location. The search process will be enhanced if particles revisit some of their historical promising locations and exploit them further. The second best learning is developed for this purpose. It allows the particles to revisit their favorable spotted locations and reexamine them furthermore. After every learning period, each particle is affected by the second best learning operator. The operator simply sets the current position of the particle to its second best position.

4.3. The search process of CLAMS

The search phase of the algorithm consists of ξ updating steps. During each updating step the swarms are evolved based on PSO updating rules and second best mutation strategy. Particles utilize their selected topologies in the topology selection phase during the search procedure. As stated previously the search process consists of ξ updating steps, rather than a single one, for improving evaluations made by the CLA.

4.3.1. Updating swarms based on PSO rules

During each step of the search process, the velocity of the particles is updated according to their adopted topologies. The number of informers involved in the velocity update equation of a particular particle is related to its selected connectivity degree. Several implementations of velocity updating rule based on the defined connectivity degree are possible. Herein, we propose two velocity update rules and discuss their characteristics. Considering a particle like X_i^k, its first velocity update rule will be defined as follows:

$$v_i^k(t) = \begin{cases} Wv_i^k(t-1) + \dfrac{\chi}{a_i^k(t)+1}\sum_{j=1}^{a_i^k(t)+1} r_j \square \left(b_j - x_i^k(t-1)\right) & \textit{if } a_i^k(t) > 1 \\ Wv_i^k(t-1) + \dfrac{\chi}{2} r_1 \square \left(p_i^k(t-1) - x_i^k(t-1)\right) & \\ \qquad + \dfrac{\chi}{2} r_2 \square \left(p_g^k(t-1) - x_i^k(t)\right) & \textit{if } a_i^k(t) = 1 \end{cases}$$

$$b = randperm\left(p_i^k(t-1), p_g^k(t-1), p_g^{N(1,k)}(t-1), p_g^{N(2,k)}(t-1), \ldots, p_g^{N(\gamma,k)}(t-1)\right) \qquad (4)$$

$$r_j = \begin{cases} \left[r_{j,1}, r_{j,2}, \ldots, r_{j,D}\right]^D ; r_{j,\mathrm{h}} \sim u(0,1) & \textit{with probability } 0.5 \\ r\left[1,1,\ldots,1\right]^D ; r \sim u(0,1) & \textit{otherwise} \end{cases}$$

Here, $x_i^k(t)$ represents the position of the ith particle belonging to the kth swarm; $v_i^k(t)$ and $p_i^k(t)$ are its flying velocity and personal best position, respectively. $a_i^k(t)$ stands for the adopted connectivity degree of the particle (or the selected action of its associated learning automaton) during time t. It should be noted that the connectivity degree of particles remains unchanged for a period of ξ search steps between two consecutive action selection phases. $p_g^k(t)$ represents the global best position of the kth swarm. $N(j,k)$ defines a function which returns the jth swarm in the neighborhood of swarm k, and $randperm(A)$ is a function that arranges the elements of A in a random order. W and χ are the inertia weight and the acceleration constant, respectively. Finally, Θ stands for the Hadamard operator, which is the element by element multiplication of two matrices.

Considering Equation 4, there are some points worth noting. First, a particle only employs intra-swarm information when its connectivity degree is set to 2 (or alternatively the first action is selected by its associated LA). Under this connectivity condition, the particle performs its search locally. When the connectivity degree of a particle becomes larger than one, it will be impacted by some attractors out of its parent swarm. In this case, it is also possible that the particle is merely guided by the outer-swarm elements, which helps it in escaping from poor local optima. The second point is about the random element in equation 4 which scales the summation terms in the update rule, i.e. *r*. When this element is a random scalar, the velocity update rule is called linear. On the other hand, the canonical velocity update rule normally uses a vector of random scalar numbers. Both updating rules have their benefits and drawbacks, which are studied in some literature like [34]. Equation 4 uses a combination of scalar and vector random numbers to utilize the benefits of both approaches.

The described velocity update rule can be modified in a number of ways in order to induce different behavioral dynamics in the particles. A variant of Equation 4 can be obtained by replacing "*randperm*" with a function that returns a random combination of its arguments with repetitions allowed. In this case, particles can move with larger steps in specific directions, which results in a high explorative behavior.

$$
v_i^k(t) = \begin{cases} Wv_i^k(t-1) + \dfrac{\chi}{a_i^k(t)+1} \sum_{j=1}^{a_i^k(t)+1} r_j \odot \left(b_j - x_i^k(t-1)\right) & if\ a_i^k(t) > 1 \\ Wv_i^k(t-1) + \dfrac{\chi}{2} r_1 \odot \left(p_i^k(t-1) - x_i^k(t-1)\right) & \\ \qquad + \dfrac{\chi}{2} r_2 \odot \left(p_g^k(t-1) - x_i^k(t)\right) & if\ a_i^k(t) = 1 \end{cases}
$$

$$
b_j \sim u\left\{p_i^k(t-1), p_g^k(t-1), p_g^{N(1,k)}(t-1), p_g^{N(2,k)}(t-1), \dots, p_g^{N(\gamma,k)}(t-1)\right\} \tag{5}
$$

$$
r_j = \begin{cases} \left[r_{j,1}, r_{j,2}, \dots, r_{j,D}\right]^D ; r_{j,\text{h}} \sim u(0,1) & with\ probability\ 0.5 \\ r\left[1,1,\dots,1\right]^D ; r \sim u(0,1) & otherwise \end{cases}
$$

Here, b_j is obtained randomly according to a uniform distribution over the set of attractors.

4.3.2. Second best mutation (SBM)

PSO update rules affecting a particle in all coordinates would likely spoil its promising information in some dimensions. As a result, using canonical PSO update rules, it would require much effort to refine a particle when its personal best position has eligible information in most dimensions except some few ones. In order to alleviate this problem, we can utilize mutation mechanisms which do not inflict sever changes in the particles. The second best mutation is developed for this purpose. This mutation mechanism only affects the global best particle of each swarm.

In order to generate a mutant vector, SBM affects the personal best position of a particle in two random dimensions; if the newly generated mutant vector is evaluated to be fitter, it will replace the personal best position of its parent particle. The second best mutation operator changes the first selected dimension of a particle like X_i^k in the following way: first, a random particle is selected from a nearby swarm; then, the personal best position of X_i^k is replaced by the second best position of the random particle in the corresponding mutation dimension. As the particles within a same swarm are liable to carry comparable information after some generations, the random particle for the mutation is selected from a distinct swarm. The utilization of second best positions instead of personal best positions helps in maintaining the diversity of the swarms. Additionally, a second best position may possess useful information in some certain coordinates in spite of its relatively lower fitness value. The second dimension of the particle to be affected by the SBM operator is perturbed using Cauchy distribution. This kind of perturbation enables the particle to escape from local optima and to explore areas beyond its swarm limits. The parameters of the Cauchy distribution are determined based on the personal best locations of the particle itself and the aforementioned randomly selected particle. A pseudo code for the described operator is given in Figure 3.

```
Second best mutation (SBM operator)
Input:
K: swarms index.
1.    Select the best particle of the swarm.
2.    Let the personal best location of the selected particle be $p_i^k$.
3.    Set $u = p_i^k$.
4.    Select two random dimensions $s, t \in \{1, 2, \ldots, D\}$.
```

```
5.  Select a random particle like $X_j^l$ from a nearby swarm.
6.  Modify $u$ in its sth dimension based on a Cauchy distribution: $u_s = u_s + |p_{js}^l - u_s| C(1)$.
7.  Modify $u$ in its tth dimension based on the second best position of $X_j^l$: $u_t = sb_{jt}^l$.
8.  Replace $p_i^k$ with $u$ if it has a better fitness value.
```

Figure 3. A pseudo code for second best mutation

4.4. Learning the effectiveness of the selected topologies

The learning automata control the topology of their attached particles. Each learning automaton has a set of $\gamma+1$ actions, where each action corresponds to one of the "$\gamma+1$" permissible values for the connectivity degree. At the start of each learning period, the learning automaton associated with a particular particle selects one of its actions according to its action probability vector. Then, the particle uses the related topology for a period of ξ iterations. Meanwhile, the behavior of the particle under the selected topology is observed, and the effectiveness of the selected action is measured accordingly. Using a relatively large learning period (ξ), the selected actions can be monitored for a long time; consequently, better performance evaluations can be achieved. But, a large learning period has two disadvantages: relatively few actions can be selected during the course of evolution; as a result, few LA update steps can be performed which itself results in a slow learning. Second, inappropriate selected actions would be given more time to corrupt the swarms. Therefore, proper tuning of the learning period has a major impact on the performance of the proposed method. In our experiments, values of ξ between 5 and 10 exhibited appropriate behavior.

After a learning period, each learning automaton in a cell receives a reinforcement signal from its environment which reflects the favorability of its selected action. Then, based on this signal, its initial probability vector will be updated using Equations 2 or 3. Similar to the velocity update rules, there are several ways to define reinforcement signals. A reinforcement signal tries to conjecture the effectiveness of a selected action; a suitable definition should exhibit this effectiveness as accurate as possible in order to steer eligible topology selection. In order to define the re-enforcement signals formally, we first define two terms of average behavior and average improvement as follows:

Average behavior: the average behavior of a particle like X_i^k during a learning period from time t_1 to t_2 is the average fitness of its observed positions during the period, which will be denoted by $FM(X_i^k, t_1, t_2)$:

$$FM\left(X_i^k,t_1,t_2\right)=\frac{1}{\xi}\sum_{t=t_1}^{t=t_2} f\left(x_i^k(t)\right) \quad (6)$$

Average improvement: the average improvement of a particle like X_i^k during a learning period from time t_1 to t_2 is the amount of improvement in the particle's average behavior during the period, which will be denoted by $D(X_i^k,t_1,t_2)$:

$$D(X_i^k,t_1,t_2)=f(sb_i^k(t_1-1))-FM_i^k(t_1,t_2) \quad (7)$$

Here, sb_i^k represents the second best position of X_i^k. It should be noted that the second best learning operator changes the position of each particle to its second best location at the start of each learning period (Figure 1). These two defined criteria can be used to measure the performance of a particle. In order to evaluate the effectiveness of the selected actions, the performance of their associated particles is compared with the performance of other particles in the closest nearby swarms; the set of these particles will be defined as follows:

$$NX^k=\left\{X_j^l \mid l\in\left\{N(1,k),N(2,k)\right\}\right\} \quad (8)$$

Finally in the CLA approach, each learning automaton learns from the experiences of other nearby entities. For this purpose, we wish to encourage the learning automata to select the most promising actions in their neighborhoods. A promising action will be defined as follows:

$$a_*^k(t_1,t_2)=a\left(\arg\max_{X_j^l\in NX^k}(D(X_j^l,t_1,t_2)),t_1\right) \quad (9)$$

where $a(X_i^j, t_1) = a_i^j(t_1)$ is the selected action of the ith learning automaton in the jth cell. Based on these definitions, the reinforcement signal for a particle like X_i^k will be defined as follows:

$$\beta_i^k=\begin{cases} 0 \;\; if & \begin{pmatrix} D(X_i^k,t_1,t_2)>median\left(\left\{D(X,t_1,t_2)\mid X\in NX^k\right\}\right) and \\ FM(X_i^k,t_1,t_2)>median\left(\left\{FM(X,t_1,t_2)\mid X\in NX^k\right\}\right) \end{pmatrix} \\ & or\left(a\left(X_i^k,t_1\right)=a_*^k(t_1,t_2)\ and\ D(X_i^k,t_1,t_2)>0\right) \\ 1 \;\; otherwise & \end{cases} \quad (10)$$

Here, median is the statistical median operation. The reinforcement signals defined using Equation 10 utilize both personal experience of a learning automaton as well as the social experience of its nearby learning automata.

A general pseudo code for the described CLAMS algorithm consisting all of its building blocks is given in Figure 4.

```
CLA based multi swarm algorithm (CLAMS)
Input:
$\gamma$: Maximum number of neighboring swarms.
$M$: number of swarms.
$N$: number of particles in each swarm.
$D$: problem dimension
Initialization:
•   Initialize the position and velocity of each particle randomly in the allowable domains.
•   Set the current location of each particle as its personal best.
•   Select the particle with the highest fitness value in each swarm as the global best of that swarm.
•   Initialize a CLA with $M$ cells, each residing $N$ learning automata. Each LA has $\gamma$+1 actions corresponding to $\gamma$+1
    allowable topologies.
•   Set $t = 0$.
Repeat until a termination condition is satisfied:
1.  Action selection: Each learning automaton selects an action based on its internal probability vector. Assume
    $a_i^j(t+1)$ represents the selected action for the $i$th learning automaton in the $j$th cell.
2.  Consider $a_i^j(t+2) = a_i^j(t+3) = \ldots = a_i^j(t+\xi) = a(X_i^j, t+1) = a_i^j(t+1)$.
3.  Perform Second best learning: Set le current position of each particle to its second best position.
4.  Repeat for $\xi$ iterations
    1.  Set $t = t + 1$.
    2.  Update the velocity of each particle using Equation 4 (or 5).
    3.  Update the position of each particle according to Equation 1.
    4.  Evaluate each obtained position, and update the personal best and second best positions of each particle
    5.  Update the global best position of each swarm.
    6.  For each swarm $k$ perform the second best mutation as illustrated in Figure 3.
5.  Using Equation 10, generate a reinforcement signal for each learning automaton.
6.  Using the obtained signals, update each learning automaton according to Equations 2 and 3.
```

Figure 4. A pseudo code for CLAMS.

4.5. Time complexity of CLAMS

The main criteria in the time complexity analysis of stochastic optimization methods is the "number of fitness function evaluations" since fitness evaluation is considered the most time computing operation in the optimization process. Accordingly, optimization methods are usually compared using an equal number of fitness evaluations. Each search step of CLAMS requires some extra operations in comparison to the canonical PSO. Hence, we can analyze the complexity of CLAMS according to the overhead of these extra operations. Table 1 gives the time complexity of different building blocks of CLAMS in one search step (Only the extra operations are considered).

Table 1. Time complexity of different building blocks of CLAMS

operation	Time complexity	description
Action selection	O(MN(λ+1)/ξ)	Is repeated once in every ξ search steps. MN: Number of learning automaton in the CLA 'λ+1': action set size of each LA
Second best learning	O(MND/ξ)	Simply sets the current position of each particle to its second best position.
Learning the effectiveness of the selected topologies	O(MN+MN(λ+1)/ξ)	Computing average improvement: O(MN) Computing median improvement in the neighborhood: O(2MN/ξ) Determining the promising action in the neighborhood: O(2MN/ξ) Updating learning automata: O(MN(λ+1)/ξ)
Updating velocities	O(MND(λ+2))	The maximum number of particles affecting a single particle is

		(λ+2).
Updating second best positions	O(MND)	Simply changes a D dimensional vector. Can occur at most MN times in a search step.
Second best learning	O(M)	Only affects the global best position of each swarm in two random dimensions.

Considering the time complexity of different building blocks of CLAMS, the overhead of the proposed method in comparison to canonical PSO is not considerable. This observation was somehow predictable since learning automata are simple decision making units that operate under simple rules. In fact, the simplicity and low computational cost of LA approach is one of the main reasons for its popularity. Analogous discussions can be made concerning the CLA approach since CLA only consists of learning automata and simple local rules.

5. Experimental results

In order to evaluate the performance of the proposed approach and to investigate its behavior, CLAMS is examined on two sets of benchmark problems. The first set includes some classical well-known functions, which are given in Table 2. Owing to their relatively straightforward definitions, their characteristics are rather easy to perceive. Accordingly, we have selected this set of benchmarks to exhibit the behavior of the proposed method. In order to demonstrate its efficiency on this set of problems, CLAMS is compared with some other state of the art approaches. Due to their specific properties, these peer methods have become desirable tools for solving many classes of optimization problems. Consequently, they can unveil the flaws of the introduced method and demonstrate its effectiveness in a clear way.

The second test set consists of CEC2013 benchmark problems [35]. CEC2013 presents a comprehensive set of optimization problems, which are useful in the verification of optimization approaches. In order to validate the effectiveness of the introduced method on these benchmarks, the suggested experiments in [35] are conducted to compare CLAMS with some recent population-based optimization methods. As CEC2013 benchmarks are completely defined and categorized in [35]; hence here, we only give the definitions and the properties of the first benchmark set.

The test functions of the first problem set are taken from [36-37] and are presented in Table 2. These benchmarks can be categorized into three groups [36]: unimodal, unrotated multimodal, and rotated multimodal. Unimodal functions are rather easy to optimize and are mostly used to demonstrate the convergence speed of the algorithms. Unrotated multimodal functions mostly possess a large number of local optima. They provide a good means for investigating the ability of optimization algorithms in escaping from poor local optima. One issue with many test functions is their separability, meaning that they can be solved using *D* (the number of dimensions) independent searches. To alleviate this problem, we can make a function inseparable by means of rotation. To rotate a function $f(x)$, the original vector x is left multiplied with an orthogonal matrix M: $y = M \times x$; then, the newly generated vector y is used to calculate the fitness of the associated individual [36]. Herein, Salomon's method will be used to generate the orthogonal matrices [38]. Most of the benchmarks presented in Table 2 have their optimum point in the center of the search space. These benchmarks are also symmetric with the property that their local optima are around their global optimum. These characteristics result in a symmetric distribution of the population in the stochastic optimization algorithms. Therefore, when their individuals are attracted to each other by means of evolutionary operators, they may accidently fall near the global optimum point. In order to deal with this issue, we employ the shifted version of the given functions. Consequently, $f(x-\alpha)$ is used instead of $f(x)$ as the fitness function. We set α to 1/10 of the search range. This shifting is only used for the functions with their optimum point at the origin.

Table 2. Classical benchmark problems along with their General attributes

function	range	Opt val	precision
$f_1 = 418.9829 \times D + \sum_{i=1}^{D}\left(-x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$	$[-500,500]^D$	0	0.001
$f_2 = \sum_{i=1}^{D}\left(-x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[-5.12,5.12]^D$	0	0.001
$f_3(X) = f_2(Z)$, $z_i = \begin{cases} x_i & \lvert x_i \rvert < \frac{1}{2} \\ \frac{round(2x_i)}{2} & \lvert x_i \rvert \geq \frac{1}{2} \end{cases}$	$[-5.12,5.12]^D$	0	0.001
$f_4 = -20\exp\left[-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right] - \exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right] + 20 + \exp(1)$	$[-32,32]^D$	0	0.001
$f_5 = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600,600]^D$	0	0.00001

$f_6 = \frac{\pi}{D}\left\{\begin{matrix} 10\sin^2(\pi y_1) + (y_D - 1)^2 + \\ \sum_{i=1}^{D-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right]\end{matrix}\right\}$ $+\sum_{i=1}^{N} u(x_i, 10, 100, 4)\ , \quad y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - 1)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50,50]^D$	0	0.001
$f_7 = 0.1\left\{\begin{matrix} \sin^2(3\pi x_1) + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right] \\ +\sum_{i=1}^{D-1}(x_i - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right]\end{matrix}\right\}$ $+\sum_{i=1}^{D} u(x_i, 10, 100, 4)$	$[-50,50]^D$	0	0.001
$f_8 = \sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	0	0.001
$f_9 = \sum_{i=1}^{D}\lvert x_i\rvert + \prod_{i=1}^{D}\lvert x_i\rvert$	$[-10,10]^D$	0	0.001
$f_{10} = \sum_{i=1}^{D}\left[\sum_{j=1}^{i} x_j\right]^2$	$[-100,100]^D$	0	0.001
$f_{11} = \sum_{j=1}^{D-1}\left[100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2\right]$	$[-100,100]^D$	0	30
f_{12}: $f_2(y)$, $y = M \times x$	$[-5.12,5.12]^D$	0	50
f_{13}: $f_5(y)$, $y = M \times x$	$[-600,600]^D$	0	0.00001
f_{14}: $f_4(y)$, $y = M \times x$	$[-32,32]^D$	0	0.001
f_{15}: $f_3(y)$, $y = M \times x$	$[-5.12,5.12]^D$	0	50
f_{16}: $f_1(y)$, $y = M \times x$	$[-500,500]^D$	0	3000

Table 3. Compared stochastic optimization approaches for the classical benchmark set

Algorithm	Base method	Settings
CLPSO	Particle Swarm Optimization algorithm (PSO)	$c = 1.49445$, $w \in [0.4, 0.9]$
FIPS	PSO	$\varphi = 4.1$, $\chi = 0.7298$
FPSO	PSO	$\varphi = 4.0$, $w \in [0.4, 0.9]$
APSO	PSO	$1.5 < c_1, c_2 < 2.5$ and $c_1 + c_2 < 4$, $w \in [0.4, 0.9]$
MPEDE	Differential evolution (DE)	λ_1, λ_2, $\lambda_3 = 0.2$; $NP = 250$; $ng = 20$

5.1. Compared methods

CLAMS is compared on the first benchmark set with five other population based optimization methods including: CLPSO [36], FIPS [6], FPSO [7], APSO [18], and MPEDE [39]. Their configurations are taken from the related literature, and the same settings suggested by their authors are used in the experiments; these settings are summarized in Table 3. Fully informed PSO is based on the idea of using multiple informers for each particle. It utilizes a U-ring topology for its population. FPSO starts with a fully connected topology: each particle uses the personal best position of all other particles to update its velocity. Then, the connectivity of the particles decreases gradually during the search process until FPSO ends up with a ring topology. APSO adaptively changes the PSO parameters based on the evolutionary stage of its search. In CLPSO, each particle uses a different set of attractors in each dimension to update its flying velocity. A particle changes its attractor set when it fails to improve for some period of time. MPEDE is a multi-population differential evolution approach which combines multiple strategies into one DE variant.

A number of recent stochastic optimization methods are chosen to be compared with CLAMS on CEC2013. A brief description of these methods is given in Table 4.

Table 4. Compared stochastic optimization approaches for CEC2013

Algorithm	A brief description
NGDE/rand/1 [44]	In the neighborhood guided DE (NGDE), a neighborhood guided selection (NGS) is introduced to guide the mutation process by extracting the promising search directions.
COOA [43]	A novel creativity oriented optimization model (COOM) and algorithm (COOA) inspired by the creative thinking process.
IVbBoPSO [42]	Improved Velocity Bounded Boolean Particle Swarm Optimization (IVbBoPSO) that introduces a velocity min parameter
CoFFWA [41]	A cooperative framework for fireworks algorithm (CoFFWA), which can greatly enhance the exploitation ability of non-core fireworks by using independent selection operator. It also increases the exploration capacity by crowdness-avoiding cooperative strategy among the fireworks.
MEABC [40]	A novel multi-strategy ensemble artificial bee colony (MEABC) algorithm in which a pool of distinct solution search strategies coexists throughout the search process and competes to produce offspring.
ABC-SPSO [45]	A hybrid PSO and ABC, the PSO algorithm is augmented with an ABC method to improve the personal bests of the particles.
f_k-PSO [46]	A dynamic Heterogeneous particle swarm optimizers which allow particles to use different update equations, referred to as behaviors. f_k-PSO learns the favorability of each behavior to select the next behavior of a particle.

5.2. CLAMS parameter settings

Like other stochastic optimization methods, CLAMS has some parameters which require appropriate tuning. These parameters are listed in Table 5.

Table 5. Parameter settings for CLAMS

Parameter	description	value
M	Number of swarms (Number of cells in the CLA)	14
N	Number of particles in each swarm (Number of learning automata in each cell)	3
$\lambda + 1$	Number of actions in a learning automaton	5
χ	Acceleration constant	3
ξ	Learning period	7
W	inertia weight	$w \in [0.4, 0.9]$
a	reward parameter for LA	0.2
b	Penalty parameter for LA	0.02

Some of these parameters can be set intuitively, while some others can be chosen from the related literature. However, CLAMS has some specific parameters like ξ and χ which require suitable adjustment. This section investigates the sensitivity of the proposed method to the learning period. Other parameters can be analyzed in the same way; however, for the sake of space, we briefly discuss their settings in this section.

Like some other developed PSO methods, the inertia weight will be defined as a decreasing function of "the number of objective function calls", and its value will be changed linearly from 0.9 to 0.4 [36]. The $L_{R\text{-}\varepsilon P}$ learning scheme (b<<a) with a=0.2 and b=0.02 will be used in CLAMS as suggested in [21]. However, CLAMS is not much sensitive to the parameters a and b as long as the learning scheme is guaranteed to be $L_{R\text{-}\varepsilon P}$. $M\times N$ defines the number of particles in the population. We will use a population size near 40, which is suggested in some literature for 30-*D* and 50-*D* problems. In order to make the use of multi swarm approach sensible, we intend to pick the number of swarms as large as possible. To this end, the settings N=3 and M=14 will be used in the rest of the paper. The maximum connectivity degree can be in the range [2,..,M+2]; accordingly, the action set size of each LA is at most M, i.e. $\lambda+1\leq M$. Also, it is reasonable to assume $\lambda+1\geq 2$ since all swarms evolve in isolation when the maximum connectivity degree is set to 2. A large connectivity degree would be unsuitable as it makes a particle to oscillate between too many locations. Additionally, the behavior of a particle is almost similar for relatively large connectivity degrees. Accordingly, a maximum connectivity degree of 5 or 6 seems a reasonable choice, i.e. $\lambda+1=4$ or $\lambda+1=5$. Learning period can be determined in the same way as χ, We have tested different values of the learning period in the range [1,15], and the values in the range [5,10] exhibited better performance. As a result, their midpoint, i.e. 7, is selected in the experiments.

In order to study the sensitivity of CLAMS to the acceleration constant, it is examined on a set of 30-D benchmarks from Table 2. The experiments are conducted using different values of χ in the range [2.4, 3.8], and the experimental results are obtained over 30 independent runs on each benchmark. Each run of the algorithm is terminated after 1E+5 fitness evaluations. We use the Friedman test to compare the effect of different parameter settings on the performance of CLAMS. The Friedman ranks are obtained according to the achieved average fitness values, and are given in Figures 5-8; also, the average ranks of the algorithm under different settings are provided in Figures 9. In the results, a smaller rank indicates a better performance.

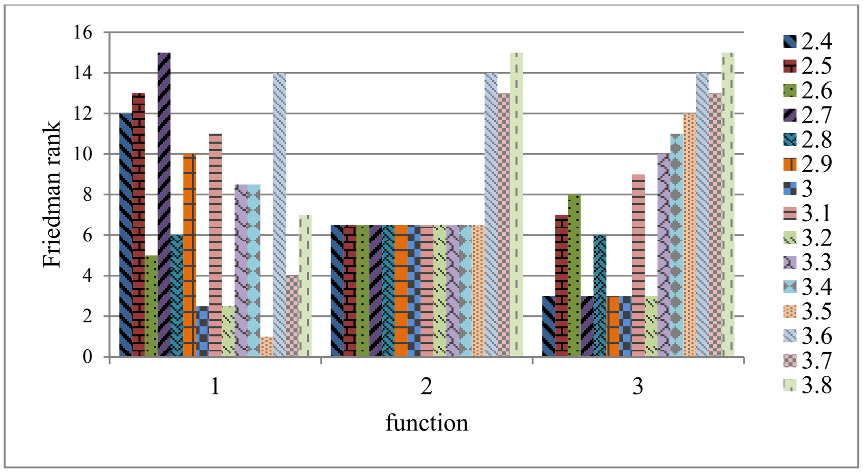


Figure 5. Freidman ranks of CLAMS under different settings of χ.

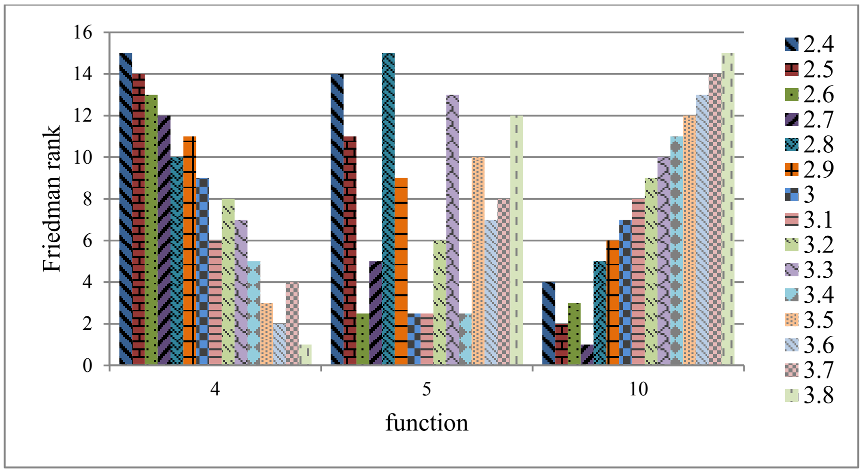


Figure 6. Freidman ranks of CLAMS under different settings of χ.

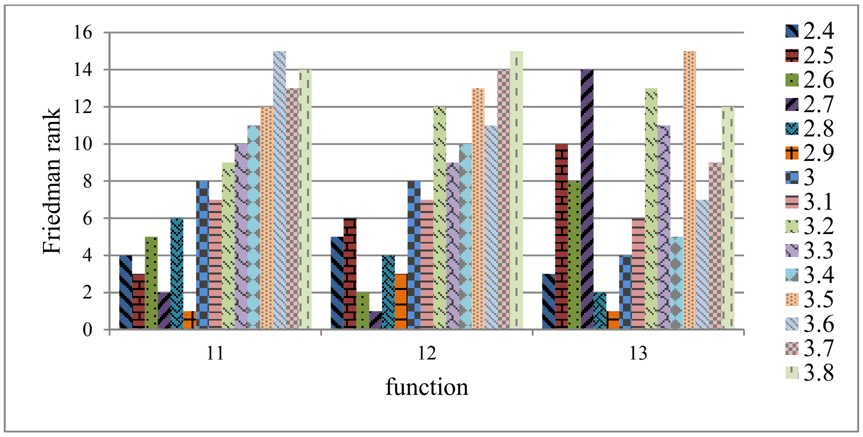


Figure 7. Freidman ranks of CLAMS under different settings of χ.

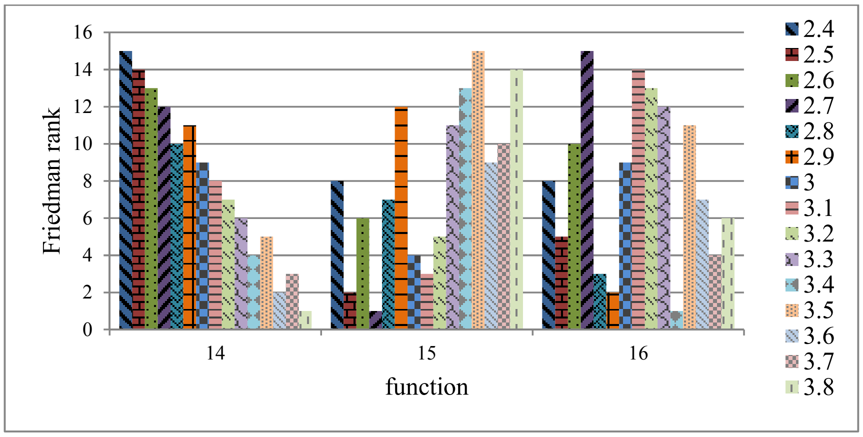


Figure 8. Freidman ranks of CLAMS under different settings of χ.

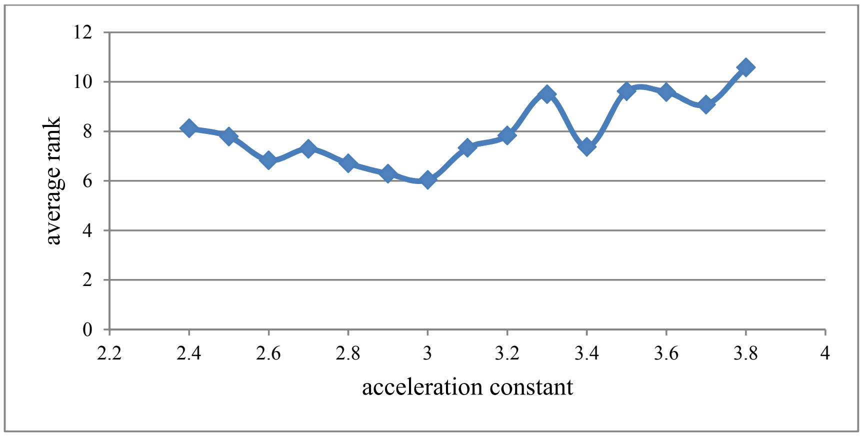


Figure 9. The average Freidman ranks of CLAMS under different settings of χ.

It is generally assumed that a large acceleration constant facilitates a global search while a small acceleration constant facilitates a local search. Although there is a general insight on how the acceleration constant affects the performance of the

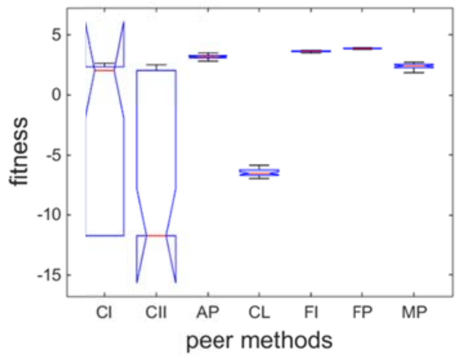
PSO approaches, from the obtained results in Figures 5-8 it is clear that there is no single value of the parameter efficient for all benchmarks. The effect of acceleration constant on some problems is clearly observable in the Figures. For instance, the performance of the algorithm on f_{10} and f_{11} improves as χ decreases. These problems possess the properties of unimodal functions and require an efficient local search. Problems like f_2 and f_3 are multimodal with many local optima. The proposed method is able to find their global optimum using a wide range of tested values of χ; however, its convergence speed and hence the resolution of its final results increases when χ decreases. Analogous discussions can be made concerning problems f_{12} and f_{15}. On some other problems like f_1, the algorithm achieved better results for relatively large tested values of χ. In summary, the results of the proposed method on the first set of benchmarks are barely distinguishable for the various values of χ in the tested range, especially when $\chi<3.5$ (see the comprehensive statistical results provided in appendix A). For the rest of the paper, the configuration $\chi=3$ is adopted as it achieves the best average rank on the tested problems (Figure 9).

5.3. Experimental results on the classical problems

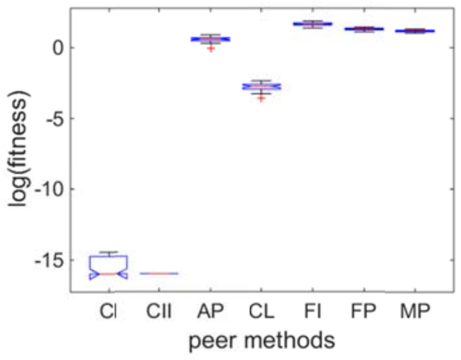
In this section, we analyze the performance of the proposed method in comparison to the other peer approaches on the benchmark set given in Table 2. The results of the compared methods on each benchmark are obtained over 30 independent runs, and each run is terminated after 1E5 fitness evaluations on the 30D instances of the problems and after 3E5 on their 50D cases.

5.3.1. Experiments on 30 dimensional problems

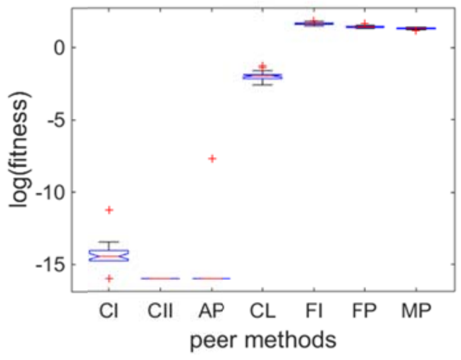
Box plots of the obtained results are given in Figure 10; additionally, the Friedman ranks are provided in Figures 11-13 (other statistical results are given in Appendix B). Box plot provides an appropriate tool to visualize the distribution of the obtained results. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually.



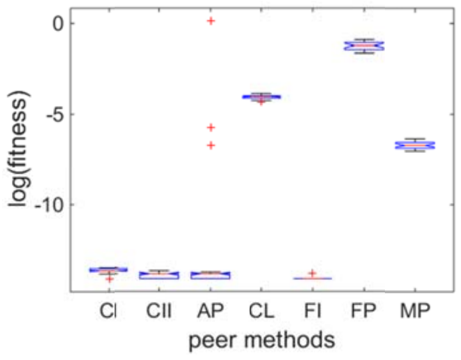
f_1



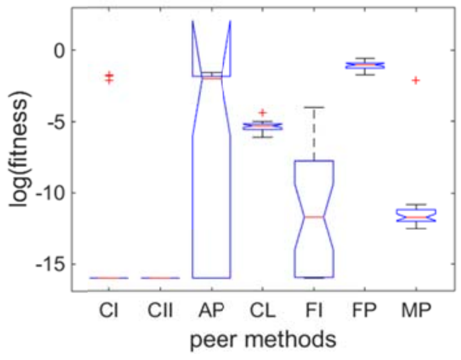
f_2



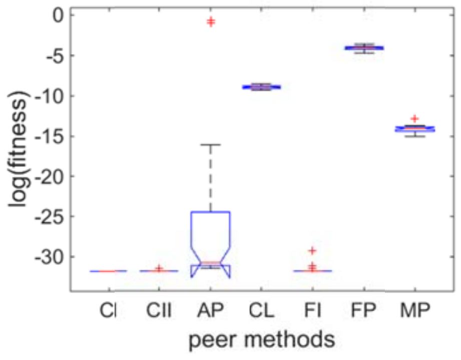
f_3



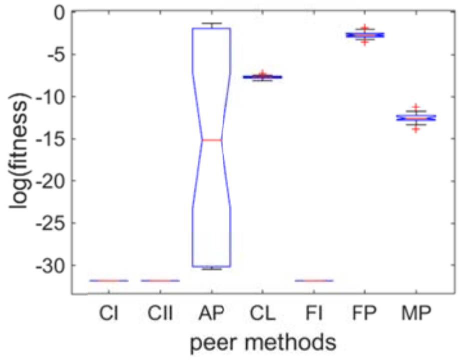
f_4



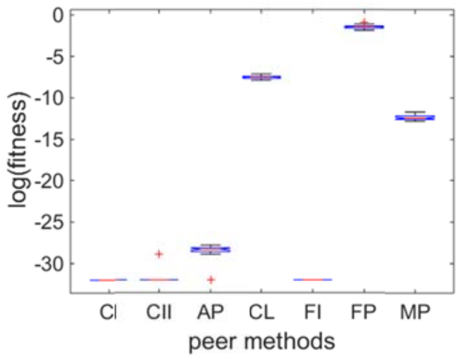
f_5



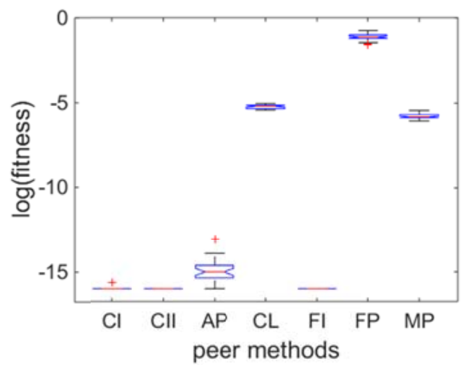
f_6



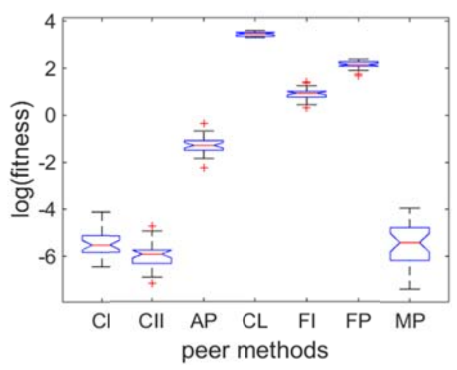
f_7



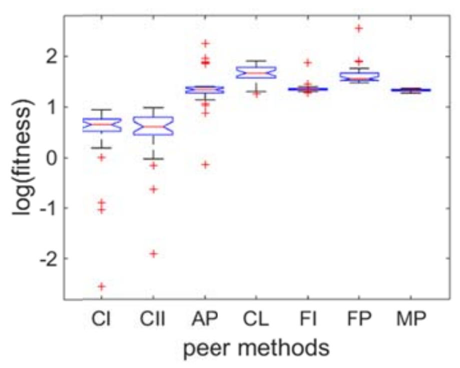
f_8



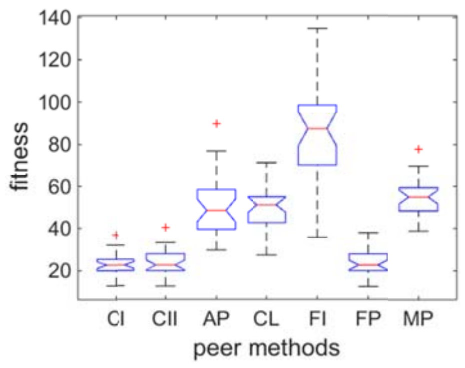
f_9



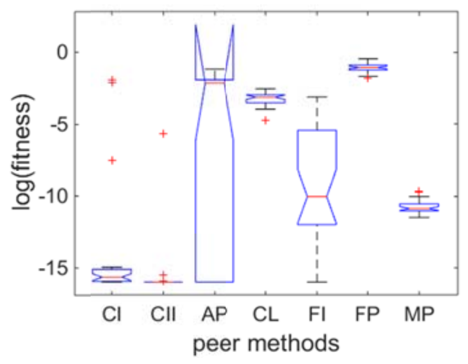
f_{10}



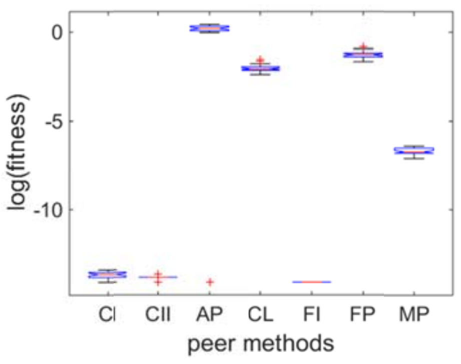
f_{11}



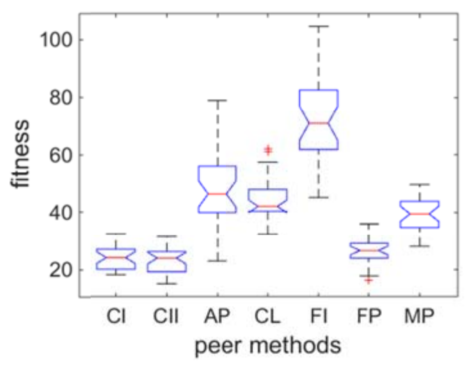
f_{12}



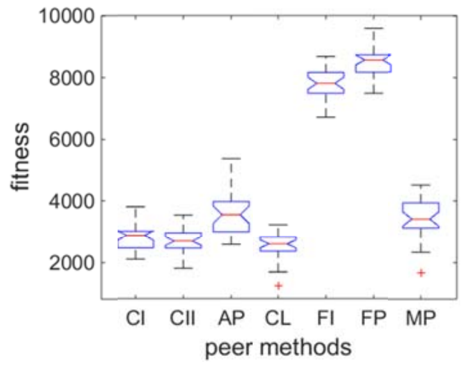
f_{13}



f_{14}



f_{15}



f_{16}

Figure 10. The boxplots for the compared methods on the classical problems f_1-f_{16}. CI = CLAMS-I, CII = CLAMS-II, AP = APSO, CL = CLPSO, FI = FIPS, FP = FPSO, MP = MPDE.

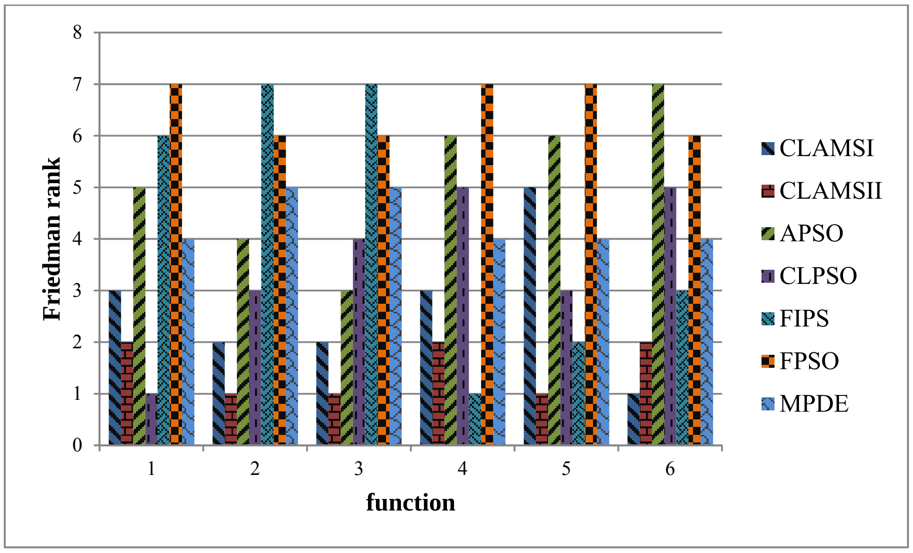


Figure 11. Freidman ranks of the compared methods on the classical benchmarks f_1-f_6.

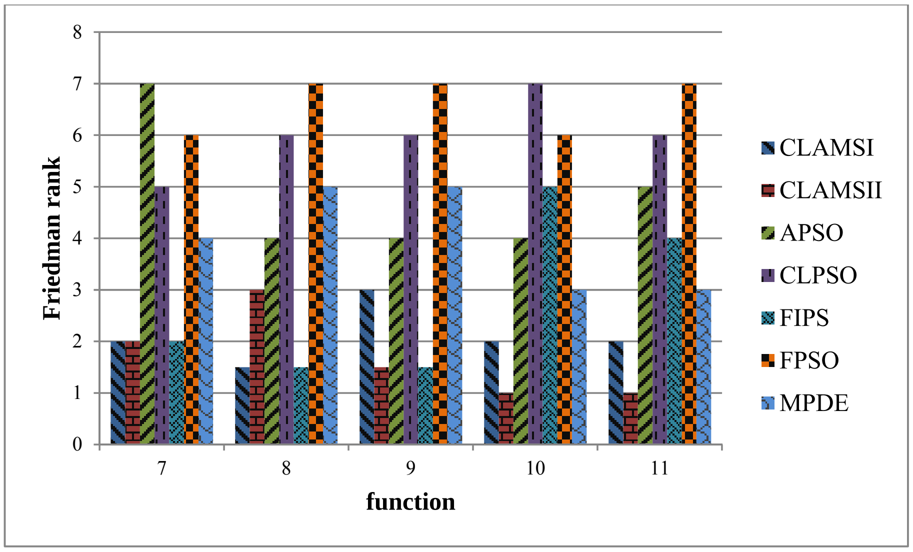


Figure 12. Freidman ranks of the compared methods on the classical benchmarks f_7-f_{11}.

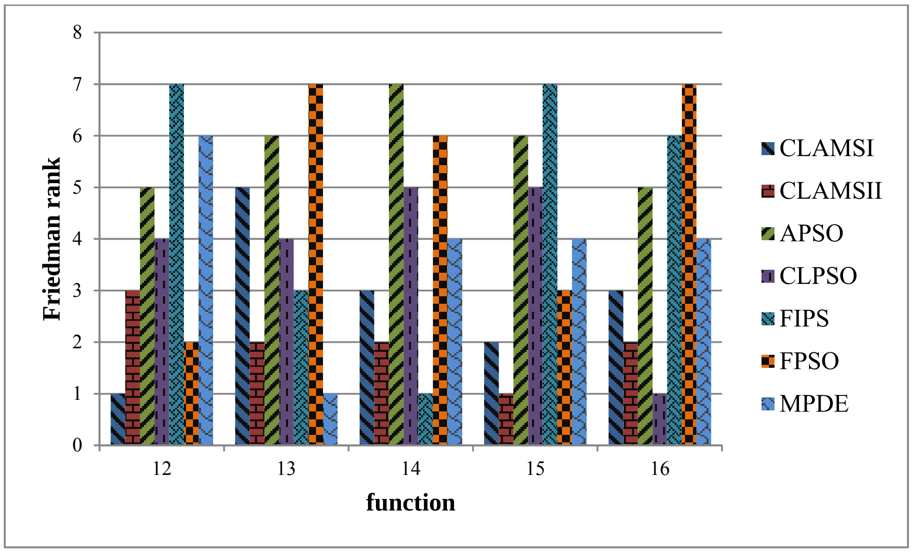


Figure 13. Freidman ranks of the compared methods on the classical benchmarks f_{12}-f_{16}.

As it can be observed from Figure 10, the two developed methods have a very close performance. However, CLAMS-II possesses relatively better exploration abilities when it is tested on f_1, f_5, and f_{13}; on these problems, CLAMS-I shows more liability to entrapment in local optima than CLAMS-II. Additionally, CLAMS-II has a bit higher accuracy results on most problems, which may suggest its better exploitative behavior. These observations may justify the velocity definition of CLAMS-II and the discussions over its definition presented in section 4.3.1.

When CLAMS-II is compared to the other peer approaches, its higher accuracy is quite clear. On almost all of the tested problems, both variants of CLAMS obtained the best results. CLPSO has the best average rank on f_1 whereas CLAMS-II is prone to entrapment in local optima. f_1 has its optimum point near the boundaries of its landscape while its fittest local optima lay on the other corners. Accordingly, when a particle is trapped in a relatively high fitness local optimum, it should traverse a long path to reach the global one. In CLAMS, a particle may be attracted by several other particles at the same time. If these particles reside on the local optima at different sides of the landscape, the particle may be attracted toward the interior areas of the search space; As a result, it may lose track of the eligible regions at the search space boundaries. This issue resulted in the weaker average rank of the proposed method on f_1. CLAMS utilizes the second best mutation to deal with this kind of issues; hence, as it can observed from Figure 10, CLAMS-II achieved the most accurate results on f_1 in most of its performed searches.

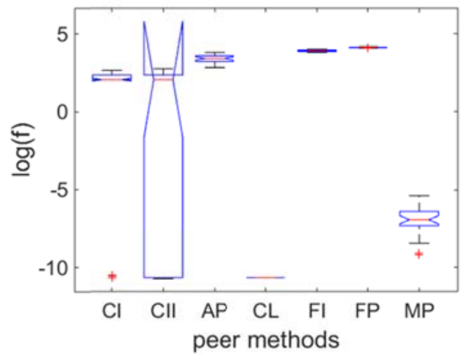
Considering the different compared methods, CLAMS-II obtained the most accurate results or very comparable ones on the multi modal problems f_2-f_7. The performances of APSO and FIPS are very close on some of these functions to CLAMS-II; however, none of these methods outperform CLAMS-II. CLPSO was also able to find the global optimum of these problems on all of its performed runs; though, its low convergence speed resulted in less accurate final results.

Considering the two unimodal problems f_8 and f_9, almost all of the compared methods obtained appropriate results. However, the proposed methods and MPEDE are superior to the other compared algorithms on the unimodal benchmark f_{10}. CLAMS-II also obtained satisfactory results on the multi modal problems f_{11}-f_{16}. It achieved the most accurate results in almost all of its performed searches on the problems f_{11}, f_{13}, and f_{15}, and obtained the best average ranks on f_{11} and f_{15}. Considering problems f_{12} and f_{15}, the introduced methods and FPSO all acquired very competitive results. The decreasing connectivity of FPSO would be the result of its appropriate performance on these two problems. The introduced methods also achieve appropriate results on these benchmarks as a result of their topology tuning mechanism. However, CLAMS achieves this issue adaptively during its search process. Consequently, its population topology changes according to the properties of the fitness landscape, which aids it to outperform FPSO on the other tested benchmarks.

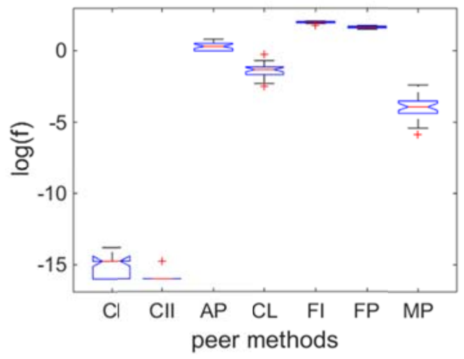
4.3.2. Experimental results on 50 Dimensional problems

This section repeats the experiments of the previous section in 50 dimensions. The obtained results are given in Figures 14-17. Except for MPDE, the obtained results are analogous to those observed in the previous section. MPEDE achieved a satisfactory performance in this section owing to the increased number of fitness evaluations. It exhibits an appropriate exploration behavior, which leads to eligible results after a relatively long period of search process. However, the designated maximum number of fitness evaluations in the previous section seems to be inadequate for its fulfillment. Considering Figures 15-17, the two proposed methods obtained the best ranks on 9 benchmarks and achieved very competitive results on the reminder. The satisfactory performance of the introduced method on problems with different characteristics demonstrates its effectiveness in maintaining a suitable balance between its explorative and exploitative behaviors, which is accomplished through the developed topology adaption mechanism.

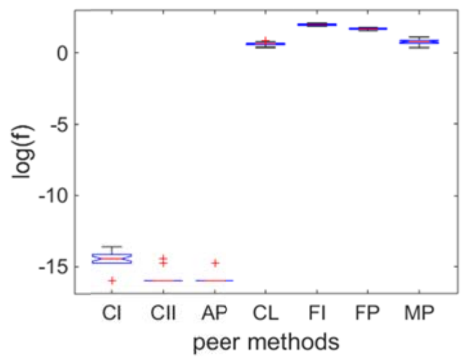
We argue that this can be accomplished most effectively by carefully balancing exploration and exploitation



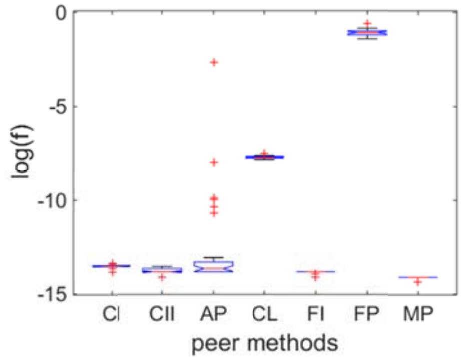
f_1



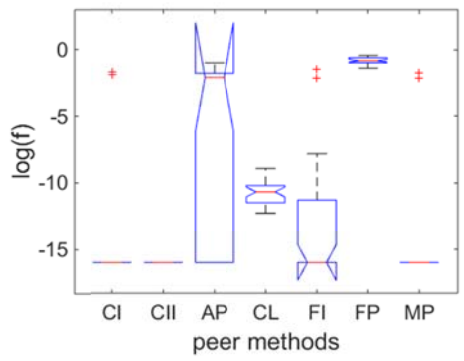
f_2



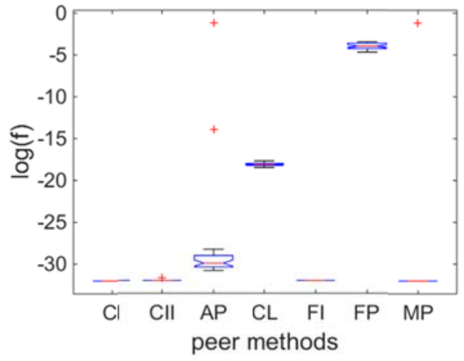
f_3



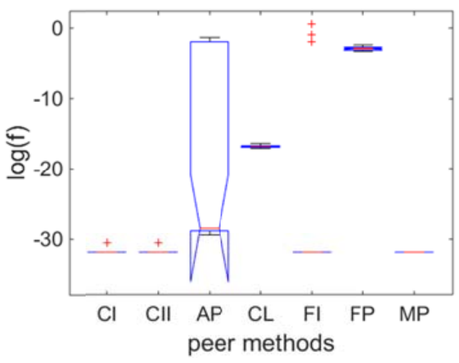
f_4



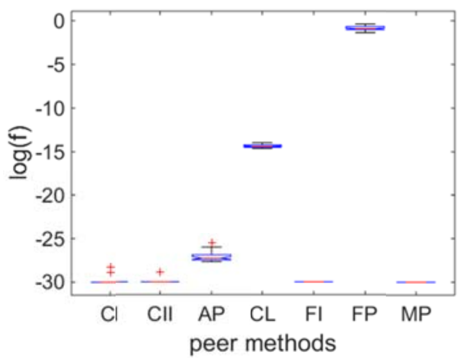
f_5



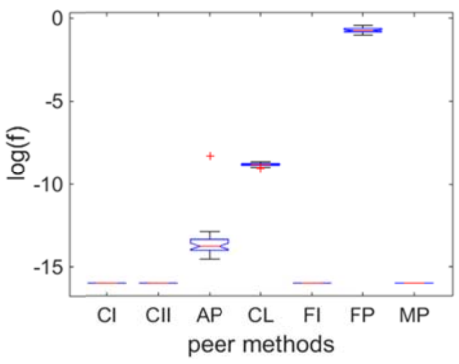
f_6



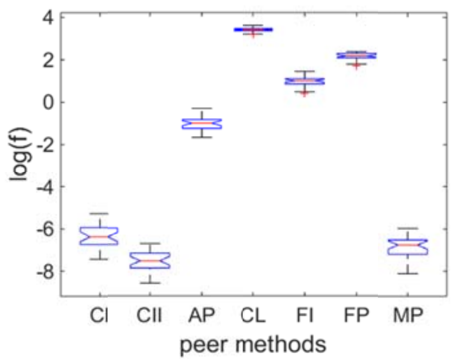
f_7



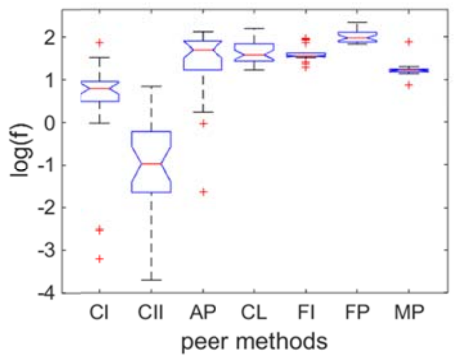
f_8



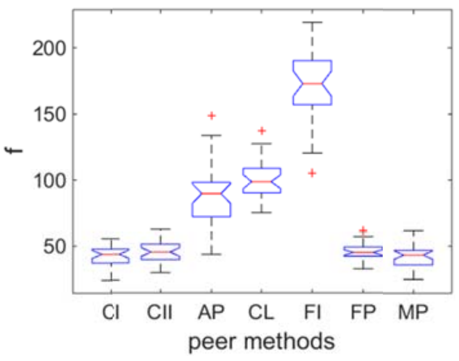
f_9



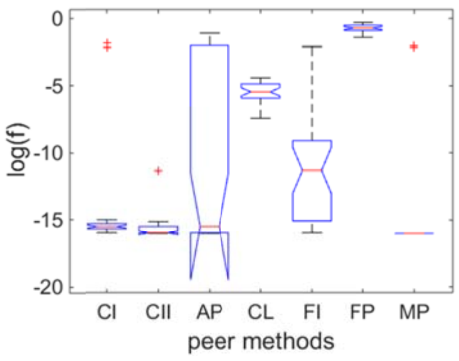
f_{10}



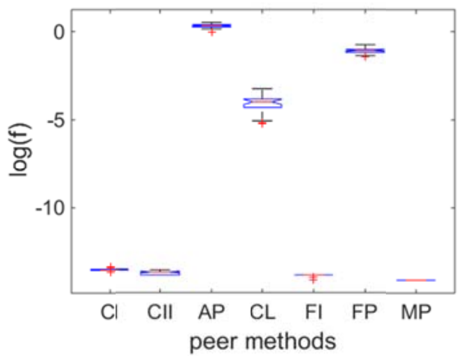
f_{11}



f_{12}



f_{13}



f_{14}

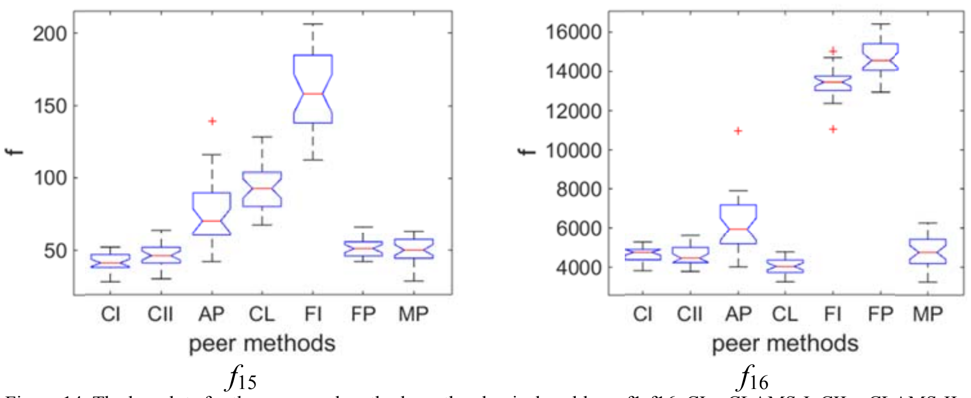


Figure 14. The boxplots for the compared methods on the classical problems f1-f16. CI = CLAMS-I, CII = CLAMS-II, AP = APSO, CL = CLPSO, FI = FIPS, FP = FPSO, MP = MPDE.

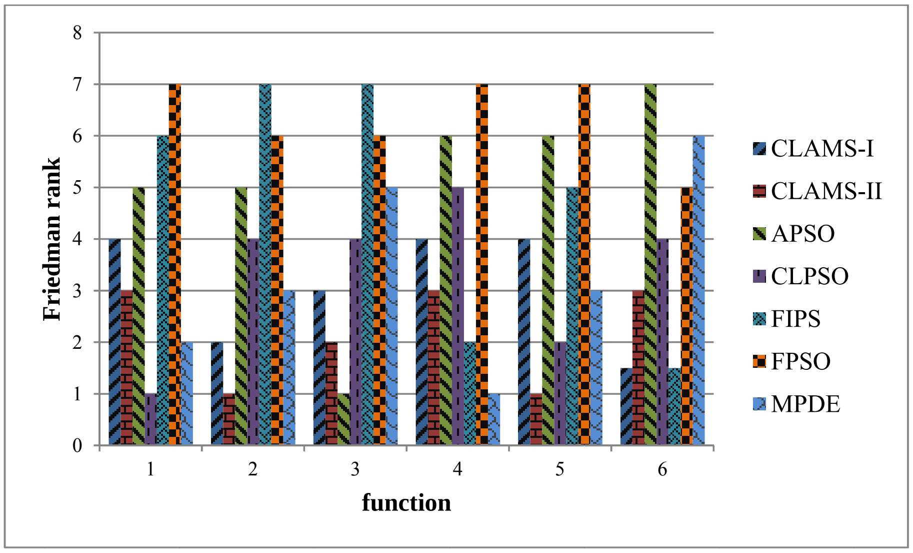


Figure 15. Freidman ranks of the compared methods on the classical benchmarks f_1-f_6.

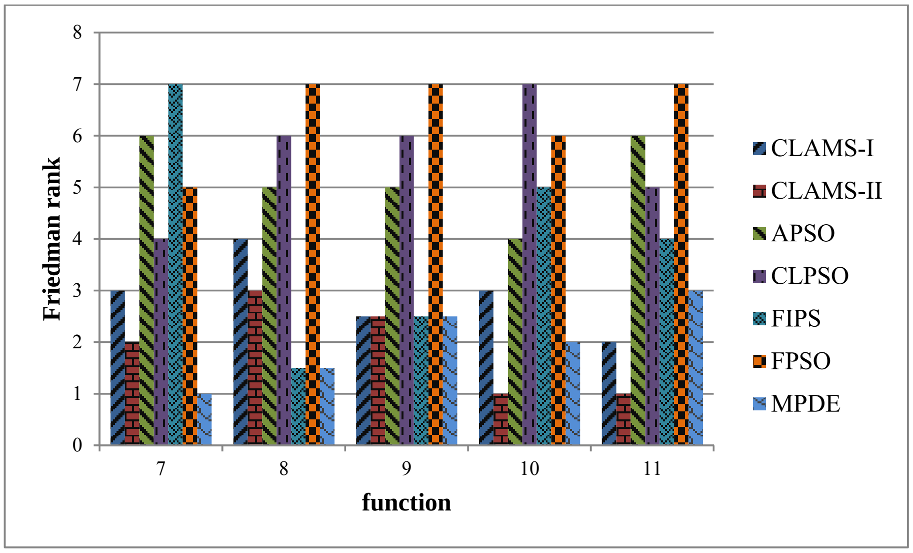


Figure 16. Freidman ranks of the compared methods on the classical benchmarks f_7-f_8.

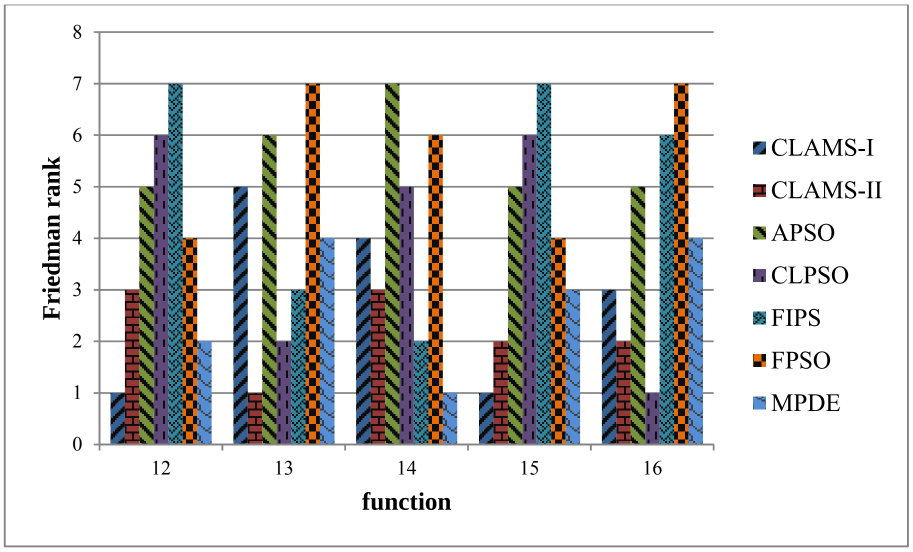


Figure 17. Freidman ranks of the compared methods on the classical benchmarks f_{12}-f_{16}.

5.4. Experimental results on the CEC2013 benchmark set

In this section, we analyze the performance of the proposed approach on the CEC2013 benchmark set. As suggested in [35], each run of an algorithm on a benchmark is repeated 51 times and its result is recorded after a maximum number of 1E4D fitness evaluations. Figures 18-21 depict the Friedman ranks of the compared methods on each of the tested problems. Also, the average Freidman rank of each method provided in Figure 22. Other statistical results suggested in [35] are provided in the Appendix D.

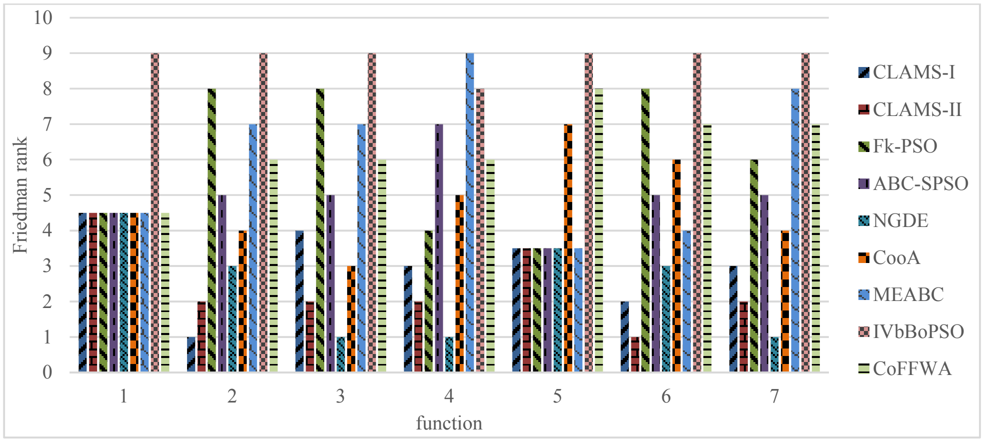


Figure 18. Freidman ranks of the compared methods on the CEC2013 benchmark set, for benchmarks f_1-f_7.

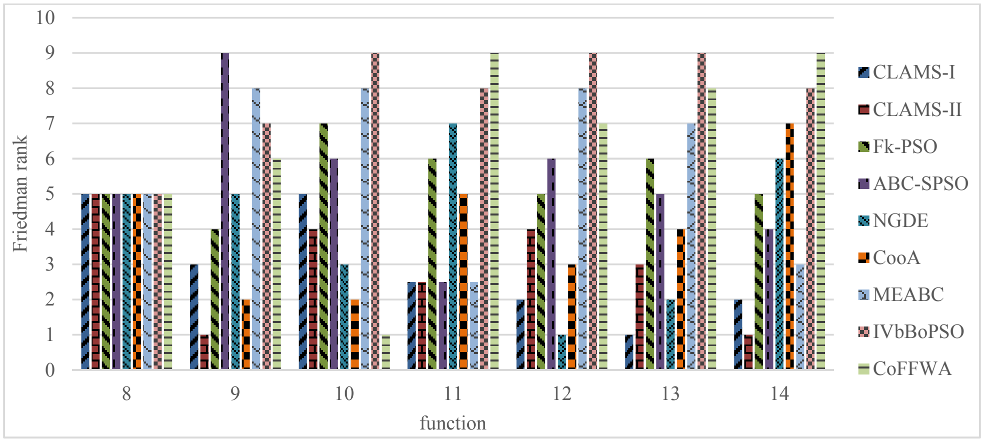


Figure 19. Freidman ranks of the compared methods on the CEC2013 benchmark set, for benchmarks f_8-f_{14}.

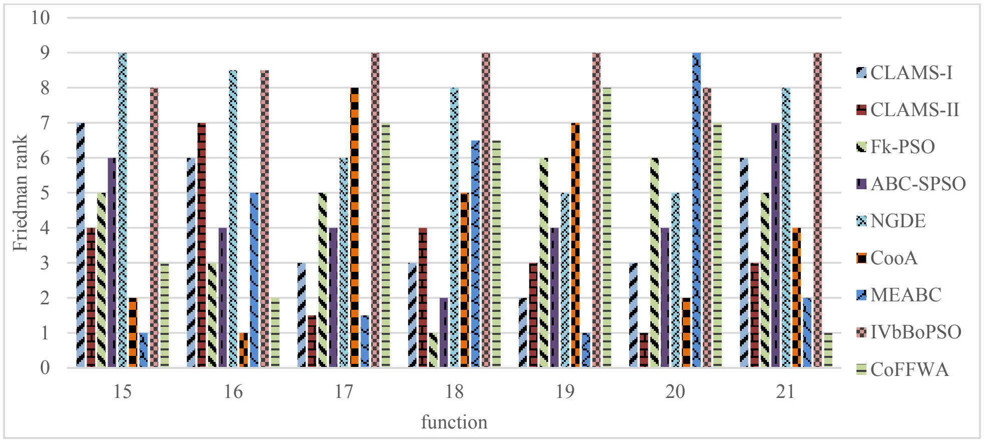


Figure 20. Freidman ranks of the compared methods on the CEC2013 benchmark set, for benchmarks f_{15}-f_{21}

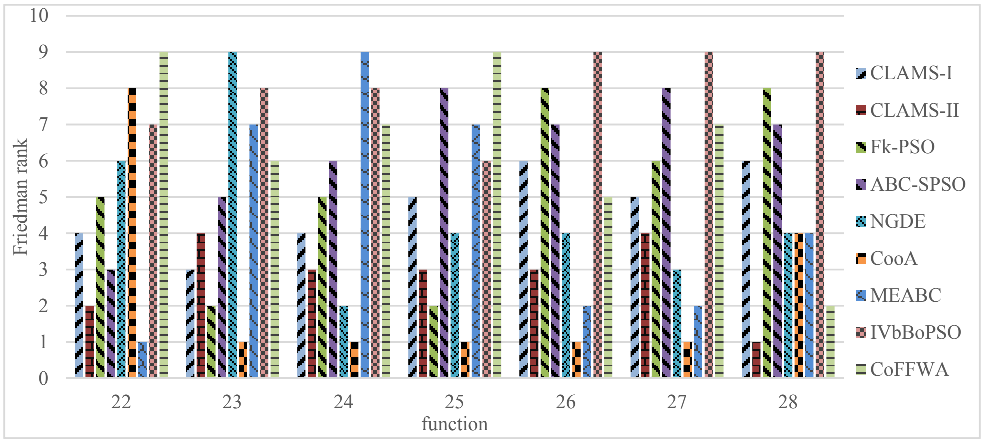


Figure 21. Freidman ranks of the compared methods on the CEC2013 benchmark set, for benchmarks f_{15}-f_{21}

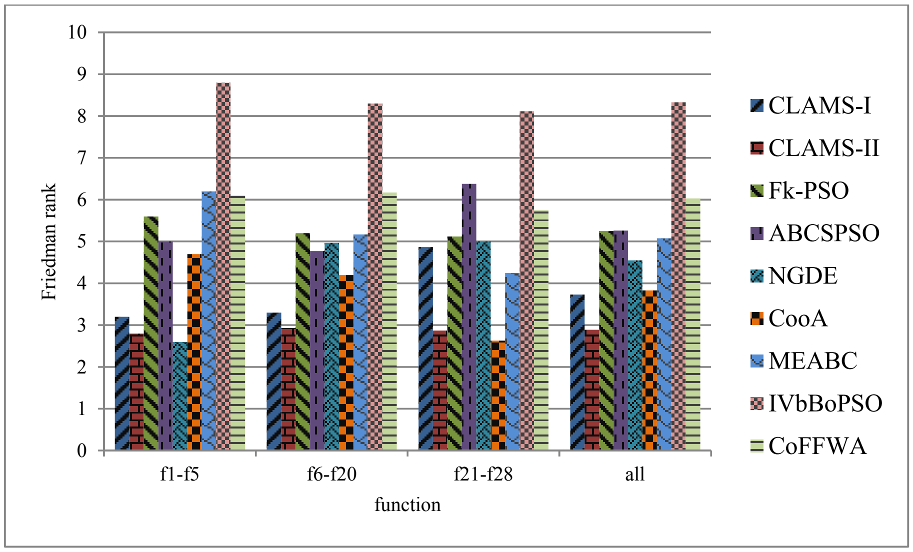


Figure 22. Average Freidman ranks of the compared methods on the CEC2013 benchmark set.

Considering the obtained results, none of the compared methods are superior to the others on all of the tested problems. But, the proposed methods are very competitive with the best performing ones on most of the benchmarks. Also, considering the average Freidman ranks in Figure 22, CLAMS-II has the best performance, and CLAMS-I comes in the second place. According to the average ranks, we can sort the algorithms in the following order: CLAMS-II, CLAMS-I, CooA, NGDE/rand/1, MEABC, F_k-PSO, ABC-SPSO, CoFFWA, and IVbBoPSO. On the unimodal benchmarks f_1-f_5, NGDE has the best performance and obtained the smallest average errors on 4 problems. On these benchmarks, the two CLAMS approaches are very competitive with NGDE. CLAMS-II obtained the best average results on 2 problems and the second best averages on the reminder. Also, CLAMS-I possesses the best average on 3 problems of this group. The introduced

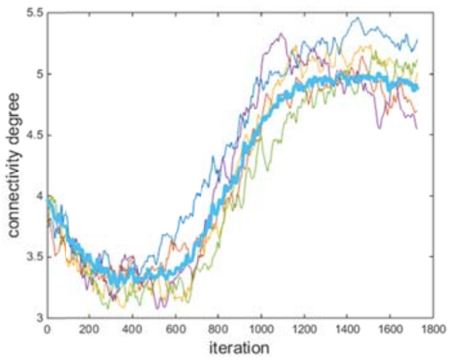
CLAMS-II is the best performing method on the basic multimodal problems, f_6-f_{20}, while CLAMS-I comes in the second place. CLAMS-II has the smallest rank on 7 problems of this category, while CLAMS-I achieved the smallest ranks in 3 cases. CooA obtained the third average rank on this group of problems. It obtained the lowest average errors on the composition problems, where it possesses the best ranks in 5 cases. However, the results of CLAMS-II are very close, and it achieved the second overall ranking on the composition problems.

5.5. A study on the learning capabilities of CLAMS

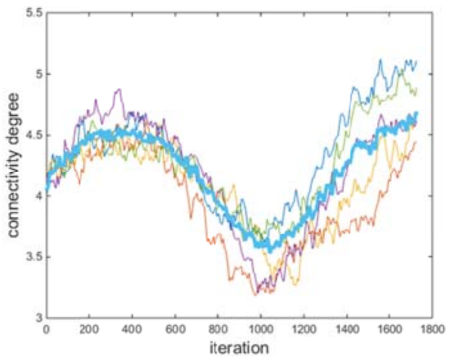
In this section, we study the learning capabilities of the proposed approach. CLAMS is based on several mechanisms that together control its dynamics. In this algorithm, each particle uses multiple topologies and, related to them, multiple updating rules. The algorithm also utilizes a multi-swarm structure and two mechanisms to enhance its exploration and exploitation abilities. Finally, a CLA is incorporated into the algorithm, which enables the particles to select their topologies in an intelligent manner. Accordingly, various factors influence the search behavior of the introduced approach. In order to investigate the learning properties of CLAMS and its convergence behavior, we have developed three specific experiments.

Experiment 1: topology adaption process

The experiments in this section are employed to illustrate the topology adaption process. In this regard, CLAMS-II is examined on set of benchmarks from Table 2, and the average connectivity degree of the population is recorded during the search process. Figure 23 shows the obtained results only on 6 selected benchmarks since similar observations can be achieved on the other functions.



f_1



f_2

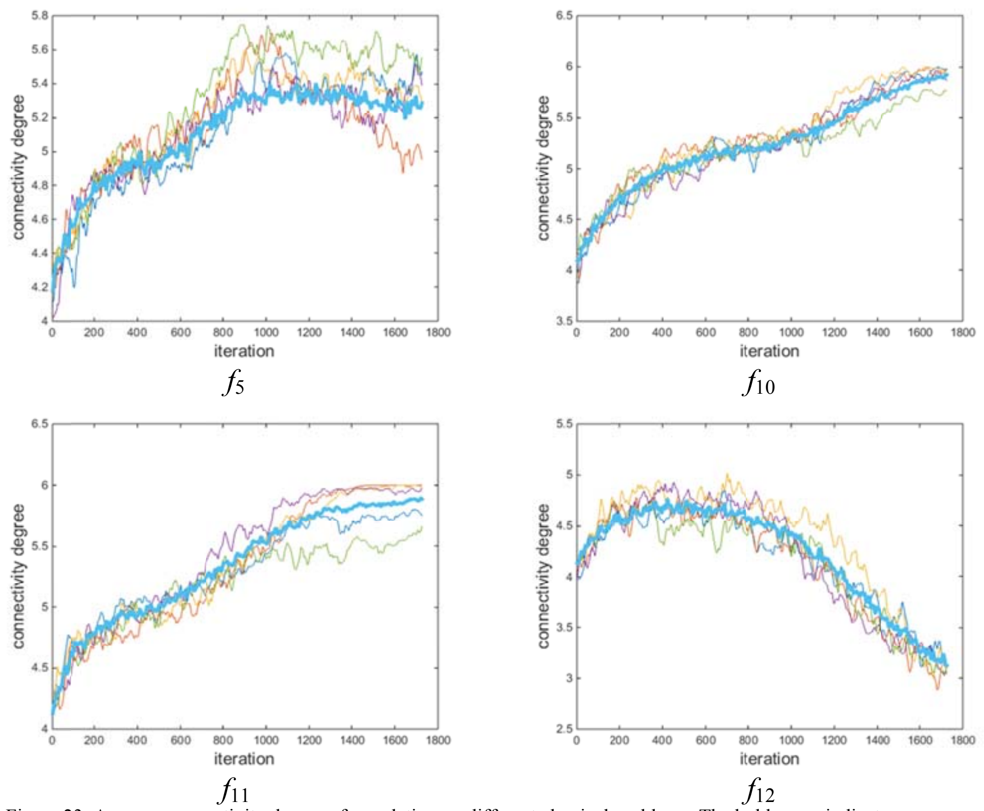


Figure 23. Average connectivity degree of population on different classical problems. The bold curve indicates average connectivity degree over 30 independent runs on each problem. The average connectivity degree in sample runs on each benchmark is depicted in lighter curves.

Although it is not exactly clear that how each connectivity degree affects different stages of the search process on different problems, we can use the properties of the tested problems to interpret the observed results. One interesting observation about the obtained results is that the connectivity degree of the particles changes in a similar pattern in all runs of the algorithm on a specific benchmark. This observation leads to the speculation that the proposed method is capable of adjusting its population topology according to the properties of the fitness landscape.

Particles utilize similar information sources in their update rules when they are using large connectivity degrees. Hence, a large connectivity degree can be beneficial for convergence after detection of a global optimum. Consequently, on all of the tested benchmarks except f_{12}, CLAMS tends to use large connectivity degrees at the final stages of its search. The connectivity degree of particles deceases immediately at the beginning of the search when CLAMS is applied to f_1. On this benchmark, particles employ small connectivity degrees in the early stages of their search to avoid the low fitness interior areas of the landscape. In contrast, the fittest local optima are around the global optimum in f_2, and the quality of these local optima decreases with respect to their distances from the global optimum. Accordingly, CLAMS increases the connectivity degree of its particles to draw them to the areas around the global optimum and then decreases their connectivity degrees so that they can exploit their local area. The algorithm is capable of finding the global optimum of f_5 in a short period of time using different connectivity degrees and utilizes large connectivity degrees for fast convergence. f_{10} and f_{11} can be solved using an appropriate local search. Although f_{11} is considered to be multi modal, its characteristics are much similar to unimodal problems. The algorithm uses the convergence state from the beginning of its search on unimodal problems. f_{12} is the rotated version of f_2, and CLAMS uses a similar connectivity adaption pattern on this benchmark; however, the algorithm is unable to reach the convergence state as it is unable to find the global optimum.

Experiment 2: convergence analysis

In order to investigate the convergence behavior of CLA in the proposed method, we use the concept of entropy, which was introduced in the context of information theory by Shannon [47]. If we consider the selected action of a learning automaton like LA_i^j during time t as a random variable, we can define its entropy using the following equation:

$$H_i^j(t) = -\sum_{k=1}^{\lambda+1} q_i^j(k,t).\ln\left(q_i^j(k,t)\right) \quad (11)$$

Here $q_i^j(k,t)$ represents the selection probability of the k^{th} action of LA_i^j at time instant t. $H_i^j(t)$ can be considered as a measure of uncertainty associated with LA_i^j at time t; larger values of $H_i^j(t)$ represent more uncertainty in the decisions of LA_i^j. The defined measure only considers a single learning automaton. We can extend this definition to obtain a measure of uncertainty for CLAMS:

$$H(t) = \sum_{j=1}^{M} \sum\nolimits_{i=1}^{N} H_i^j(t) \quad (12)$$

where M is the number of cells in CLAMS, and N is the number of learning automata within each cell. Figure 24 shows the entropy of CLAMS on different tested problems.

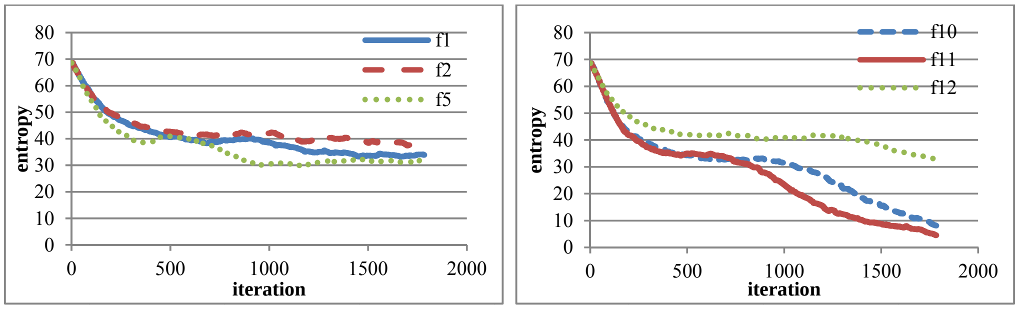


Figure 24. Entropy of CLAMS during the search process

Considering Figure 24, it can be observed that the entropy of CLAMS is high at the beginning of the algorithm, but decreases during the search procedure. It ends up below 10 on f_{10} and f_{11}, indicating that the entropy of each learning automaton is on average below 0.24. Considering a 5 action learning automaton, its entropy is below 0.24 when the selection probability of one action is above 0.93. On the other problems the entropy is near 35 at the final stages of the algorithm. Accordingly, the average entropy of each LA is about 0.83. Figure 25 shows different sample probability vectors that have entropies near 0.83. Although we cannot confirm the existence of a single high probability action using the entropy criteria in this situation, but we can confirm that the selection probabilities of at least three actions are very low.

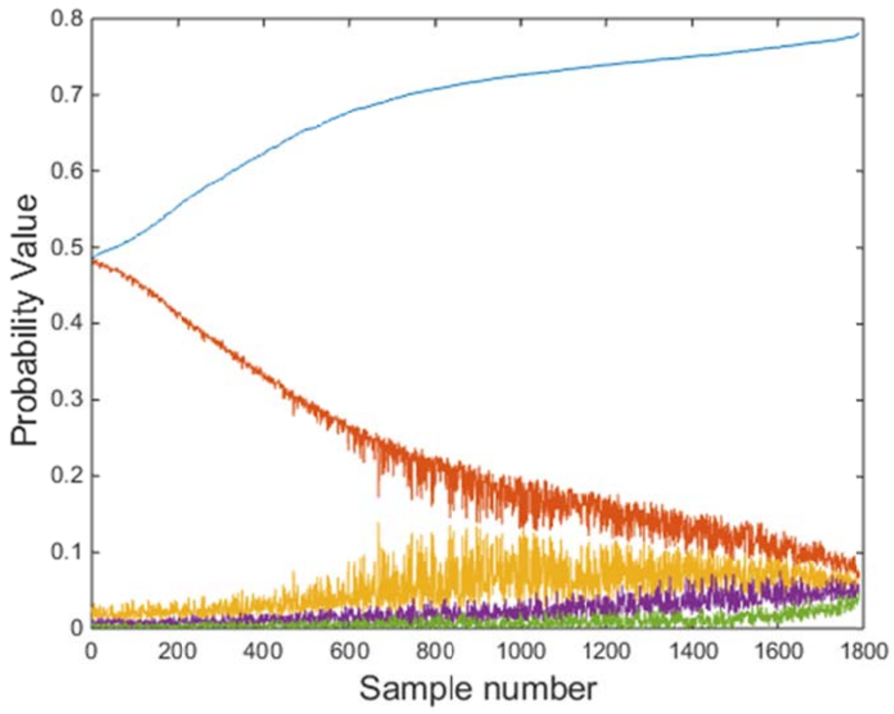


Figure 25. A set of sample probability distributions with entropy near 0.83

Experiment 3: effectiveness of learning

In order to exhibit the impact of the proposed learning scheme on the performance of CLAMS, we can statistically compare CLAMS with its pure chance version. In the pure chance version, the learning automata are disabled; consequently, each particle chooses its topology in a completely random manner. CEC2013 benchmark functions are employed in this section, and Wilcoxon rank sum test for a confidence level of 95% [48] is performed on each benchmark. Wilcoxon test is a nonparametric alternative to the paired t-test [48] and is based on the null hypothesis that the two compared algorithms are statistically equivalent. The results of the test are given in Table 6, where '+' denotes the statistical superiority of CLAMS to its pure chance version, '-' denotes the opposite case, and '=' is used if the two versions of CLAMS are statistically indistinguishable.

From the obtained results, it is clear that the introduced CLAMS-II is statistically different from its pure chance version. Also, it can be seen that CLAMS-II is statistically superior to the pure chance CALMS on most of the tested benchmarks. There are only 8 tested problems that the proposed learning method is statistically indistinguishable from a pure chance scheme; additionally, considering Figures 18-21, almost every ordinary optimization method performs well on some of these 8 problems. These observations indicate the effectiveness of the proposed learning method on different optimization problems.

Table 6. Wilcoxon rank sum test results on CLAMS-II and its pure chance variant. The p-values of the tests are provided for each benchmark in CEC2013.

1	2	3	4	5	6	7
(=)NAN	(+)7E-12	(=)4E-01	(+)4E-17	(=)NAN	(+)5E-16	(+)2E-07
8	9	10	11	12	13	14
(=)9E-01	(+)3E-02	(=)8E-01	(=)NAN	(+)3E-14	(+)8E-11	(+)4E-06
15	16	17	18	19	20	21
(+)4E-12	(=)2E-01	(+)4E-18	(+)4E-17	(+)7E-15	(+)8E-10	(+)5E-04
22	23	24	25	26	27	28
(+)5E-10	(+)2E-10	(=)7E-01	(+)5E-02	(+)5E-17	(+)4E-04	(+)3E-03

6. Conclusion

Topological properties have an intense impact on the search behavior of the population based stochastic optimization methods. Accordingly, adaptive tuning of these properties is a promising research trend in PSO, which has been considered in a relatively few studies of the field. This paper investigated a CLA based multi swarm algorithm in which a CLA adaptively controls the topology connectivity of the particles. The effectiveness of the proposed approach was exhibited in two phases. First, it was shown that the proposed algorithm using an adaptive topology surpasses its variant which uses a random topology. Then, the effectiveness of using adaptive topology was demonstrated by comparing the proposed approach with other population based optimization methods. The introduced method was compared with 12 other recently developed stochastic optimization methods on two sets of benchmark problems. The comparison results demonstrated the effectiveness and superiority of the proposed approach.

Appendix A.

Statistical results of CLAMS-II under different settings of χ.

Table 7. Statistical results (average, median, min, max) of CLAMS-II on 30-D classical problems under different settings of χ.

χ f	f_1	f_2	f_3	f_4	f_5	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
2.4	9.86E+01	0	0	2.23E-14	8.21E-04	5.35E-07	2.49E+00	2.32E+01	1.07E-16	2.27E-14	2.40E+01	2.69E+03
	1.18E+02	0	0	2.22E-14	0	1.57E-07	2.04E+00	2.35E+01	0	2.22E-14	2.38E+01	2.64E+03
	1.81E-12	0	0	1.50E-14	0	4.08E-09	3.74E-02	1.50E+01	0	7.99E-15	1.50E+01	2.07E+03
	3.55E+02	0	0	3.28E-14	9.85E-03	5.78E-06	8.64E+00	3.24E+01	4.44E-16	3.28E-14	4.49E+01	3.45E+03
2.5	1.06E+02	0	2.25E-15	1.84E-14	4.10E-04	3.83E-07	2.23E+00	2.34E+01	3.28E-04	2.14E-14	2.31E+01	2.65E+03
	1.18E+02	0	0	1.50E-14	0	1.23E-07	1.23E+00	2.29E+01	1.11E-16	2.22E-14	2.31E+01	2.65E+03
	1.81E-12	0	0	7.99E-15	0	3.28E-09	3.53E-03	1.04E+01	0	1.50E-14	1.53E+01	1.94E+03
	3.55E+02	0	1.59E-14	3.28E-14	1.23E-02	3.62E-06	1.01E+01	3.35E+01	9.85E-03	2.93E-14	3.00E+01	3.62E+03
2.6	7.50E+01	0	2.84E-15	1.77E-14	0	4.50E-07	2.58E+00	2.22E+01	2.46E-04	1.91E-14	2.36E+01	2.70E+03
	1.81E-12	0	0	1.50E-14	0	1.78E-07	2.31E+00	2.11E+01	1.11E-16	2.22E-14	2.36E+01	2.72E+03
	1.81E-12	0	0	7.99E-15	0	1.54E-08	1.34E-02	1.56E+01	0	7.99E-15	1.56E+01	2.02E+03
	3.55E+02	0	3.19E-14	2.93E-14	0	3.43E-06	7.62E+00	3.46E+01	7.39E-03	2.93E-14	3.55E+01	3.40E+03
2.7	1.14E+02	0	0	1.74E-14	3.70E-18	3.23E-07	2.20E+00	2.15E+01	8.21E-04	1.74E-14	2.20E+01	2.82E+03
	1.18E+02	0	0	1.50E-14	0	1.64E-07	1.58E+00	2.06E+01	0	1.50E-14	2.07E+01	2.81E+03
	1.81E-12	0	0	7.99E-15	0	1.24E-08	1.08E-02	1.40E+01	0	7.99E-15	1.32E+01	2.00E+03
	3.55E+02	0	0	2.93E-14	1.11E-16	3.09E-06	8.24E+00	3.46E+01	9.85E-03	2.93E-14	3.30E+01	3.68E+03
2.8	7.89E+01	0	1.83E-15	1.56E-14	8.21E-04	6.37E-07	3.11E+00	2.30E+01	7.40E-17	1.52E-14	2.39E+01	2.65E+03
	1.81E-12	0	0	1.50E-14	0	2.85E-07	2.49E+00	2.40E+01	0	1.50E-14	2.41E+01	2.72E+03
	1.81E-12	0	0	7.99E-15	0	2.34E-08	1.54E-04	1.13E+01	0	7.99E-15	1.53E+01	1.74E+03
	4.73E+02	0	2.84E-14	3.28E-14	9.85E-03	3.22E-06	1.42E+01	3.69E+01	4.44E-16	2.22E-14	3.21E+01	3.64E+03
2.9	8.68E+01	0	0	1.61E-14	2.46E-04	8.55E-07	1.95E+00	2.22E+01	4.81E-17	1.59E-14	2.48E+01	2.62E+03
	1.18E+02	0	0	1.50E-14	0	5.11E-07	1.71E+00	2.20E+01	0	1.50E-14	2.54E+01	2.64E+03
	1.81E-12	0	0	7.99E-15	0	2.30E-08	3.27E-03	1.38E+01	0	7.99E-15	1.41E+01	1.68E+03
	3.55E+02	0	0	2.93E-14	7.39E-03	4.78E-06	5.76E+00	2.92E+01	3.33E-16	2.93E-14	3.24E+01	3.52E+03
3.0	6.71E+01	0	0	1.34E-14	0	2.17E-06	4.23E+00	2.42E+01	7.02E-08	1.43E-14	2.33E+01	2.70E+03
	1.81E-12	0	0	1.50E-14	0	1.21E-06	4.00E+00	2.26E+01	0	1.50E-14	2.40E+01	2.69E+03
	1.81E-12	0	0	7.99E-15	0	6.59E-08	1.22E-02	1.25E+01	0	7.99E-15	1.50E+01	1.80E+03
	3.55E+02	0	0	2.22E-14	0	1.90E-05	9.67E+00	4.05E+01	2.10E-06	2.22E-14	3.15E+01	3.55E+03
3.1	9.08E+01	0	3.52E-14	1.22E-14	0	4.91E-06	3.13E+00	2.38E+01	1.46E-06	1.29E-14	2.32E+01	2.76E+03
	5.92E+01	0	0	1.50E-14	0	2.84E-06	2.98E+00	2.23E+01	0	1.50E-14	2.32E+01	2.89E+03
	1.81E-12	0	0	7.99E-15	0	8.89E-08	1.34E-03	1.46E+01	0	7.99E-15	1.68E+01	1.84E+03
	3.55E+02	0	1.00E-12	2.22E-14	0	2.17E-05	1.07E+01	3.95E+01	4.11E-05	2.22E-14	3.10E+01	3.83E+03
3.2	6.71E+01	0	0	1.27E-14	1.92E-09	8.02E-06	5.29E+00	2.73E+01	4.93E-04	1.20E-14	2.33E+01	2.74E+03
	1.81E-12	0	0	1.50E-14	0	4.40E-06	5.23E+00	2.71E+01	0	1.50E-14	2.39E+01	2.79E+03
	1.81E-12	0	0	7.99E-15	0	4.79E-07	3.19E-03	1.89E+01	0	7.99E-15	1.20E+01	1.63E+03
	3.55E+02	0	0	2.22E-14	5.13E-08	3.79E-05	1.86E+01	3.69E+01	7.39E-03	1.50E-14	3.38E+01	3.48E+03
3.3	8.68E+01	0	8.49E-13	1.24E-14	6.57E-04	3.72E-05	8.10E+00	2.50E+01	3.42E-04	1.10E-14	2.46E+01	2.71E+03
	5.92E+01	0	1.77E-15	1.50E-14	0	2.29E-05	6.41E+00	2.48E+01	0	7.99E-15	2.42E+01	2.79E+03
	1.81E-12	0	0	7.99E-15	0	1.97E-06	5.25E-04	1.23E+01	0	7.99E-15	1.63E+01	1.76E+03
	2.36E+02	0	1.62E-11	2.22E-14	9.85E-03	2.22E-04	7.25E+01	3.87E+01	9.85E-03	2.22E-14	3.30E+01	3.71E+03
3.4	8.68E+01	0	1.66E-10	1.10E-14	0	1.28E-04	1.03E+01	2.60E+01	1.83E-07	9.65E-15	2.49E+01	2.54E+03
	5.92E+01	0	1.77E-15	7.99E-15	0	6.33E-05	8.30E+00	2.67E+01	0	7.99E-15	2.43E+01	2.47E+03
	1.81E-12	0	0	7.99E-15	0	1.69E-05	6.33E-03	1.51E+01	0	7.99E-15	1.63E+01	1.69E+03
	3.55E+02	0	4.94E-09	2.22E-14	0	7.50E-04	6.89E+01	3.79E+01	4.59E-06	1.50E-14	3.40E+01	3.50E+03
3.5	5.52E+01	0	4.32E-10	8.94E-15	2.46E-04	9.29E-04	1.26E+01	2.79E+01	9.21E-04	9.88E-15	2.77E+01	2.71E+03
	1.81E-12	0	1.37E-13	7.99E-15	0	5.26E-04	1.02E+01	2.79E+01	0	7.99E-15	2.81E+01	2.72E+03
	1.81E-12	0	0	7.99E-15	0	3.54E-05	1.19E-03	1.87E+01	0	7.99E-15	1.80E+01	1.46E+03
	3.55E+02	0	1.22E-08	1.50E-14	7.39E-03	6.21E-03	6.46E+01	4.11E+01	1.02E-02	1.50E-14	3.90E+01	4.09E+03
3.6	1.10E+02	1.05E-14	1.31E-08	8.70E-15	3.05E-07	2.75E-03	2.27E+01	2.70E+01	5.92E-05	8.70E-15	2.41E+01	2.66E+03
	1.18E+02	0	2.56E-11	7.99E-15	0	2.11E-03	1.30E+01	2.61E+01	0	7.99E-15	2.42E+01	2.66E+03
	1.81E-12	0	0	7.99E-15	0	5.61E-04	9.79E-01	1.47E+01	0	7.99E-15	1.59E+01	2.01E+03
	3.55E+02	3.16E-13	2.37E-07	1.50E-14	4.07E-06	8.38E-03	8.49E+01	4.41E+01	1.65E-03	2.22E-14	3.40E+01	3.37E+03
3.7	7.10E+01	3.55E-16	1.20E-08	9.05E-15	2.72E-05	1.69E-02	1.63E+01	2.88E+01	2.58E-04	9.05E-15	2.42E+01	2.65E+03
	1.81E-12	0	3.69E-10	7.99E-15	0	1.46E-02	1.38E+01	2.96E+01	0	7.99E-15	2.42E+01	2.64E+03
	1.81E-12	0	1.06E-14	7.99E-15	0	1.89E-03	1.55E-01	1.50E+01	0	4.44E-15	1.77E+01	2.00E+03
	3.55E+02	1.77E-15	2.48E-07	1.50E-14	4.00E-04	4.54E-02	7.10E+01	4.04E+01	7.39E-03	1.50E-14	3.08E+01	3.24E+03
3.8	8.29E+01	2.17E-13	2.32E-05	8.11E-15	5.07E-04	9.55E-02	2.08E+01	2.94E+01	3.90E-04	8.58E-15	2.60E+01	2.65E+03
	1.81E-12	0	3.14E-09	7.99E-15	0	6.22E-02	1.68E+01	2.82E+01	0	7.99E-15	2.64E+01	2.62E+03
	1.81E-12	0	3.96E-13	4.44E-15	0	1.35E-02	8.61E+00	1.75E+01	0	4.44E-15	1.85E+01	1.75E+03
	3.55E+02	2.63E-12	4.83E-04	1.50E-14	7.77E-03	3.77E-01	8.74E+01	4.63E+01	1.03E-02	1.50E-14	3.26E+01	3.78E+03

Appendix B.

Statistical results of the peer methods on 30-D classical problems. The results include the average, median, best, and worst achieved fitness values in 30 independent runs as well as the Wilcoxon rank sum tests. In Wilcoxon tests each peer approach is compared with CLAMS-II. Some minor differences in the results presented in Table 8 are due to the precision limitations of the variables in the implementations. Accordingly, we have considered values less than 5.0e-15, 5.0e-15, 5.0e-16, 1.0e-28, 5.0e-16, and 5.0e-16, respectively for the benchmarks f_2, f_3, f_5, f_8, f_9, and f_{13} as zero. Also, the results less than 0.1 of the obtained minimum in f_4 and f_{15} are set to their minimum in the Wilcoxon tests.

Table 8. Statistical results of the peer methods on 30-D classical problems. Average, Median, Minimum, and Maximum of the final obtained solutions along with the Wilcoxon test results are reported on each benchmark problem.

f	CLAMS-I	CLAMS-II	APSO	CLPSO	FIPS	FPSO	MPEDE
1	1.38E+2	6.71E+1	1.72E+3	4.27E-7	4.71E+3	7.99E+3	3.03E+2
	1.18E+2	1.81E-12	1.59E+3	3.24E-7	4.73E+3	7.91E+3	2.83E+2
	1.81E-12	1.81E-12	7.10E+2	1.18E-7	3.33E+3	6.61E+3	7.76E+1
	4.73E+2	3.55E+2	3.39E+3	1.34E-6	5.53E+3	9.22E+3	5.72E+2
	(+)8E-3		(+)2E-11	(=)2E-1	(+)2E-11	(+)2E-11	(+)2E-8
2	1.06E-15	0	4.04E+0	2.09E-3	4.66E+1	2.08E+1	1.51E+1
	0	0	3.97E+0	1.97E-3	4.34E+1	2.06E+1	1.55E+1
	0	0	0.99E+0	2.92E-4	2.42E+1	1.26E+1	1.05E+1
	3.55E-15	0	7.95E+0	4.97E-3	7.41E+1	2.80E+1	2.01E+1
	(=)NAN		(+)1E-12	(+)1E-12	(+)1E-12	(+)1E-12	(+)1E-12
3	2.05E-13	0	6.30E-10	1.32E-2	4.29E+1	2.55E+1	1.96E+1
	3.55E-15	0	0	1.10E-2	4.29E+1	2.50E+1	1.96E+1
	0	0	0	2.70E-3	2.91E+1	2.01E+1	1.43E+1
	5.98E-12	0	1.89E-8	5.46E-2	6.11E+1	4.10E+1	2.32E+1
	(+)3E-5		(=)3E-1	(+)1E-12	(+)1E-12	(+)1E-12	(+)1E-12
4	2.29E-14	1.34E-14	4.43E-2	9.20E-5	8.23E-15	6.38E-2	2.23E-7
	2.22E-14	1.50E-14	1.50E-14	9.48E-5	7.99E-15	6.06E-2	1.97E-7
	7.99E-15	7.99E-15	7.99E-15	4.81E-5	7.99E-15	2.41E-2	9.61E-8
	3.28E-14	2.22E-14	1.34E+0	1.39E-4	1.50E-14	1.24E-1	4.50E-7
	(=)NAN		(=)8E-2	(+)1E-12	(=)NAN	(+)1E-12	(+)1E-12
5	1.31E-3	0	9.85E-3	5.64E-6	3.27E-6	9.64E-1	4.93E-4
	0	0	9.85E-3	4.45E-6	2.39E-12	9.36E-2	2.05E-12
	0	0	0	7.27E-7	0	1.87E-2	3.15E-13
	1.72E-2	0	2.70E-2	3.95E-5	9.07E-5	2.69E-1	7.39E-3
	(=)8E-2		(+)5E-8	(+)1E-12	(+)2E-8	(+)1E-12	(+)1E-12
6	1.57E-32	1.70E-32	1.03E-2	1.44E-9	7.49E-28	9.81E-5	1.41E-14
	1.57E-32	1.57E-32	2.01E-31	1.34E-9	1.57E-32	8.04E-5	9.27E-15
	1.57E-32	1.57E-32	3.63E-32	3.61E-10	1.57E-32	9.70E-6	9.88E-16
	1.57E-32	3.63E-32	2.07E-1	2.55E-9	2.24E-26	2.56E-4	1.55E-13
	(=)NAN		(+)2E-6	(+)1E-12	(=)2E-1	(+)1E-12	(+)1E-12
7	1.34E-32	1.34E-32	3.29E-3	2.34E-8	1.34E-32	2.36E-3	5.66E-13
	1.34E-32	1.34E-32	1.11E-30	1.91E-8	1.34E-32	2.23E-3	2.59E-13
	1.34E-32	1.34E-32	3.29E-31	1.00E-8	1.34E-32	4.31E-4	1.45E-14
	1.34E-32	1.34E-32	4.39E-2	5.42E-8	1.34E-32	6.81E-3	5.11E-12
	(=)NAN		(+)1E-12	(+)1E-12	(=)NAN	(+)1E-12	(+)1E-12
8	0	4.20E-31	6.22E-29	3.80E-8	0	3.88E-3	5.39E-13
	0	0	5.04E-29	3.79E-8	0	3.81E-3	3.76E-13
	0	0	0	1.49E-8	0	1.03E-2	1.56E-13
	0	1.26E-29	1.64E-28	8.46E-8	0	1.20E-3	1.99E-12
	(=)NAN		(+)1E-2	(+)1E-12	(=)NAN	(+)1E-12	(+)1E-12
9	7.40E-18	0	4.57E-15	6.39E-6	0	7.99E-3	1.70E-6
	0	0	9.99E-16	6.65E-6	0	7.33E-3	1.49E-6
	0	0	0	3.96E-6	0	2.50E-3	8.40E-7
	2.22E-16	0	8.85E-14	9.45E-6	0	1.77E-3	3.75E-6
	(=)NAN		(+)1E-7	(+)1E-12	(=)NAN	(+)1E-12	(+)1E-12
10	7.07E-6	2.17E-6	7.47E-2	2.84E+3	9.99E+0	1.54E+2	1.88E-5
	3.04E-6	1.21E-6	5.18E-2	2.86E+3	9.03E+0	1.49E+2	3.79E-6

	3.54E-7 7.53E-5 (+)8E-4	6.59E-8 1.90E-5	5.60E-3 4.42E-1 (+)3E-11	2.00E+3 3.91E+3 (+)3E-11	2.10E+0 2.62E+1 (+)3E-11	4.82E+1 2.49E+2 (+)3E-11	3.99E-8 1.10E-4 (+)2E-2
11	4.31E+0 4.40E+0 2.76E-3 8.78E+0 (=)7E-1	4.23E+0 4.00E+0 1.22E-2 9.67E+0	3.40E+1 2.18E+1 7.21E-1 1.75E+2 (+)5E-10	4.71E+1 4.69E+1 1.80E+1 8.16E+1 (+)3E-11	2.58E+1 2.20E+1 1.89E+1 7.65E+1 (+)3E-11	5.16E+1 3.63E+1 3.03E+1 3.50E+2 (+)3E-11	2.13E+1 2.16E+1 1.88E+1 2.33E+1 (+)3E-11
12	2.28E+1 2.26E+1 1.27E+1 3.69E+1 (=)6E-1	2.42E+1 2.26E+1 1.25E+1 4.05E+1	5.04E+1 4.87E+1 2.98E+1 8.95E+1 (+)1E-10	4.91E+1 5.14E+1 2.74E+1 7.10E+1 (+)2E-10	8.52E+1 8.72E+1 3.58E+1 1.34E+2 (+)3E-11	2.37E+1 2.26E+1 1.23E+1 3.80E+1 (=)8E-1	5.44E+1 5.49E+1 3.87E+1 7.76E+1 (+)3E-11
13	1.56E-3 2.22E-16 0 1.23E-2 (+)6E-3	7.02E-8 0 0 2.10E-6	1.18E-2 7.39E-3 0 6.87E-2 (+)9E-7	8.03E-4 7.54E-4 1.82E-5 2.89E-3 (+)2E-12	5.73E-5 9.50E-11 0 7.80E-4 (+)7E-10	1.08E-1 8.85E-2 1.62E-2 3.59E-1 (+)2E-12	3.47E-11 1.45E-11 3.44E-12 2.35E-10 (+)5E-11
14	2.26E-14 2.22E-14 7.99E-15 3.99E-14 (=)NAN	1.43E-14 1.50E-14 7.99E-15 2.22E-14	1.53E+0 1.57E+0 7.99E-15 2.66E+0 (+)2E-10	1.01E-2 8.96E-3 4.21E-3 2.79E-2 (+)1E-12	7.99E-15 7.99E-15 7.99E-15 7.99E-15 (=)NAN	6.05E-2 5.55E-2 2.22E-2 1.47E-1 (+)1E-12	2.24E-7 1.91E-7 7.96E-8 4.05E-7 (+)1E-12
15	2.41E+1 2.41E+1 1.81E+1 3.25E+1 (=)5E-1	2.33E+01 2.40E+01 1.50E+01 3.15E+01	4.76E+1 4.65E+1 2.30E+1 7.90E+1 (+)4E-10	4.43E+1 4.21E+1 3.24E+1 6.19E+1 (+)3E-11	7.27E+1 7.09E+1 4.52E+1 1.04E+2 (+)3E-11	2.62E+1 2.66E+1 1.61E+1 3.60E+1 (+)3E-2	3.95E+1 3.95E+1 2.81E+1 4.98E+1 (+)5E-11
16	2.80E+3 2.88E+3 2.10E+3 3.82E+3 (=)3E-1	2.70E+03 2.69E+03 1.80E+03 3.55E+03	3.61E+3 3.56E+3 2.59E+3 5.36E+3 (+)4E-7	2.58E+3 2.60E+3 1.23E+3 3.23E+3 (=)4E-1	7.79E+3 7.82E+3 6.71E+3 8.69E+3 (+)3E-11	8.54E+3 8.58E+3 7.50E+3 9.59E+3 (+)3E-11	3.41E+3 3.40E+3 1.66E+3 4.50E+3 (+)9E-6

Appendix C.

Statistical results on 50-D classical problems.

Table 9. Statistical results of the peer methods on 50-D classical problems. Average, Median, Minimum, and Maximum of the final obtained solutions along with the Wilcoxon test results are reported on each benchmark problem.

f	CLAMS-I	CLAMS-II	APSO	CLPSO	FIPS	FPSO	MPEDE
1	1.58E+02 1.18E+02 2.18E-11 4.74E+02 (=)7E-02	1.38E+02 1.18E+02 1.82E-11 5.92E+02	2.99E+03 2.73E+03 7.10E+02 6.76E+03 (+)3E-11	2.18E-11 2.18E-11 2.18E-11 2.18E-11 (-)1E-04	8.61E+03 8.62E+03 6.70E+03 1.06E+04 (+)3E-11	1.40E+04 1.41E+04 1.22E+04 1.52E+04 (+)3E-11	4.31E-07 1.24E-07 6.58E-10 4.37E-06 (=)2E-01
2	2.07E-15 1.78E-15 0.00E+00 1.60E-14 (+)4E-02	5.92E-17 0.00E+00 0.00E+00 1.78E-15	2.42E+00 1.98E+00 9.94E-01 5.96E+00 (+)1E-12	7.49E-02 4.48E-02 3.23E-03 5.42E-01 (+)1E-12	9.37E+01 9.32E+01 5.82E+01 1.12E+02 (+)1E-12	4.28E+01 4.11E+01 3.06E+01 5.56E+01 (+)1E-12	4.07E-04 1.22E-04 1.45E-06 4.03E-03 (+)1E-12
3	4.97E-15 3.55E-15 0.00E+00 2.49E-14 (+)3E-04	2.37E-16 0.00E+00 0.00E+00 3.55E-15	1.77E-16 0.00E+00 0.00E+00 1.77E-15 (=)NaN	3.98E+00 4.06E+00 2.34E+00 6.62E+00 (+)1E-12	8.85E+01 8.93E+01 6.71E+01 1.14E+02 (+)1E-12	4.47E+01 4.42E+01 3.32E+01 5.50E+01 (+)1E-12	5.94E+00 5.71E+00 2.25E+00 1.18E+01 (+)1E-12
4	2.99E-14 2.93E-14 1.51E-14 4.35E-14 (=)NaN	1.82E-14 1.51E-14 7.99E-15 2.93E-14	7.33E-05 2.22E-14 1.50E-14 2.19E-03 (+)6E-03	1.85E-08 1.81E-08 1.33E-08 2.92E-08 (+)1E-12	1.43E-14 1.51E-14 7.99E-15 1.51E-14 (=)NaN	9.04E-02 8.25E-02 3.77E-02 2.48E-01 (+)1E-12	7.64E-15 7.99E-15 4.44E-15 7.99E-15 (=)NaN
5	1.23E-03 0.00E+00 0.00E+00 2.21E-02 (=)2E-01	0.00E+00 0.00E+00 0.00E+00 0.00E+00	1.49E-02 8.62E-03 0.00E+00 1.06E-01 (+)9E-07	8.45E-11 2.17E-11 5.20E-13 1.12E-09 (+)1E-12	1.47E-03 0.00E+00 0.00E+00 3.68E-02 (+)7E-05	1.76E-01 1.55E-01 4.22E-02 3.78E-01 (+)1E-12	9.03E-04 0.00E+00 0.00E+00 1.97E-02 (=)2E-01
6	9.42E-33 9.42E-33 9.42E-33 9.42E-33 (=)NaN	1.02E-32 9.42E-33 9.42E-33 2.18E-32	6.22E-03 1.13E-30 1.46E-31 6.22E-02 (+)1E-12	7.91E-19 6.88E-19 3.07E-19 1.92E-18 (+)1E-12	9.42E-33 9.42E-33 9.42E-33 9.42E-33 (=)NaN	1.37E-04 1.10E-04 1.80E-05 3.42E-04 (+)1E-12	2.07E-03 9.42E-33 9.42E-33 6.22E-02 (=)3E-01
7	3.45E-32	2.40E-32	4.39E-03	1.62E-17	1.24E-01	1.49E-03	1.35E-32

	1.35E-32	1.35E-32	3.86E-29	1.52E-17	1.35E-32	1.30E-03	1.35E-32
	1.35E-32	1.35E-32	4.12E-30	6.64E-18	1.35E-32	5.89E-04	1.35E-32
	3.29E-31	3.29E-31	4.39E-02	3.97E-17	3.60E+00	3.81E-03	1.35E-32
	(=)6E-01		(+)2E-12	(+)2E-12	(=)3E-01	(+)2E-12	(=)3E-01
	2.10E-30	1.26E-30	2.32E-27	4.29E-15	0.00E+00	1.64E-01	0.00E+00
	0.00E+00	0.00E+00	5.55E-28	3.81E-15	0.00E+00	1.22E-01	0.00E+00
8	0.00E+00	0.00E+00	2.14E-28	1.79E-15	0.00E+00	4.38E-02	0.00E+00
	5.05E-29	1.26E-29	2.78E-26	1.03E-14	0.00E+00	4.17E-01	0.00E+00
	(=)NaN		(+)1E-12	(+)1E-12	(=)NaN	(+)1E-12	(=)NaN
	0.00E+00	0.00E+00	1.60E-10	1.43E-09	0.00E+00	1.85E-01	0.00E+00
	0.00E+00	0.00E+00	1.74E-14	1.42E-09	0.00E+00	1.78E-01	0.00E+00
9	0.00E+00	0.00E+00	2.88E-15	8.33E-10	0.00E+00	9.34E-02	0.00E+00
	0.00E+00	0.00E+00	4.81E-09	2.07E-09	0.00E+00	3.58E-01	0.00E+00
	(=)NaN		(+)1E-12	(+)1E-12	(=)NaN	(+)1E-12	(=)NaN
	8.67E-07	4.73E-08	9.50E-02	2.52E+03	9.99E+00	1.54E+02	2.24E-07
	4.25E-07	3.06E-08	7.18E-02	2.40E+03	9.03E+00	1.50E+02	1.73E-07
10	3.69E-08	2.62E-09	1.56E-02	1.43E+03	2.10E+00	4.83E+01	7.67E-09
	5.24E-06	2.00E-07	4.48E-01	4.06E+03	2.63E+01	2.49E+02	1.05E-06
	(+)2E-09		(+)3E-11	(+)3E-11	(+)3E-11	(+)3E-11	(+)1E-05
	9.00E+00	9.24E-01	5.16E+01	5.01E+01	4.67E+01	1.06E+02	1.83E+01
	6.23E+00	1.05E-01	5.25E+01	3.85E+01	3.60E+01	9.37E+01	1.67E+01
11	6.13E-04	1.95E-04	2.39E-02	1.70E+01	1.98E+01	6.69E+01	7.55E+00
	7.04E+01	7.01E+00	1.30E+02	1.54E+02	9.14E+01	2.16E+02	7.59E+01
	(+)8E-07		(+)2E-09	(+)3E-11	(+)3E-11	(+)3E-11	(+)3E-11
	4.28E+01	4.50E+01	8.99E+01	1.00E+02	1.70E+02	4.63E+01	4.28E+01
	4.38E+01	4.55E+01	9.00E+01	9.90E+01	1.73E+02	4.52E+01	4.33E+01
12	2.41E+01	2.99E+01	4.37E+01	7.56E+01	1.06E+02	3.27E+01	2.49E+01
	5.55E+01	6.32E+01	1.48E+02	1.37E+02	2.19E+02	6.23E+01	6.17E+01
	(=)4E-01		(+)2E-10	(+)3E-11	(+)3E-11	(=)6E-01	(=)3E-01
	7.39E-04	1.47E-13	1.18E-02	7.18E-06	2.73E-04	2.26E-01	5.75E-04
	3.33E-16	1.11E-16	3.33E-16	3.64E-06	5.99E-12	1.93E-01	0.00E+00
13	1.11E-16	0.00E+00	0.00E+00	3.57E-08	1.11E-16	3.97E-02	0.00E+00
	1.48E-02	4.40E-12	8.01E-02	3.98E-05	8.14E-03	5.00E-01	9.86E-03
	(=)6E-02		(+)7E-04	(+)3E-12	(+)3E-08	(+)3E-12	(=)7E-01
	3.08E-14	1.97E-14	2.14E+00	1.38E-04	1.42E-14	9.20E-02	7.99E-15
	2.93E-14	2.22E-14	2.17E+00	1.06E-04	1.51E-14	8.82E-02	7.99E-15
14	2.22E-14	1.51E-14	8.79E-01	5.96E-06	7.99E-15	3.80E-02	7.99E-15
	4.35E-14	2.93E-14	3.25E+00	6.03E-04	1.51E-14	1.76E-01	7.99E-15
	(=)NaN		(+)1E-12	(+)1E-12	(=)NaN	(+)1E-12	(=)NaN
	4.17E+01	4.64E+01	7.53E+01	9.40E+01	1.60E+02	5.13E+01	4.96E+01
	4.11E+01	4.62E+01	7.05E+01	9.30E+01	1.58E+02	5.11E+01	5.01E+01
15	2.80E+01	3.01E+01	4.20E+01	6.76E+01	1.12E+02	4.21E+01	2.85E+01
	5.21E+01	6.40E+01	1.39E+02	1.28E+02	2.06E+02	6.63E+01	6.33E+01
	(-)2E-02		(+)2E-08	(+)3E-11	(+)3E-11	(+)2E-02	(=)1E-01
	4.66E+03	4.61E+03	6.24E+03	4.02E+03	1.35E+04	1.47E+04	4.76E+03
	4.75E+03	4.45E+03	5.94E+03	4.02E+03	1.35E+04	1.46E+04	4.76E+03
16	3.80E+03	3.76E+03	3.99E+03	3.23E+03	1.10E+04	1.30E+04	3.24E+03
	5.28E+03	5.62E+03	1.09E+04	4.77E+03	1.50E+04	1.64E+04	6.26E+03
	(=)4E-01		(+)1E-07	(-)1E-04	(+)3E-11	(+)3E-11	(=)3E-01

Appendix D.

Statistical results on CEC2013 benchmark set in terms of the average, median, minimum, maximum, and standard deviation of the error obtained in different runs (unavailable statistical information are identified with dashes).

Table 10. Statistical results on CEC2013 at 30D. Average, median, minimum, maximum, std of error in 51 independent runs.

f	CLAMS-I	CLAMS-II	F_k-PSO	ABC-SPSO	NGDE/rand/1	CooA	MEABC	IVbBoPSO	CoFFWA
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.83E+02	0.00E+00
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-
	2.46E+04	5.08E+04	1.59E+06	8.78E+05	5.67E+04	1.03E+05	1.23E+06	1.17E+08	8.80E+05
	2.08E+04	4.60E+04	1.46E+06	2.84E+05	-	-	-	-	-
2	7.01E+03	1.65E+04	3.19E+05	1.00E+05	-	-	-	-	-
	1.17E+05	1.50E+05	3.53E+06	7.62E+06	-	-	-	-	-
	1.66E+04	2.69E+04	8.03E+05	1.69E+06	3.78E+04	1.68E+05	-	-	-

	1.02E+07	3.00E+06	2.40E+08	5.16E+07	1.23E+05	7.84E+06	1.40E+08	6.07E+10	8.04E+07
	3.10E+06	6.08E+05	9.91E+07	2.44E+07	-	-	-	-	-
3	5.90E+00	3.24E+01	1.20E+05	4.55E+05	-	-	-	-	-
	9.02E+07	2.93E+07	1.99E+09	3.93E+08	-	-	-	-	-
	1.79E+07	5.50E+06	3.71E+08	8.00E+07	6.26E+05	8.38E+06	-	-	-
	3.37E+02	1.12E+02	4.78E+02	6.02E+03	5.87E+00	6.67E+02	8.35E+04	3.46E+04	2.01E+03
	2.71E+02	8.28E+01	4.43E+02	6.19E+03	-	-	-	-	-
4	6.42E+01	2.31E+01	1.95E+02	1.90E+03	-	-	-	-	-
	1.06E+03	4.19E+02	1.11E+03	1.08E+04	-	-	-	-	-
	2.41E+02	7.79E+01	1.96E+02	2.30E+03	4.40E+00	1.24E+03	-	-	-
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.67E-04	0.00E+00	3.34E+02	7.41E-04
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.57E-04	-	-	-
	4.07E+00	9.23E-01	2.99E+01	1.09E+01	6.91E+00	1.31E+01	1.01E+01	2.00E+02	2.47E+01
	3.51E+00	3.74E-01	2.45E+01	1.17E+01	-	-	-	-	-
6	9.99E-03	1.09E-05	3.55E+00	4.19E-04	-	-	-	-	-
	2.64E+01	2.64E+01	8.13E+01	6.79E+01	-	-	-	-	-
	4.96E+00	3.66E+00	1.76E+01	1.09E+01	5.86E+00	1.19E+01	-	-	-
	3.03E+01	1.05E+01	6.39E+01	5.11E+01	3.88E+00	4.19E+01	9.23E+01	2.56E+02	8.99E+01
	2.85E+01	8.10E+00	6.18E+01	4.64E+01	-	-	-	-	-
7	9.09E+00	2.84E+00	4.72E+00	1.71E+01	-	-	-	-	-
	6.65E+01	4.57E+01	1.55E+02	9.94E+01	-	-	-	-	-
	1.21E+01	6.99E+00	3.09E+01	2.04E+01	3.77E+00	1.43E+01	-	-	-
	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01
	2.09E+01	2.09E+01	2.09E+01	2.09E+01	-	-	-	-	-
8	2.08E+01	2.08E+01	2.08E+01	2.08E+01	-	-	-	-	-
	2.10E+01	2.10E+01	2.10E+01	2.10E+01	-	-	-	-	-
	4.68E-02	5.89E-02	6.28E-02	4.92E-02	4.81E-02	1.04E-03	-	-	-
	1.79E+01	1.16E+01	1.85E+01	2.95E+01	2.03E+01	1.60E+01	2.88E+01	2.82E+01	2.40E+01
	1.80E+01	1.10E+01	1.86E+01	3.01E+01	-	-	-	-	-
9	9.20E+00	6.18E+00	1.20E+01	1.93E+01	-	-	-	-	-
	2.77E+01	2.07E+01	2.42E+01	3.29E+01	-	-	-	-	-
	4.82E+00	3.05E+00	2.69E+00	2.62E+00	4.67E+00	1.88E+00	-	-	-
	7.74E-02	6.49E-02	2.29E-01	1.32E-01	5.66E-02	4.56E-02	5.57E+00	6.58E+02	4.10E-02
	7.64E-02	5.91E-02	2.04E-01	1.18E-01	-	-	-	-	-
10	2.22E-02	1.97E-02	5.18E-02	2.46E-02	-	-	-	-	-
	1.80E-01	1.50E-01	5.91E-01	2.96E-01	-	-	-	-	-
	3.68E-02	3.33E-02	1.32E-01	6.23E-02	3.01E-02	6.08E-02	-	-	-
	0.00E+00	0.00E+00	2.36E+01	0.00E+00	2.75E+01	1.01E+01	0.00E+00	7.95E+01	9.90E+01
	0.00E+00	0.00E+00	2.39E+01	0.00E+00	-	-	-	-	-
11	0.00E+00	0.00E+00	6.96E+00	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	4.28E+01	0.00E+00	-	-	-	-	-
	0.00E+00	0.00E+00	8.76E+00	0.00E+00	8.19E+00	2.14E+01	-	-	-
	4.77E+01	5.37E+01	5.64E+01	6.44E+01	3.27E+01	5.36E+01	2.07E+02	2.30E+02	1.40E+02
	4.83E+01	5.22E+01	5.47E+01	6.22E+01	-	-	-	-	-
12	2.58E+01	3.30E+01	2.89E+01	3.23E+01	-	-	-	-	-
	7.47E+01	9.20E+01	9.45E+01	9.49E+01	-	-	-	-	-
	9.16E+00	1.27E+01	1.51E+01	1.48E+01	8.86E+00	1.91E+01	-	-	-
	7.78E+01	8.10E+01	1.23E+02	1.15E+02	7.92E+01	1.13E+02	2.29E+02	2.80E+02	2.50E+02
	7.54E+01	8.13E+01	1.24E+02	1.14E+02	-	-	-	-	-
13	5.22E+01	4.29E+01	6.96E+01	6.66E+01	-	-	-	-	-
	1.22E+02	1.21E+02	1.61E+02	1.62E+02	-	-	-	-	-
	1.56E+01	1.80E+01	2.19E+01	2.24E+01	2.30E+01	3.46E+01	-	-	-
	2.92E+00	2.42E+00	7.04E+02	1.55E+01	9.42E+02	2.03E+03	1.37E+01	2.09E+03	2.70E+03
	1.77E+00	1.64E+00	6.79E+02	1.43E+01	-	-	-	-	-
14	1.88E-01	1.71E-01	2.44E+02	3.71E+00	-	-	-	-	-
	9.03E+00	1.53E+01	1.19E+03	3.26E+01	-	-	-	-	-
	2.52E+00	2.59E+00	2.38E+02	6.13E+00	3.26E+02	5.62E+02	-	-	-
	3.88E+03	3.38E+03	3.42E+03	3.55E+03	7.18E+03	2.65E+03	3.41E+02	6.62E+03	3.37E+03
	3.99E+03	3.40E+03	3.50E+03	3.59E+03	-	-	-	-	-
15	2.32E+03	2.27E+03	2.26E+03	2.92E+03	-	-	-	-	-
	5.05E+03	4.34E+03	4.36E+03	4.21E+03	-	-	-	-	-
	5.78E+02	4.41E+02	5.16E+02	3.04E+02	2.32E+02	4.90E+02	-	-	-
	2.10E+00	2.18E+00	8.48E-01	1.03E+00	2.50E+00	1.29E-01	1.44E+00	2.50E+00	4.56E-01
	2.12E+00	2.20E+00	8.56E-01	1.06E+00	-	-	-	-	-
16	1.31E+00	1.47E+00	2.43E-01	5.55E-01	-	-	-	-	-
	2.67E+00	2.76E+00	1.22E+00	1.54E+00	-	-	-	-	-
	2.85E-01	3.32E-01	2.20E-01	2.01E-01	2.60E-01	1.22E-01	-	-	-
	3.05E+01	3.04E+01	5.26E+01	3.09E+01	5.53E+01	1.37E+02	3.04E+01	1.85E+02	1.10E+02
	3.05E+01	3.04E+01	5.19E+01	3.09E+01	-	-	-	-	-
17	3.04E+01	3.04E+01	3.89E+01	3.06E+01	-	-	-	-	-
	3.06E+01	3.05E+01	7.25E+01	3.12E+01	-	-	-	-	-
	3.29E-02	1.33E-02	7.11E+00	1.23E-01	6.47E+00	1.95E+01	-	-	-
18	1.07E+02	1.14E+02	6.81E+01	9.01E+01	1.91E+02	1.15E+02	1.80E+02	3.64E+02	1.80E+02
	1.07E+02	1.13E+02	6.79E+01	9.03E+01	-	-	-	-	-

	7.32E+01	9.08E+01	5.00E+01	6.76E+01	-	-	-	-	-
	1.32E+02	1.42E+02	8.94E+01	1.09E+02	-	-	-	-	-
	1.30E+01	1.13E+01	9.68E+00	8.95E+00	2.31E+01	2.91E+01	-	-	-
	1.45E+00	1.53E+00	3.12E+00	1.71E+00	2.77E+00	3.93E+00	3.94E-01	2.08E+01	6.51E+00
	1.74E+00	2.02E+00	2.92E+00	1.56E+00	-	-	-	-	-
19	1.09E-01	1.29E-01	1.62E+00	9.73E-01	-	-	-	-	-
	2.27E+00	2.45E+00	6.69E+00	3.62E+00	-	-	-	-	-
	7.00E-01	8.53E-01	9.83E-01	4.68E-01	8.21E-01	1.25E+00	-	-	-
	1.08E+01	1.03E+01	1.20E+01	1.11E+01	1.13E+01	1.06E+01	1.56E+01	1.41E+01	1.32E+01
	1.07E+01	1.03E+01	1.21E+01	1.11E+01	-	-	-	-	-
20	9.33E+00	7.82E+00	1.02E+01	9.18E+00	-	-	-	-	-
	1.19E+01	1.18E+01	1.40E+01	1.26E+01	-	-	-	-	-
	6.63E-01	7.15E-01	9.26E-01	7.60E-01	5.47E-01	1.77E-01	-	-	-
	3.17E+02	2.83E+02	3.11E+02	3.18E+02	3.31E+02	2.92E+02	2.10E+02	8.99E+02	2.06E+02
	3.00E+02	3.00E+02	3.00E+02	3.00E+02	-	-	-	-	-
21	1.00E+02	1.00E+02	2.00E+02	2.00E+02	-	-	-	-	-
	4.44E+02	4.44E+02	4.44E+02	4.44E+02	-	-	-	-	-
	7.77E+01	6.59E+01	7.92E+01	7.53E+01	1.02E+02	7.30E+01	-	-	-
	8.45E+01	7.71E+01	8.59E+02	8.41E+01	9.76E+02	3.05E+03	1.78E+01	2.30E+03	3.32E+03
	1.07E+02	1.05E+02	8.56E+02	8.72E+01	-	-	-	-	-
22	8.32E+00	1.06E+01	2.45E+02	2.85E+01	-	-	-	-	-
	2.04E+02	1.16E+02	1.48E+03	1.35E+02	-	-	-	-	-
	4.08E+01	3.74E+01	3.10E+02	3.90E+01	4.42E+02	3.11E+02	-	-	-
	3.88E+03	3.89E+03	3.57E+03	4.18E+03	7.26E+03	2.96E+03	5.16E+03	6.16E+03	4.47E+03
	3.86E+03	3.85E+03	3.45E+03	4.29E+03	-	-	-	-	-
23	2.23E+03	2.55E+03	2.47E+03	2.49E+03	-	-	-	-	-
	5.06E+03	5.81E+03	4.98E+03	5.26E+03	-	-	-	-	-
	5.02E+02	6.44E+02	5.90E+02	5.62E+02	3.75E+02	5.41E+02	-	-	-
	2.41E+02	2.37E+02	2.48E+02	2.51E+02	2.13E+02	2.02E+02	2.81E+02	2.73E+02	2.68E+02
	2.42E+02	2.37E+02	2.48E+02	2.50E+02	-	-	-	-	-
24	2.14E+02	2.10E+02	2.31E+02	2.16E+02	-	-	-	-	-
	2.67E+02	2.66E+02	2.69E+02	2.79E+02	-	-	-	-	-
	1.24E+01	1.19E+01	8.11E+00	1.43E+01	5.22E+00	7.10E+00	-	-	-
	2.67E+02	2.60E+02	2.49E+02	2.75E+02	2.61E+02	2.40E+02	2.74E+02	2.72E+02	2.94E+02
	2.66E+02	2.59E+02	2.49E+02	2.75E+02	-	-	-	-	-
25	2.53E+02	2.44E+02	2.33E+02	2.55E+02	-	-	-	-	-
	2.88E+02	2.85E+02	2.65E+02	2.97E+02	-	-	-	-	-
	8.66E+00	9.37E+00	7.82E+00	9.76E+00	1.37E+01	6.46E+01	-	-	-
	2.15E+02	2.02E+02	2.95E+02	2.60E+02	2.04E+02	2.00E+02	2.01E+02	3.70E+02	2.13E+02
	2.00E+02	2.00E+02	3.41E+02	2.00E+02	-	-	-	-	-
26	2.00E+02	2.00E+02	2.00E+02	2.00E+02	-	-	-	-	-
	3.32E+02	3.13E+02	3.65E+02	3.77E+02	-	-	-	-	-
	4.10E+01	1.59E+01	7.06E+01	7.62E+01	2.10E+01	5.44E-02	-	-	-
	5.11E+02	4.88E+02	7.76E+02	9.10E+02	4.62E+02	3.74E+02	4.02E+02	1.02E+03	8.71E+02
	5.17E+02	4.55E+02	7.73E+02	9.58E+02	-	-	-	-	-
27	3.56E+02	3.52E+02	6.35E+02	4.92E+02	-	-	-	-	-
	6.35E+02	8.68E+02	9.99E+02	1.13E+03	-	-	-	-	-
	6.52E+01	1.07E+02	7.11E+01	1.62E+02	1.00E+02	5.91E+01	-	-	-
	3.01E+02	2.76E+02	4.01E+02	3.33E+02	3.00E+02	3.00E+02	3.00E+02	1.58E+03	2.84E+02
	3.00E+02	3.00E+02	3.00E+02	3.00E+02	-	-	-	-	-
28	1.00E+02	1.00E+02	1.00E+02	1.00E+02	-	-	-	-	-
	1.34E+03	1.30E+03	1.50E+03	1.51E+03	-	-	-	-	-
	1.60E+02	2.26E+02	3.48E+02	2.32E+02	8.96E-06	2.02E-05	-	-	-

References

[1] Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp 1942–1948

[2] Shi Y, Eberhart RC (1998) A modified particle swarm optimizer. In: Proceedings of the IEEE international conference on evolutionary computation, Piscataway, NJ, pp 69–73

[3] Angeline P (1998) Evolutionary optimization versus particle swarm optimization: philosophy and performance difference. In: Proceedings of the seventh annual conference on evolutionary programming, pp 601–610

[4] Li C, Yang S, Nguyen TT (2012) A self-learning particle swarm optimizer for global optimization problems. IEEE Trans Syst Man Cybern Part B Cybern 42:627-646

[5] Li C, Yang S (2009) An adaptive learning particle swarm optimizer for function optimization. In: Proceedings of 2009 IEEE Congress on Evolutionary Computation. IEEE, pp 381-388

[6] Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler, maybe better. IEEE Trans Evol Comput 8:204-210

[7] De Oca MAM, Stutzle T, Birattari M, Dorigo M (2009) Frankenstein's PSO: a composite particle swarm optimization algorithm. IEEE Trans Evol Comput 13:1120-1132

[8] Lim WH, Isa NaM (2014) Particle swarm optimization with increasing topology connectivity. Eng Appl Artif Intell 27:80-102

[9] Beigy H, Meybodi MR (2004) A mathematical framework for cellular learning automata. Adv Complex Syst 7:295-319

[10] Beigy H, Meybodi MR (2008) Asynchronous cellular learning automata. Automatica 44:1350-1357

[11] Beigy H, Meybodi MR (2009) Cellular learning automata based dynamic channel assignment algorithms. Int J Comput Intell Appl 8:287-314
[12] Narendra KS, Thathachar MA (2012) Learning automata: an introduction. Courier Corporation
[13] Thathachar MA, Sastry PS (2011) Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media
[14] Beigy H, Meybodi MR (2003) Open synchronous cellular learning automata. J Comput Sci Eng 1:39-51
[15] Arumugam MS, Rao M (2008) On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems. Appl Soft Comput 8:324-336
[16] Yang X, Yuan J, Yuan J, Mao H (2007) A modified particle swarm optimizer with dynamic adaptation. Appl Math Comput 189:1205-1213
[17] Hashemi AB, Meybodi MR (2011) A note on the learning automata based algorithms for adaptive parameter selection in PSO. Appl Soft Comput 11:689-705
[18] Zhan Z-H, Zhang J, Li Y, Chung HS (2009) Adaptive particle swarm optimization. IEEE Trans Syst Man Cybern Part B Cybern 39:1362-1381
[19] Xu G (2013) An adaptive parameter tuning of particle swarm optimization algorithm. Appl Math Comput 219:4560-4569
[20] Piperagkas GS, Georgoulas G, Parsopoulos KE, Stylios CD, Likas A (2012) Integrating particle swarm optimization with reinforcement learning in noisy problems. In: Proceedings of the 14th annual conference on Genetic and evolutionary computation. ACM, pp 65-72
[21] Vafashoar R, Meybodi MR (2016) Multi swarm bare bones particle swarm optimization with distribution adaption. Appl Soft Comput 47:534-552
[22] Janson S, Middendorf M (2005) A hierarchical particle swarm optimizer and its adaptive variant. IEEE Trans Syst Man Cybern Part B Cybern 35:1272-1282
[23] Lim WH, Isa NaM (2014) Particle swarm optimization with adaptive time-varying topology connectivity. Appl Soft Comput 24:623-642
[24] De Oca MAM, Stutzle T, Birattari M, Dorigo M (2009) Frankenstein's PSO: a composite particle swarm optimization algorithm. IEEE Trans Evol Comput 13:1120-1132
[25] Lim WH, Isa NaM (2014) Particle swarm optimization with increasing topology connectivity. Eng Appl Artif Intell 27:80-102
[26] Tomassini M (2005) Spatially structured evolutionary algorithms: artificial evolution in space and time (Natural Computing Series). Springer-Verlag, New York
[27] Chen H, Zhu Y, Hu K (2010) Discrete and continuous optimization based on multi-swarm coevolution. Nat Comput 9:659-682
[28] Li J, Xiao X (2008) Multi-swarm and multi-best particle swarm optimization algorithm. In: Proceedings of Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on. IEEE, pp 6281-6286
[29] Niu B, Zhu Y, He X, Wu QH (2007) MCPSO: A multi-swarm cooperative particle swarm optimizer. Appl Math Comput 185:1050-1062
[30] Zhang J, Ding X (2011) A multi-swarm self-adaptive and cooperative particle swarm optimization. Eng Appl Artif Intell 24:958-967
[31] Wang H, Zhao X, Wang K, Xia K, Tu X (2014) Cooperative velocity updating model based particle swarm optimization. Appl Intell 40 (2):322-342
[32] Zhao X, Liu Z, Yang X (2014) A multi-swarm cooperative multistage perturbation guiding particle swarm optimizer. Appl Soft Comput 22:77-93
[33] Cheung NJ, Ding X-M, Shen H-B (2015) A supervised particle swarm algorithm for real-parameter optimization. Appl Intell 43 (4):825-839
[34] Wilke DN, Kok S, Groenwold AA (2007) Comparison of linear and classical velocity update rules in particle swarm optimization: notes on diversity. Int J Numer Methods Eng 70 (8):962-984
[35] Liang J, Qu B, Suganthan P, Hernández-Díaz AG (2013) Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report 201212
[36] Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10:281-295
[37] Leung Y-W, Wang Y (2001) An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Trans Evol Comput 5:41-53
[38] Salomon R (1996) Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. BioSyst 39:263-278
[39] Wu G, Mallipeddi R, Suganthan PN, Wang R, Chen H (2016) Differential evolution with multi-population based ensemble of mutation strategies. Inf Sci 329:329-345
[40] Wang H, Wu Z, Rahnamayan S, Sun H, Liu Y, Pan J-s (2014) Multi-strategy ensemble artificial bee colony algorithm. Inf Sci 279:587-603
[41] Zheng S, Li J, Janecek A, Tan Y (2015) A cooperative framework for fireworks algorithm. IEEE/ACM Trans Comput Biol Bioinf 14: 27-41
[42] Gunasundari S, Janakiraman S, Meenambal S (2016) Velocity Bounded Boolean Particle Swarm Optimization for improved feature selection in liver and kidney disease diagnosis. Expert Syst Appl 56:28-47
[43] Feng X, Zou R, Yu H (2015) A novel optimization algorithm inspired by the creative thinking process. Soft Comput 19 (10):2955-2972
[44] Cai Y, Zhao M, Liao J, Wang T, Tian H, Chen Y (2016) Neighborhood guided differential evolution. Soft Comput:1-44
[45] El-Abd M Testing a particle swarm optimization and artificial bee colony hybrid algorithm on the CEC13 benchmarks. In: Evolutionary Computation (CEC), 2013 IEEE Congress on, 2013. IEEE, pp 2215-2220
[46] Nepomuceno FV, Engelbrecht AP A self-adaptive heterogeneous pso for real-parameter optimization. In: Evolutionary Computation (CEC), 2013 IEEE Congress on, 2013. IEEE, pp 361-368
[47] Shannon C (1948) A mathematical theory of communication. Bell Syst Tech J 27: 379-423
[48] Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1-30