Article

A cooperation of multileader fruit fly and the probabilistic random walk with adaptive normalization for solving solution of the unconstrained optimization problems

Wirote Apinantanakon ¹, Khamron Sunat ^{2,*} and Sirapat Chiewchanwattana ²

¹ Department of Computer Science, Faculty of Science, Khon Kaen University, Thailand 40002; wirotta@gmail.com

² Department of Computer Science, Faculty of Science, Khon Kaen University, Thailand 40002; khamron_sunat@yahoo.com

* Correspondence: khamron_sunat@yahoo.com; Tel.: +66-80-461-3391

Abstract: A swarm based nature-inspired optimization algorithm namely fruit fly optimization algorithm (FOA) has simple structure and ease of implementation. However, FOA has a low success rate and a slow convergence because FOA generates new positions around the best location using fixed search radius. Several improved FOAs have been proposed. But their exploration ability is questionable. To make the search process to transit from the exploration phase to the exploitation phase smoothly, this paper proposes a new FOA constructed from a cooperation of the multileader and the probabilistic random walk strategies (CPFOA). It has two population types working together. CPFOA's performance is evaluated by 18 well-known standard benchmark, and 30 CEC'2017 functions. The results showed that CPFOA outperforms both the original FOA and its variants in terms of convergence speed and performance accuracy. The results base on CEC'2017 show that CPFOA can achieve a very promising accuracy when compared with the well-known competitive algorithms. CPFOA is applied to optimize two applications; the MLPs classifying real datasets and extracting parameters of T-S fuzzy system for modelling Box and Jenkins gas furnace data set. CPFOA can find parameters having a very high quality compared with the best known competitive algorithms.

Keywords: nature-inspired optimization algorithm; fruit fly optimization algorithm; multileader strategy; random walk; cooperative algorithm

1. Introduction

Over the last few decades, researchers have started adapting knowledge from natural phenomena for the development of optimization techniques. The main concepts of the aptly named sources 'nature-inspired algorithms' have been observed within successful biological systems. Accordingly, most nature-inspired algorithms are biologically inspired, or bio-inspired, and mimic specific behavior in nature. Examples of such popular nature-inspired algorithms include the particle swarm optimization algorithm (PSO) [1], which was inspired by the social behavior of flocking birds, or schooling fish; the ant colony optimization algorithm (ACO) [2], which mimics an ant colony's behavior in their search for food; the artificial bee colony algorithm (ABC) [3], motivated by the intelligent behavior of a honey bee swarm; the cuckoo search algorithm (CS) [4], inspired by the parasitic bio-interactions of a cuckoo species that lays their eggs in the nests of other host birds; and the bat-inspired algorithm (BA) [5], which was inspired by the echolocation behavior of bats, to name but a few. However, each nature-inspired algorithm has different capabilities when it comes to finding solutions, which depend on the personal ability of the living things in nature. Developing a successful, modern nature-inspired algorithm is a challenging task, even today, as there is no one particular nature-inspired algorithm capable of solving every scientific problem. Hence the continual development of a new algorithm is required. As can be seen the details and the broadly classified the nature-inspired algorithms, which presented the development of algorithms into the current modern literatures [6,7].

In 2011, a swarm intelligence method-based stochastic optimization technique namely the fruit fly optimization algorithm (FOA) was proposed by Pan [8]. It is based upon the foraging behavior of fruit flies. FOA is very user-friendly because of its simplicity and shortness. FOA can be easily understood by most researchers in this field. This also means that it can be easily implemented into program code, when compared with other well-known algorithms such as differential evolution (DE) [9], genetic algorithm (GA) [10] and particle swarm optimization (PSO). The FOA has achieved success in several applications; including research into optimization problems [11-14], neural network parameters optimization [15-17], swarm techniques for mini autonomous surface vehicles (ASVs) [18], identification of dynamic protein complexes based on fruit fly optimization algorithm [19], support vector regression with fruit fly optimization algorithm for seasonal electricity consumption forecasting [20], efficient truss optimization using the contrast-based fruit fly optimization algorithm [21] and improved fruit fly optimization algorithm for application to structural engineering design optimization problems [22,23], a prediction model for China agricultural output value based on the optimization of neural network [24], and a novel phase angle-encoded fruit fly optimization algorithm with mutation adaptation mechanism applied to UAV path planning [25].

Pan's FOA has a good structure and mechanism for the solution finding of optimization problems. However, the algorithm is prone to trapping into a local extreme and premature convergence, since FOA generates new fruit flies swarm around the current best solution by using random uniform distribution with a fixed radius. Especially when FOA deals with multi-dimensional and complex optimization problems.

The drawback caused by the fixed radius updating has encouraged many researches effort into various dynamic radius updating techniques to improve the FOA, e.g., an improved fruit fly optimization algorithm for solving optimization problems (LGMS-FOA) [11], an improved fruit fly optimization algorithm for continuous function optimization problems (IFFO) [12], a novel multi-swarm fruit fly optimization algorithm (MFOA) [14], a novel multi-scale cooperative mutation fruit fly optimization algorithm (MSFOA) [26], and a novel fruit fly optimization algorithm with trend search and co-evolution (CEFOA) [27].

There is one remaining imperfection that is caused by the single leader usage, although the FOA variants showed that the dynamic radius updating can improve the quality of the produced solutions. The single leader usage makes FOA to lack of diversity when the search process has to deal with a multi-dimensional or a complex optimization problem. In this paper, we proposed a novel FOA that uses a multileader instead a single leader and the probabilistic random walk radius

updating instead of random uniform dynamic radius updating (CPFOA). The details of proposed CPFOA are described as in section 3.

The remaining of the paper are as follows: Section 2 presents the FOA; Section 3 presents the strategies for CPFOA construction, the multileader, and the probabilistic random walk strategies; Section 4 explains the evaluation process of algorithms and settings; Section 5 shows the experimental results and discussion; Section 6 shows two applications of CPFOA; and lastly, Section 7 concludes the paper.

2. The Fruit Fly Optimization Algorithm

The drosophila optimized algorithm or fruit fly optimization algorithm (FOA) was developed in 2011 by Pan [8]. FOA determines global optimization based on the foraging behavior of fruit flies. Compared to other species, the fruit fly possesses a keener sense of smell and sight in the search for their food. Their drosophila olfactory organ can detect a food source as far as 40 kilometers away, which triggers a flight reaction toward the target location. For the intelligent sense of fruit flies, the novel FOA optimization algorithm is inspired and established through the simple behavior of fruit flies' search for food.

The FOA's process is similar to as with the other swarm optimization algorithms. The first phase of the fruit flies' quest for food is initiated with a random uniform distribution [6,28-32], with no specific position or direction. In the second phase, the fruit fly with the best sense of smell or the *best fitness* from the first phase within the group is determined. The fruit fly with the best sense of smell is used to be a center for generating the new populations in the next generation. Figure 1 illustrates the pattern of the fruit flies' search for food. Figure 1 (Initial phase), X_{axis}^0 is the fruit fly with the best sense of smell that is determined in current generation. In the latter generation, the X_{axis}^t is used to be the center for generating a new position of each individual fruit fly. The computational steps of FOA method are summarized in Algorithm 1.

Figure 1. The simulation state of the FOA's single leader.

Algorithm 1. The FOA algorithm

Step 1. Initialize FOA's parameters: random location(X_axis, Y_axis), size of population($popsiz$), and the maximum iteration(Max_iter).

Step 2. Give the random position and fly direction of an individual fruit fly in their search for food.

$$X_i = X_axis + rand()$$

$$Y_i = Y_axis + rand()$$

Step 3. Calculate the distance ($Dist$) to the food's origin, as the exact position of the food's location is not known at this stage.

$$Dist_i = \sqrt{x_i^2 + y_i^2}$$

Step 4. Calculate the smell concentration judgment value (S_i).

$$S_i = \frac{1}{Dist_i}$$

Step 5. Calculate the fitness: the smell concentration judgment of the individual fruit fly, obtained from Step 4, is calculated by substituting S_i into the smell concentration judgment function (also called *fitness function*), in order to find the optimal smell.

$$Smell_i = \text{objective function}(S_i)$$

Step 6. Determine the fruit fly with the optimal smell concentration judgment among the fruit fly group.

$$[bestSmell, bestIndex] = \text{find_the_best}(Smell)$$

Step 7. Keep the best (x, y) position and the optimal concentration value; and use this position as the center of flying towards the next location (in step 2).

$$Smell_{best} = bestSmell$$

$$X_axis = X(bestIndex)$$

$$Y_axis = Y(bestIndex)$$

Step 8. Repeat steps 2-7 and determine whether the smell concentration is better than the previous iterative smell concentration. If yes, go to step 7. The process will stop if either the smell concentration no longer changes or the iterative number reaches the maximum iteration number(Max_iter). The output are X_axis and Y_axis .

124

125 2.1. An Analysis of the Original FOA and the FOA Variants

126 2.1.1. Analysis of the Original FOA

127 There are problems of FOA that makes the algorithm unsuitable to deal with multi-dimensional
128 and complex optimization problems. The problems are investigated in [11,12,14,26,27,33,34] and are
129 briefly described as follows.

130 1. Problems regarding the smell value S_i , in according to Step 4, cannot appropriately evaluate
131 the "objective function (S_i)" when there are negative numbers in the domain because
132 $S_i = 1/Dist_i > 0$ so that the function cannot determine S_i as negative [11,14,33,34].

133 2. The fixing radius with random uniform distribution within the initial process limited the
134 convergence of FOA in procedure of the exploration and exploitation process [11,12,14,26,27].

2.1.2 Analysis of the FOA Variants

To overcome the disadvantages of the original FOA, researchers have developed continuously new strategies to improve the FOA for solving high-dimensional function optimization problems. The recently proposed FOAs can be found in the two categories. The first category is defining each of fruit flies through the random initial base point of X_axis, Y_axis and the positions of X_i and Y_i as in the original FOA for updating the new generation of populations. The other functions, such as smell value (S_i) and the evaluating of the objective function (S_i), are modified with including the extra mechanisms. The second category is trying to solve these the problems by (i) omitting Y_i , (ii) defining each of fruit flies through $X \in \mathbb{R}^{N \times D}$, where D is the number of decision variables (or dimensions), and N is the population size, i.e., $x_i = (x_i^1, x_i^2, \dots, x_i^D) \in \mathbb{R}^D$, and (iii) removing the distance $Dist_i$ and smell concentration judgment function S_i . The fitness value is now calculated by substituting x_i into the smell concentration judgment function with $Smell_i = \text{objective_function}(x_i)$. The position of the fruit fly having the minimal concentration value, X_axis , is the base point for flying towards the next location. A brief summary of the two categories improved FOAs are as follows.

The first category is as follow.

- Babalık et al. [35] proposed an improvement in fruit fly optimization algorithm by using sign parameters (SFOA). The algorithm presents the improvement method by using two sign variables r and q vectors in order to determine a sign for each decision variable of fruit flies.
- CEFOA [27] proposed by Han et al. revealed that the simple structure of FOA limited the search space and made the fruit flies to be trapped into a local minimum easily. To overcome the drawback, the CEFOA used two mechanisms; the trend search strategy and the co-evolution mechanism. The trend search enhances the local search capability of swarm. The co-evolution mechanism is employed to avoid the premature convergence and to improve the global searching ability. However, the key of the CEFOA is the multi-scales equation for updating fruit fly swarm. To set the variable capacity of each fruit fly connecting with its food quality, the CEFOA used the variance of the multi-level evolutionary operator. The search radius is dynamically adjusted through $X_i(t) = X_i(t-1) + LR \times N(0, \delta_i(t))$ and $Y_i(t) = Y_i(t-1) + LR \times N(0, \delta_i(t))$, where $\delta_i(t)$ are the multi-scale factor of i -th fruit fly.

The second category is as follow.

- LGMS-FOA, proposed by Shan et al. [11], presented two parameters to tune up the search radius by adding the weight parameter w when the radius is changed with respect to time. A new fruit fly location is generated as $x_i^d = X_axis_i^d + w \times \text{rand}[0, 1]$, $w = w_0 \times \alpha^t$, where $w_0 = 1$, $\alpha = 0.95$, t = iteration index, and X_axis is the best position obtained during iterations.
- IFFO, proposed by Pan et al. [12], introduced a new control parameter to adjust the search radius adaptively. The search radius is dynamically changed during iterations through $\lambda = \lambda_{max} \times \exp(\log(\lambda_{min}/\lambda_{max}) \times t/t_{max})$, where λ is the radius variants in each iteration, $\lambda_{max} = (UB - LB)/2$, UB is the upper bound and LB is the lower bound of domain problems, $\lambda_{min} = 10^{-5}$, t is the iteration index and t_{max} is the maximum iteration number (Max_iter).
- MFOA, proposed by Yuan et al. [14], presented a multi-swarm fruit fly that employed sub-swarm to explore the solutions in the search space simultaneously. Moreover, MFOA shrinks the search radius through $R(t) = (UB - LB/2) \times (G_{max} - G/G_{max})^\theta$, where t is the iteration index, UB is the upper bound, LB is the lower bound, G is the number of sub-swarms and G_{max} is the maximum number of sub-swarms.
- MSFOA, proposed by Zhang et al. [26], presented a strategy to analyze the convergence and showed that the convergence depends on the initial positions of swarms. MSFOA used the Gaussian mutation operator rather than the uniform random number (as can be seen the more

details in [26]). For a fruit fly flying, MSFOA used a linear generation mechanism, through the equation of $x_{i,j}^t = X_j^t + \omega \times \text{rand}(R_{min}, R_{max})$, where $\omega = \omega_0 \times \alpha^t$, $\omega_0 = 1$, $\alpha = 0.95$, t is current iteration, ω is the search coefficient, α is the initial weight and R_{min}, R_{max} are obtained from the domain boundary of the problem.

To recap, the improved FOAs such as SFOA, CEFOA, LGMS-FOA, IFFO, MFOA and MSFOA were proposed strategies to enhance the search ability. The search radius was a main tackle point in several proposed. However, only the dynamical mechanism of the search radius itself might not enough efficient for overwhelming the lacking of diversity and premature convergence because these FOA variants still used only one leader as the flying base point. The single leader strategy might affect to FOA in to be trapped in a local optimum easily when to optimize a multi-dimensional optimization problem. In this paper, the proposed CPFOA comprises of two strategies. The first strategy focuses on the enhancement of search ability based on the multileader fruit fly. The latter is a probabilistic dynamic search radius with adaptive normalization. These two mechanisms are different from the existing FOA variants. The details of proposed CPFOA are presented in the next section.

3. The Proposed CPFOA

This section presents a cooperation of multileader fruit flies and a probability search of random walk for FOA to solve the unconstrained optimization problems (CPFOA). The CPFOA consists of (i) the cooperation strategies called the multileader, and (ii) the probabilistic search by a random walk. The multileader strategy used a main leader and several other leaders as the flying bases. The probabilistic search by a random walk changes the search radius for controlling the search spaces of the main leader.

3.1. Multileader Strategy

Contrary to FOA that uses a single leader fruit fly, the CPFOA uses multileader fruit flies. Suppose that there is a swarm of fruit fly $X \in \mathbb{R}^{N \times D}$, where D is the number of decision variables (or dimensions), and N is the number of fruit flies, i.e., $x_i = (x_i^1, x_i^1, x_i^2, \dots, x_i^D) \in \mathbb{R}^D$. The multileader strategy generating has four computational steps as follow.

Step 1. Sort X in ascending order, based on the individual fitness values, to be

$$\dot{X} = \{\dot{X}_1, \dot{X}_2, \dots, \dot{X}_N\}. \quad (1)$$

Step 2. Divide $\dot{X}$ into M disjoint sub-swarms as Equation (2).

$$\dot{X} = \left\{ \dot{X}_1, \dot{X}_2, \dots, \dot{X}_{\frac{N}{M}} \right\} \cup \left\{ \dot{X}_{\frac{N}{M}+1}, \dot{X}_{\frac{N}{M}+2}, \dots, \dot{X}_{2\frac{N}{M}} \right\} \cup \dots \cup \left\{ \dot{X}_{\frac{(M-1)N}{M}+1}, \dot{X}_{\frac{(M-1)N}{M}+2}, \dots, \dot{X}_N \right\} \quad (2)$$

where M is the total number of sub-swarms (aka the number of leaders).

Step 3. Compute $leader_1, \dots, leader_M$ by Equation (3).

$$leader_{j,d} = \frac{M}{N} \sum_{l=1}^{\frac{N}{M}} \dot{X}_{\frac{(j-1)N}{M}+l,d}, j = 1, 2, \dots, \frac{N}{M}, d = 1, \dots, D \quad (3)$$

Step 4. Generate M new fruit flies based on $leader_1, \dots, leader_M$ by Equation (4).

$$M_C_j = \beta - X_{axis} \otimes leader_j \times \text{rand}[0,1), \quad j = 1, 3, \dots, \frac{N}{M} \quad (4)$$

where $\otimes$ is the Hadamard product.

The multileader strategy generates M new fruit flies from M leaders. These fruit flies are generated from the shared information and might result in the improvement of the search ability. Moreover, CPFOA controls the shrinking of search radius (β), in step 4, through the probability of

219 P_s . There are two types of radius; the normal scope (N_{scope}) and high scope (H_{scope}). The generation
220 of β is controlled by P_s as follow:

$$\beta = \begin{cases} N_{scope}, & \text{if } P_s > 0.1 \\ H_{scope}, & \text{otherwise} \end{cases} \quad (5)$$

$$P_s = t / \text{Max_iter} \quad (6)$$

221 where Max_iter is the maximum number of iteration, t is the iteration index and $0 \leq t \leq \text{Max_iter}$.

$$N_{scope} = \max(UB) + \min(UB) \quad (7)$$

222 where Ub = upper-bound, Lb = lower-bound of domain search problems.

$$H_{scope} = \max(X_{axis}) + \min(X_{axis}) \quad (8)$$

223

224 Base on Equation (6), the radius of fruit fly search is probabilistically large if P_s is a small value
225 (or t is at the early phase of the optimization). Otherwise, it is probabilistically a small value if the
226 iteration is in the latter phase of the optimization.

227 3.2. Probabilistic Search Strategy Based on Random Walk

228 As mentioned in the section 2.1.1, FOA uses the random uniform distribution with a fixed
229 radius of search. In this paper, CPFOA will employ a random walk generation which is inspired
230 from the ant lion optimizer [36], and a probabilistic control of the search radius. The random walk is
231 a process of mathematical equation which can provide the series of consecutive random steps
232 [37,38].

233 The value generated from a random walk at time $n > 0$ is done by a recursive formula as follow:

$$R^n = R^{n-1} + x^n \quad (9)$$

234 where x^n is a random value extracted from a random number generator, and $R^0 = 0$.

235 Equation (9) shows that the changing state of R^n is attached to previous state of R^{n-1} and
236 every step obtained from current iteration to the next iteration. The details of random walk in
237 CPFOA strategy can be described in this section.

238 1. The fruit fly with the best fitness (X_{axis}) is determined after the first generation of evaluation.
239 X_{axis} is used to be a center for update the N - M candidate solutions in the next generations.

240 2. For the updating step, CPFOA uses the random walk to generate N - M individual fruit flies. The
241 characteristics of the populations of random walk movements are described as:

$$XR^t = XR^{t+1} + \rho x^t \quad (10)$$

242 where ρ is a function that controls the direction of fruit fly in any changing step and $XR^0 = 0$.
243 In the proposed CPFOA, ρ is either -1 or 1 and is determined as

$$\rho = 2r^t - 1 \quad (11)$$

$$r^t = \begin{cases} 1, & \text{if } rand > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

244 where t denotes the iteration that came to a halt, $r(t)$ is a stochastic function and $rand$ is a
245 random uniform point in $[0, 1)$. In order to match the values generated from Eq. (10) with the
246 boundaries of the problem and, furthermore, to make the search process to transit from the
247 exploration phase to the exploitation phase smoothly, Eq. (10) is adaptively normalized as follow:

$$P_{XR_{i,d}^t} = \frac{(XR_{i,d}^t - a^t) \times (d^t - c^t)}{b^t - a^t} + c^t, i = 1, \dots, N, d = 1, \dots, D \quad (13)$$

248 where a is the minimum of $\{XR^0, \dots, XR^t\}$, b is the maximum of $\{XR^0, \dots, XR^t\}$, c and d is the
249 minimum and maximum radius at t -th iteration to control the scope of search space during in the

optimization steps, respectively. The value of c and d are determined through the changing values of L as follows:

$$c^t = \begin{cases} X_axis^t + LB^t, & \text{if } rand < 0.5 \\ X_axis^t - LB^t, & \text{otherwise} \end{cases}, \quad (14)$$

$$d^t = \begin{cases} X_axis^t + UB^t, & \text{if } rand < 0.5 \\ X_axis^t - UB^t, & \text{otherwise} \end{cases}, \quad (15)$$

where

$$LB^t = \frac{LB^t}{L} \quad (16)$$

$$UB^t = \frac{UB^t}{L} \quad (17)$$

L is a special constant parameter determined from the probability of Ps variable. The parameter of L and Ps can calculate as follow:

$$Ps = t/Max_iter \quad (18)$$

where t is the current iteration, Max_iter is the maximum number of iterations, and $L = Ps \times 10^2$ when $Ps > 0.25$, $L = Ps \times 10^3$ when $Ps > 0.5$, $L = Ps \times 10^4$ when $Ps > 0.75$, $L = Ps \times 10^5$, when $Ps > 0.8$, and $L = Ps \times 10^6$, when $Ps > 0.9$. In the proposed CPFOA, the constant L is used to adjust the accuracy level of exploitation.

The Equation (14) through Equation (18) performs the probabilistic control of the search radius for CPFOA, which is different from the mechanism in FOA variants. Figure 2 shows the simulation state of the cooperation of FOA's leader and CPFOA's co-leaders. Figure 3 (a) shows a graph of search radius generated during CPFOA is optimizing "Exponential function" ($f1$ in Table 1), and Figure 3 (b) shows the example graph of the search radius generated by IFFO. We have observed that the search radiuses in the two figures are very different. The behavior of the search radius in CPFOA is very similar to that of the chaotic gravitational constants in the gravitational search algorithm (GSA) [39]. This kind of behavior should help CPFOA in transiting from exploration stage to exploitation smoothly.

Figure 2. The simulation state of the cooperation of FOA’s leader and CPFOA’s co-leaders.

Figure 3. A graph of search radius generated from function Exponential, **(a)** is generated by CPFOA and **(b)** is the example graph of dynamic search radius generated by IFFO.

3.3. The Proposed CPFOA

The structure of CPFOA is similar to that of the IFFO in that the function $Dist_i$ and the smell concentration judgment value (S_i) are eliminated. The pseudo code of CPFOA is presented in Algorithm 2.

Algorithm 2. CPFOA

Input:

$popsiz$, Max_iter , Lb , Ub , dim , benchmark function ($Smell$)

Output: $X_axis(Max_iter)$, $bestSmell(Max_iter)$

- 1: Initial random position of an individual fruit fly (X_i)
- 2: Compute $S = \{Smell(X_i^0)\}, i = 1, \dots, N$
- 3: $bestIndex(0) = \text{ArgMin}(S, \{1, \dots, N\})$
- 4: $bestSmell(0) = S(0)_{bestIndex(0)}$
- 5: $X_axis(0) = X(0)_{bestIndex(0)}$
- 6: **for** $t = 1$ to Max_iter **do**
- 7: Generate $N-M$ fruit flies (called P_XR^t) based on X_axis using random walk by the Equation (13)
- 8: Sort X^t based on S
- 9: Generate M leaders by the Equation (3)
- 10: Generate M cooperative fruit flies based on M leaders (called $M_candidate^t$) by the Equation (4)
- 11: $X^t = M_candidate^t \cup P_XR^t$
- 12: Compute $S = \{Smell(X_i^t) | i = 1, \dots, N\}$
- 13: $bestIndex(t) = \text{ArgMin}(S, \{1, \dots, N\})$
- 14: $bestSmell(t) = S(t)_{bestIndex(t)}$
- 15: $X_axis(t) = X(t)_{bestIndex(t)}$
- 16: If $bestSmell(t) > bestSmell(t - 1)$
- 17: $bestSmell(t) = bestSmell(t - 1)$
- 18: $X_axis(t) = X_axis(t - 1)$
- 19: End If
- 20: **end for**

Note: $\text{ArgMin}(f, X)$ gives a position X_{min} at which f is minimized.

306 **Table 1.** The 18 standard benchmark functions (*f*1-*f*10 are the uni-modal, *f*11-*f*18 are the multi-modal functions).

No.	Function	x_i	$f(x)$
<i>f</i> 1	$f(x) = -exp(-0.5 \sum_{i=2}^n x_i^2)$	(-1, 1)	-1
<i>f</i> 2	$f(x) = \sum_{i=1}^n ix_i^4 + rand(0,1)$	(-1.28, 1.28)	0
<i>f</i> 3	$f(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	(-100, 100)	0
<i>f</i> 4	$f(x) = max(x_i)$	(-100, 100)	0
<i>f</i> 5	$f(x) = \sum_{i=1}^n x_i + \prod_{i=1}^n x_i $	(-10, 10)	0
<i>f</i> 6	$f(x) = \sum_{i=1}^n x_i^2$	(-100, 100)	0
<i>f</i> 7	$f(x) = \sum_{i=1}^n x_i $	(-1, 1)	0
<i>f</i> 8	$f(x) = \sum_{i=1}^n ix_i^2$	(-10, 10)	0
<i>f</i> 9	$f(x) = \sum_{i=1}^n x_i^2 - 450$	(-100, 100)	-450
<i>f</i> 10	$f(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2 - 450$	(-100, 100)	-450
<i>f</i> 11	$-exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=2}^n x_i^2}) - 20exp(\frac{1}{n} \sum_{i=2}^n cos(2\pi x_i^2))$	(-32, 32)	0
<i>f</i> 12	$f(x) = \sum_{i=1}^n x_i sin(x_i) + 0.1x_i $	(-10, 10)	0
<i>f</i> 13	$f(x) = \frac{1}{400} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n cos(\frac{x_i}{\sqrt{i}}) + 1$	(-600, 600)	0
<i>f</i> 14	$f(x) = -\sum_{i=1}^n exp(-\frac{x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1}}{8}) cos(4\sqrt{x_i^2 + x_{i+1}^2 + 0.5x_i x_{i+1}})$	(-5,5)	(1 - <i>n</i>)
<i>f</i> 15	$f(x) = \sum_{i=1}^n (x_i - 1)^2 - \sum_{i=1}^n x_i x_{i-1}$	(- <i>n</i> ² , <i>n</i> ²)	$\frac{n(n+4)(1-n)}{6}$
<i>f</i> 16	$f(x) = \sum_{i=1}^n (0.5 + \frac{sin^2(\sqrt{100x_i^2 + x_{i+1}^2 - 0.5})}{1 + 0.01(x_i^2 - 2x_i x_{i+1} + x_{i+1}^2)})$	(-100, 100)	0
<i>f</i> 17	$f(x) = \sum_{i=1}^n (x_i^2 - 10cos(2\pi x_i) + 10)$	(-5.12, 5.12)	0
<i>f</i> 18	$f(x) = 1 - cos(2\pi(\sqrt{\sum_{i=1}^n x_{i+1}^2} + 0.1\sqrt{\sum_{i=1}^n x_{i+1}^2}))$	(-100, 100)	0

316 **Table 2.** Summary of the CEC'17 benchmark test functions (search range = [-100, 100], dimension = 50 and 100).

Function	Type	Function Name	Optimum
<i>f</i> 1	Uni-modal	Shifted and Rotated Bent Cigar Function	100
<i>f</i> 2	Uni-modal	Shifted and Rotated Sum of Different Power Function	200
<i>f</i> 3	Uni-modal	Shifted and Rotated Zakharov Function	300
<i>f</i> 4	Simple multi-modal	Shifted and Rotated Rosenbrock's Function	400
<i>f</i> 5	Simple multi-modal	Shifted and Rotated Rastrigin's Function	500
<i>f</i> 6	Simple multi-modal	Shifted and Rotated Expanded Scaffer's F6 Function	600
<i>f</i> 7	Simple multi-modal	Shifted and Rotated Lunacek Bi_Rastrigin Function	700
<i>f</i> 8	Simple multi-modal	Shifted and Rotated Non-Continuous Rastrigin's Function	800
<i>f</i> 9	Simple multi-modal	Shifted and Rotated Levy Function	900
<i>f</i> 10	Simple multi-modal	Shifted and Rotated Schwefel's Function	1000
<i>f</i> 11	Hybrid Functions	Hybrid Function 1 (<i>N</i> =3)	1100
<i>f</i> 12	Hybrid Functions	Hybrid Function 2 (<i>N</i> =3)	1200
<i>f</i> 13	Hybrid Functions	Hybrid Function 3 (<i>N</i> =3)	1300
<i>f</i> 14	Hybrid Functions	Hybrid Function 4 (<i>N</i> =4)	1400
<i>f</i> 15	Hybrid Functions	Hybrid Function 5 (<i>N</i> =4)	1500
<i>f</i> 16	Hybrid Functions	Hybrid Function 6 (<i>N</i> =4)	1600
<i>f</i> 17	Hybrid Functions	Hybrid Function 6 (<i>N</i> =5)	1700
<i>f</i> 18	Hybrid Functions	Hybrid Function 6 (<i>N</i> =5)	1800
<i>f</i> 19	Hybrid Functions	Hybrid Function 6 (<i>N</i> =5)	1900
<i>f</i> 20	Hybrid Functions	Hybrid Function 6 (<i>N</i> =6)	2000
<i>f</i> 21	Composition Functions	Composition Function 1 (<i>N</i> =3)	2100
<i>f</i> 22	Composition Functions	Composition Function 2 (<i>N</i> =3)	2200
<i>f</i> 23	Composition Functions	Composition Function 3 (<i>N</i> =4)	2300
<i>f</i> 24	Composition Functions	Composition Function 4 (<i>N</i> =4)	2400
<i>f</i> 25	Composition Functions	Composition Function 5 (<i>N</i> =5)	2500
<i>f</i> 26	Composition Functions	Composition Function 6 (<i>N</i> =5)	2600
<i>f</i> 27	Composition Functions	Composition Function 7 (<i>N</i> =6)	2700
<i>f</i> 28	Composition Functions	Composition Function 8 (<i>N</i> =6)	2800
<i>f</i> 29	Composition Functions	Composition Function 9 (<i>N</i> =3)	2900
<i>f</i> 30	Composition Functions	Composition Function 10 (<i>N</i> =3)	3000

317

318

319

Table 3. The initial parameters of the FOA variants.

Algorithm	Strategy	Reference
FOA	A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example	Pan [40]
LGMS	LGMS-FOA: An Improved Fruit Fly Optimization Algorithm for Solving Optimization Problems	Shan [11]
IFFO	An improved fruit fly optimization algorithm for continuous function optimization problems	Pan [12]
MFOA	On a novel multi-swarm fruit fly optimization algorithm and its application	Yuan [14]
MSFOA	A novel multi-scale cooperative mutation fruit fly optimization algorithm	Zhang [26]
CEFOA	Novel fruit fly optimization algorithm with trend search and co-evolution	Han [27]

Table 4. The initial parameters of the six meta-heuristic algorithms.

Algorithm	Strategy	Reference
PSO	Particle swarm optimization	Kennedy [1]
DE	Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces	Storn [9]
GSA	GSA: A Gravitational Search Algorithm	Rashedi [41]
HS	A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice	Lee [42]
BA	A New Metaheuristic Bat-Inspired Algorithm	Yang [5]
FA	A comprehensive review of firefly algorithms	Fister [43]

Table 5. The initial parameters of six modified meta-heuristic algorithms.

Algorithm	Strategy	Reference
BLPSO	Biogeography-based learning particle swarm optimization	Chen [44]
CLPSO	Comprehensive learning particle swarm optimizer for global optimization of multimodal functions	Liang [45]
CoDE	Differential Evolution With Composite Trial Vector Generation Strategies and Control Parameters	Wang [46]
MGWO	A Modified Mean Gray Wolf Optimization	Singh [47]
JADE	JADE: Adaptive Differential Evolution With Optional External Archive	Zhang [48]
MFPA	A Modified Flower Pollination Algorithm for Global Optimization	Nabil [49]

4. The Experiments and Evaluations

There are three groups of competitive algorithms. The first group, shown in Table 3, is the FOA variants including FOA, LGMS, IFFO, MFOA, MSFOA, and CEFOA. The second group, shown in Table 4, comprise of CEFOA and six meta-heuristic algorithms; PSO, DE, GSA, HS, BA, and FA. The third group, shown in Table 5, comprise of CEFOA and six other meta-heuristic algorithms; BLPSO, CLPSO, CoDE, JADE, MGWO, and MFPA.

The performances of the proposed algorithm and that of the competitive algorithms are evaluated through two groups of benchmark functions; 18 scalable functions taken from [27] and 30 complex functions taken from CEC'2017 [50]. The definition of the functions in the first group and their global optima are listed in Table 1; f_1 - f_{10} are the uni-modal functions, and f_{11} - f_{18} are the multi-modal functions. Table 2 showed the details of CEC'2017 functions; f_1 - f_3 are the uni-modal functions, f_4 - f_{10} are the simple multi-modal functions, f_{11} - f_{20} are the hybrid functions and f_{21} - f_{30} are the composite functions, respectively. The CEC' 2017 benchmark functions are very challenging as they are harder than that from the first problems set.

The experimental environment is MATLAB 9.2.0 (R2017a) that ran on a personal computer with a 3.2 GHz CPU, 8 GB RAM. The operating system is Microsoft Windows 7 operating system (64-bit).

4.1 Parameters and Settings

There are three experiments, each of which is as follows.

1. The first experiment, the proposed CPFOA is compared with the competitive algorithms from the first group, i.e., CPFOA and the FOA variants are competing. The comparison is conducted on the 18 scalable functions taken from [27]. The dimension of each problem is set to three values; 30, 50, and 1000.
2. The second experiment, the proposed CPFOA is compared with the competitive algorithms from the second group, i.e., CPFOA and the original version of some state-of-the-art algorithms are competing. The comparison is based on the 18 scalable functions taken from [27]. The dimension of each problem is set as of the first experiment.
3. The third experiment, the proposed CPFOA is compared with the competitive algorithms from the third group, i.e., CPFOA and more recent challenging algorithms are competing. The comparison is conducted on 30 scalable functions taken from CEC'2017 [50]. The dimension of each problem is set to two values; 50 and 100.

The following settings are set to comply with that of the existing CEFOA; the maximum iteration (*Max_iter*) of each algorithm is fixed to 500, the population sizes (*popsiz*) is 30, and the average (*ave*) and the standard deviations (*std*) of the final objective values are computed from 50 replications. The other parameters of each competitive algorithm are set in accordance with their original literature, which are listed in Table 3, Table 4 and Table 5.

4.2 Performance Criteria

The criteria for performance evaluation of the competing algorithms are the quality, the robustness, the success rate, and the statistical test, where each of which is as follow.

1. The quality of algorithms is determined by the average value (*ave*) and standard deviation (*std*) of the final objective values. A lower value is better. Moreover, the decision can be supported by the convergence graph.

2. The robustness of algorithm is presented by the boxplot diagram. The narrow box plot represents a better distribution of final solution produced from algorithm.

3. The success rate (*SR*) determines by the number of successful runs over the total number of runs. A run is a successful run if the algorithm finds a feasible solution x , represented by $f(x) - f(x^*) \leq 1e - 10$, within the maximum number of function calls (the terminated condition of each run(Max_{NFE})), where x is a feasible optimal solution of the function, and x^* is the best known solution of a specific problem f . A higher average SR_{ave} indicates a better performance. The average success rate (SR_{ave}) is written as:

$$SR_{ave} = \frac{\text{number of successful runs}}{\text{total number of runs}} \quad (19)$$

4. Statistical test, to investigate the significant of difference between CPFOA's outcome and the competitive algorithm's outcome, the Wilcoxon signed rank test with the significance level of 0.05 is conducted to judge whether the 50 runs of CPFOA are statistically better than that of another competitor. The h values signifying the results of the Rank-sum test are indicated in Table 6 – 13 by one of three symbols; "+", "-", or "=". Where the "+" symbol means that the solutions produced by the competitive algorithm are better than, the "=" symbol means the outcomes of the competitive algorithm are comparable to or similar to, and the "-" symbol means that the outcomes of the competitive algorithm are worse than those of CPFOA. To conclude the statistical test, the total h is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively.

5. Results and Discussion

5.1 Convergence Behavior of CPFOA

This section will provide consistent information about the convergence behavior of CPFOA when the problem has an unknown number of local optima. The contours of the Egg holder function [51] and Schaffer function [52] are plotted in Figure 4 and Figure 5, respectively. In addition, the artificial fruit flies, appeared at several iterations, are also scattered in the different plots of the two functions. The optimal points of the two functions are located at near the top right corner, and at the origin, respectively. The starting points are far from the optimal points. We have observed from Figure 4 and Figure 5 that CPFOA has the capability of successfully optimizing the multimodal functions without trapping in local optima. The experiment has been repeated for 50 runs and every run reaches the optimal point. Now we can conclude that CPFOA has a balance between diversification (diversification) and intensification (exploitation), since it has not trapped in local optima.

Figure 4. Convergence behavior of CPFOA on Egg holder function.

404
405
406
407
408
409
410

Figure 5. Convergence behavior of CPFOA on Schaffer function.

411

412

413

414

415

416

417

418

419

420

421

422

423

5.2 Computational Time of CPFOA

Optimization problem should be solved in short time. Therefore, a good meta-heuristic optimization should have a short computational time. Figure 6 shows the average computational time consumed by eight algorithms when they optimized 18 of the scalable functions taken from [27]. From the figure, we found that CPFOA consumes quite short computational time compared to the other algorithms, especially when the dimension of problem is 1000. Therefore, CPFOA can be used effectively in optimizing a large scale problem.

Figure 6. Comparison of the average computational time from 50 runs based on 18 benchmark functions (500 iterations).

5.3 The First Experiment: Compare CPFOA to Six FOA Variants

The average (*ave*) and standard deviation (*std*) of the final objective values as well as *h* and *SR* values produced by seven algorithms performing 18 benchmark functions with the dimension of 30, 50 and 1000 are shown in Table 6, 7, and 8, respectively. The algorithms having the lowest *ave* or the lowest *std* in any case are highlighted in boldface. To conclude the table, the totals of *h* and average of *SR* values are shown in the last rows.

As can be seen from Table 6 and Table 7, CPFOA shows excellent performance with the total of *h* is of 11/6/1; for the uni-modal functions *f*₁-*f*₁₀, it has 5 wins, 4 ties, and 1 loss, for the multi-modal functions *f*₁₁-*f*₁₈, it has 6 wins, 2 ties, and 0 loss. Moreover, from the last row of Table 6 and 7, CPFOA has the highest the average *SR* value (0.94).

For the large scale problem, the results are shown in Table 8. CPFOA still shows a very good performance with the total of h is of 9/8/1; for the uni-modal functions f_1 - f_{10} , it has 3 wins, 5 ties, and 1 loss, for the multi-modal functions f_{11} - f_{18} , it has 6 wins, 3 ties, and 0 loss. In addition, from the last row of Table 8, CPFOA has the highest the average SR value (0.83).

To confirm the efficiency of the CPFOA, the convergence graphs of all seven algorithms when they are optimizing 18 benchmark test functions with dimension = 1000 are shown in Figure 7, where the red lines are the convergence graph of CPFOA. The graph is plotted in the log-log scale, where the x-axis is the number of iterations, and the y-axis is the average fitness values obtained at the corresponding iterations from the algorithms. From the graphs, CPFOA can reach the best solution faster than the other six FOA variants.

The box plots can reveal the distribution of final objective values produced by the algorithms. Figure 8 (a) is generated from f_4 , f_5 (dimension = 30) and Figure 8 (b) is generated from f_4 , f_5 (dimension = 50), Figure 8 (c) is generated from f_4 , f_5 (dimension = 1000). These box plots reveal that CPFOA is very robust to the increment of problem dimension as its final objective values distribution is very narrow when the problem dimension increases.

Table 6. Solution quality comparisons of the CPFOA and variant FOAs performing 18 benchmark functions (Dimension = 30). The *h* values signify the results of the Rank-sum test. The total *h* is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The *SR* is the success rate. For each function, the lowest of *ave* and *std* values, and the highest of *SR* value are highlighted with the boldfaces.

Function	Criteria	FOA	LGMS	IFFO	MFOA	MSFOA	CEFOA	CPFOA
f1	<i>ave</i>	-3.28E-3	4.56-E-2	1-	-1.86E1	-1	-1	-1
	<i>std</i>	1.65E-3	1.39E-2	1.85E-5	1.62E-9	1.62E-16	0.00	0.00
	<i>SR</i>	0	0	1	0	1	1	1
	<i>h</i>	-	-	-	-	-	=	NA
f2	<i>ave</i>	3.00E2	3.39E1	3.14E-3	3.36E-3	1.08E-1	1.02E-4	1.00E-3
	<i>std</i>	8.92E1	6.99E1	1.84E-3	1.90E-3	3.25E-2	1.98E-4	9.21E-4
	<i>SR</i>	0	0	0	0	0	0	0
	<i>h</i>	-	-	-	-	-	+	NA
f3	<i>ave</i>	2.42E5	1.43E3	2.32E1	3.15E-2	4.60E3	7.29E-14	0.00
	<i>std</i>	1.47E5	3.87E2	3.07E1	2.19E-2	3.48E3	1.98E-12	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	NA
f4	<i>ave</i>	9.55E1	1.01E1	1.48E-1	2.14E-3	1.64E1	1.98E-9	0.00
	<i>std</i>	4.51E1	8.10E-2	1.14E-1	1.11E-3	6.42E1	3.46E-8	0.00
	<i>SR</i>	0	0	0	0	0	0	1
	<i>h</i>	-	-	-	-	-	-	NA
f5	<i>ave</i>	1.34E3	1.29E1	1.21E1	4.38E-2	1.53E1	2.67E-9	0.00
	<i>std</i>	1.91E2	1.53E1	6.26E-1	2.64E-2	2.49E1	1.06E-6	0.00
	<i>SR</i>	0	0	0	0	0	0	1
	<i>h</i>	-	-	-	-	-	-	NA
f6	<i>ave</i>	9.82E4	7.88E1	4.70E-1	1.09E-4	1.13E-17	1.95E-12	0.00
	<i>std</i>	1.56E4	1.27E1	5.35E-1	1.22E-4	2.20E-18	2.08E-10	0.00
	<i>SR</i>	0	0	0	0	1	1	1
	<i>h</i>	-	-	-	-	-	-	NA
f7	<i>ave</i>	2.54	3.00E-27	3.59E-45	8.64E-40	0.00	1.45E-12	0.00
	<i>std</i>	5.68	1.64E-26	1.97E-44	1.93E-46	0.00	1.94E-11	0.00
	<i>SR</i>	0	0	1	0	1	1	1
	<i>h</i>	-	-	-	-	-	-	NA
f8	<i>ave</i>	2.98E3	9.09E1	1.08E-2	2.52E-6	3.45E1	0.00	0.00
	<i>std</i>	6.20E2	8.95E1	1.20E-2	2.38E-6	5.75E1	0.00	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	=	NA
f9	<i>ave</i>	9.03E4	-4.42E2	-4.49E2	-4.40E2	-4.50E2	-4.50E2	-4.50E2
	<i>std</i>	2.33E4	1.31E1	3.78E-1	1.78E-4	8.18E-14	0.00	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	=	NA

f10	ave	4.56E5	8.79E2	-4.48E2	-4.45E2	4.93E3	-4.50E2	-4.50E2
	std	3.96E5	4.90E2	4.37E1	2.89E-2	3.44E3	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f11	ave	2.05E1	3.46E1	1.80E-1	1.43	2.76E1	1.60E-10	8.88E-16
	std	1.84E-1	1.33E-1	1.34E-1	1.96	4.60E1	4.01E-10	0
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f12	ave	6.96E1	6.78E1	1.56E-2	1.21	7.64E1	9.15E-14	0.00
	std	1.20E1	6.26E-1	1.15E-2	2.70	2.92E1	1.55E-12	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f13	ave	8.51E2	1.01E1	2.27E-1	1.52E-4	1.09E-02	8.89E-16	0.00
	std	2.76E2	1.98E-2	2.57E-1	7.46E-5	1.29E-02	4.70E-15	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f14	ave	-3.17	1.80-E1	2.13-E1	-2.14E1	-8.54E1	-2.73E1	-2.70E1
	std	1.08	1.30E1	8.91E-1	2.82E-4	1.77E1	1.05	2.80E-2
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f15	ave	6.04E6	-5.18E2	5.30E1	3.06E1	3.19E4	-1.89E4	-3.12E3
	std	7.89E5	8.13E1	2.67E1	3.67E-1	2.82E4	3.60E3	5.79E1
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f16	ave	1.33E1	8.40E1	9.12E-1	9.51E-7	1.23E1	0.00	0.00
	std	4.26E-1	7.17E-1	2.71E-1	1.89E-6	3.84E-1	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f17	ave	4.19E2	4.19E2	1.17E-1	1.05E2	1.33E2	1.46E-8	0.00
	std	5.27E1	2.06E1	1.58E-1	1.31E2	3.90E1	4.42E-8	0.00
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f18	ave	3.72	6.60E-2	1.81E-4	9.99E-2	6.48E-14	0.00	0.00
	std	3.90	4.97E-2	2.55E-4	3.45E-7	7.76E-14	0.00	0.00
	SR	0	0	0	0	1	1	1
	h	-	-	-	-	-	=	NA
total	+	0	0	0	0	0	1	11
	-	18	18	18	18	18	11	1
h	=	0	0	0	0	0	6	6
Average SR		0.00	0.00	0.11	0.00	0.22	0.72	0.94

Table 7. Solution quality comparisons of the CPFOA and variant FOAs performing 18 benchmark functions (Dimension = 50). The *h* values signify the results of the Rank-sum test. The total *h* is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The *SR* is the success rate. For each function, the lowest of *ave* and *std* values, and the highest of *SR* value are highlighted with the boldfaces.

Function	Criteria	FOA	LGMS	IFFO	MFOA	MSFOA	CEFOA	CPFOA
<i>f</i> ¹	<i>ave</i>	-5.05E-4	4.17E-3	-1	-9.92E-1	-1	-1	-1
	<i>std</i>	1.21E-4	2.44E-3	3.03E-5	3.18E-9	2.96E-8	0.00	0.00
	<i>SR</i>	0	0	1	0	1	1	1
	<i>h</i>	-	-	-	-	-	=	NA
<i>f</i> ²	<i>ave</i>	7.69E2	1.29E2	4.99E-03	3.80E-3	5.38E-1	9.52E-5	1.03E-3
	<i>std</i>	1.09E2	1.76E1	3.39E-03	1.56E-3	1.88E-1	2.08E-4	8.76E-4
	<i>SR</i>	0	0	0	0	0	0	0
	<i>h</i>	-	-	-	-	-	+	NA
<i>f</i> ³	<i>ave</i>	1.22E6	6.49E3	4.12E1	7.52E-2	5.06E4	1.02E-13	0.00
	<i>std</i>	1.03E6	1.68E3	4.45E1	3.17E-2	1.89E4	2.10E-12	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	NA
<i>f</i> ⁴	<i>ave</i>	9.69E1	1.07E1	1.95E-1	1.26E-3	4.70E1	1.99E-9	0.00
	<i>std</i>	2.37	7.91E-2	1.49E-1	2.08E-4	7.62E1	3.50E-8	0.00
	<i>SR</i>	0	0	0	0	0	0	1
	<i>h</i>	-	-	-	-	-	-	NA
<i>f</i> ⁵	<i>ave</i>	2.42E3	2.23E1	1.71E1	8.01E-2	1.64E2	1.97E-29	0.00
	<i>std</i>	1.80E2	1.63E1	1.01E1	5.92E-2	6.57E1	1.01E-6	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	NA
<i>f</i> ⁶	<i>ave</i>	1.59E5	1.53E1	2.08E-1	1.32E-4	8.76E-5	1.02E-12	0.00
	<i>std</i>	1.73E4	1.90E	2.58E-1	1.14E-4	4.37E-4	1.88E-10	0.00
	<i>SR</i>	0	0	0	0	1	1	1
	<i>h</i>	-	-	-	-	-	-	NA
<i>f</i> ⁷	<i>ave</i>	2.57E-1	5.19E-53	2.91E-22	4.98E-44	0.00	1.95E-12	0.00
	<i>std</i>	4.33E-1	2.84E-52	2.00E21-	3.50E-43	0.00	1.48E-11	0.00
	<i>SR</i>	0	1	1	0	1	1	1
	<i>h</i>	-	-	-	-	-	-	NA
<i>f</i> ⁸	<i>ave</i>	9.81E3	2.82E2	2.57E-2	1.38E-5	6.24E1	0.00	0.00
	<i>std</i>	3.15E3	2.27E1	4.39E-2	1.09E-5	4.80E1	0.00	0.00
	<i>SR</i>	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	=	NA
<i>f</i> ⁹	<i>ave</i>	1.65E5	-4.46E2	-4.48E2	-4.45E2	-4.50E2	-4.50E2	-4.50E2
	<i>std</i>	1.90E4	5.82E-1	2.23E	9.61E-5	1.88E-4	0.00	0.00
	<i>SR</i>	0	0	0	0	1	1	1
	<i>h</i>	-	-	-	-	-	=	NA

f10	ave	3.51E6	-2.53E2	-2.60E1	-4.35E2	4.80E4	-4.50E2	-4.50E2
	std	2.99E6	3.50E2	4.23E2	1.45E-1	1.64E4	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f11	ave	2.06E1	3.53E1	1.19E-1	2.50E-3	1.15E1	1.69E-10	8.88E-16
	std	1.68E-1	1.12E-1	1.17E-1	6.19E-4	7.12E1	2.13E-10	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f12	ave	1.29E2	1.21E1	1.93E-2	1.57E-3	2.03E1	1.05E-15	0.00
	std	1.71E1	9.39E-1	1.83E-2	7.03E-4	5.40E1	1.95E-13	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f13	ave	1.37E3	1.04E1	2.01E-1	2.47E-4	3.23E-2	1.09E-17	0.00
	std	2.77E2	8.73E-3	2.05E-1	1.17E-4	1.55E-2	8.70E-14	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f14	ave	-6.00	2.83-E1	3.38E1	-3.62E1	-1.27E1	-2.58E1	-4.45E1
	std	1.80	1.41E1	1.15E1	3.77E-4	2.04E1	1.52	1.56
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f15	ave	8.73E7	1.03E3	3.90E8	5.47E1	9.22E5	-1.69E4	-8.60E3
	std	2.00E7	5.99E2	5.16E8	2.60	4.47E5	2.64E3	2.02E2
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f16	ave	2.30E1	1.65E1	9.40E-1	1.12E-6	2.18E1	0.00	0.00
	std	4.12E-1	9.15E-1	3.19E-1	1.38E-6	5.77E-1	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f17	ave	7.63E2	9.15E2	1.92E-1	4.01E-5	1.47E4	1.06E-8	0.00
	std	6.98E1	3.24E1	1.95E-1	3.32E-5	2.21E2	2.27E-8	0.00
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f18	ave	2.14	3.53E-2	1.60E-4	9.99E-2	6.47E-14	0.00	0.00
	std	3.72	4.68E-2	2.51E-4	2.09E-6	6.24E-14	0.00	0.00
	SR	0	0	0	0	1	1	1
	h	-	-	-	-	-	=	NA
total	+	0	0	0	0	0	1	11
	-	18	18	18	18	18	11	1
	=	0	0	0	0	0	6	6
Average SR		0.00	0.05	0.11	0.00	0.27	0.72	0.94

Table 8. Solution quality comparisons of the CPFOA and variant FOAs performing 18 benchmark functions (Dimension = 1000). The h values signify the results of the Rank-sum test. The total h is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The SR is the success rate. For each function, the lowest of ave and std values, and the highest of SR value are highlighted with the boldfaces.

Function	Criteria	FOA	LGMS	IFFO	MFOA	MSFOA	CEFOA	CPFOA
f_1	ave	-1.87E-35	7.20E-62	-1.00	-9.98E-6	-3.14E-54	-1	-1
	std	4.74E-35	8.73E-62	1.62E-4	2.07E-06	8.37E-54	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f_2	ave	4.01E5	7.36E4	2.21E-2	1.24E-2	1.60E5	0.00	2.53E-4
	std	1.85E4	2.29E3	1.28E-2	4.92E-3	5.16E3	0.00	2.55E-4
	SR	0	0	0	0	0	1	0
	h	-	-	-	-	-	+	NA
f_3	ave	1.24E9	7.01E7	4.96E2	5.33E3	3.26E7	0.00	0.00
	std	1.16E9	3.02E6	7.53E2	4.70E3	9.74E6	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f_4	ave	9.99E1	1.36E1	1.49E-1	4.71E-3	9.71E1	1.80E-9	0.00
	std	1.90E-1	1.47E-2	1.10E-1	1.91E-3	5.75E-1	1.06E-8	0.00
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f_5	ave	4.98E4	5.02E2	1.12E1	4.45	4.13E3	0.00	0.00
	std	8.52E2	1.01E1	8.06E1	3.26	6.01E1	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f_6	ave	3.35E6	3.67E2	2.34E1	1.98E-2	2.53E6	1.74E-12	0.00
	std	7.07E4	8.96E1	3.61E1	2.50E-2	4.64E4	3.28E-10	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f_7	ave	1.92	0.00	0.00	7.57E-1	0.00	1.15E-13	0.00
	std	1.11	0.00	0.00	1.88E-5	0.00	1.72E-11	0.00
	SR	0	1	1	0	1	1	1
	h	-	=	=	-	=	-	NA
f_8	ave	4.46E6	1.41E5	2.97E1	3.66E-2	3.06E6	0.00	0.00
	std	8.13E4	2.37E3	3.97E1	3.60E-2	7.01E4	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f_9	ave	3.35E6	-3.66E2	-4.48E2	-4.40E2	2.52E6	-4.50E2	-4.50E2
	std	7.25E4	3.21E1	2.23E1	2.05E-2	5.85E4	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA

f10	ave	1.57E9	6.34E5	4.23E2	4.66E3	7.77E6	-4.50E2	-4.50E2
	std	2.72E9	1.14E6	1.22E2	4.36E3	3.64E4	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f11	ave	2.11E1	3.81E1	5.96E-2	2.86	2.07E1	7.64E-11	8.88E-16
	std	2.71E-2	1.60E-2	5.02E-2	1.59	2.66E-2	2.00E-10	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f12	ave	2.93E3	3.06E2	2.00E-1	3.01E-2	2.18E3	1.15E-14	0.00
	std	4.12E1	3.75E1	1.58E-1	1.41E-2	5.35E1	1.51E-12	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f13	ave	2.96E4	1.81E1	1.76E-1	3.16E-3	2.26E4	2.09E-16	0.00
	std	8.47E2	1.98E-2	2.08E-1	1.40E-3	4.17E2	3.79E-15	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	NA
f14	ave	-2.31E1	4.52E2	-5.29E2	-7.37E2	-1.11E2	-2.32E1	-9.23E2
	std	3.91E1	8.21E1	6.83E	8.41E-3	1.29E1	1.05	6.45
	SR	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	NA
f15	ave	3.37E14	8.36E9	3.90E8	1.00E5	2.23E14	-1.29E4	-1.02E6
	std	1.29E13	3.83E8	5.16E8	3.39E4	6.18E12	2.54E3	1.64E3
	SR	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	NA
f16	ave	4.89E2	4.63E2	3.25E1	3.86E-4	4.78E2	0.00	0.00
	std	1.14E1	3.87E1	4.74E1	7.92E-4	4.52E1	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
f17	ave	1.79E4	1.02E4	1.39E1	7.96E2	1.47E4	1.14E-8	0.00
	std	3.77E2	1.15E2	1.25E1	4.45E2	1.61E2	3.27E-8	0.00
	SR	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	NA
f18	ave	3.93E1	4.05E-2	1.02E-4	9.99E-2	9.33E-14	0.00	0.00
	std	3.99E1	4.73E-2	9.30E-5	1.88E-5	9.48E-14	0.00	0.00
	SR	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	NA
total	+	0	0	0	0	0	1	9
	-	18	17	17	18	17	9	1
	=	0	1	1	0	1	8	8
Average SR		0.00	0.05	0.05	0.00	0.05	0.77	0.83

Figure 7. Convergence curves of 18 benchmark functions that compare CPFOA, CEFOA, FOA, LGMS, IFFO, MFOA and MSFOA. The dimension of problem is 1000.

504

(a) Dimension = 30

(b) Dimension = 50

(c) Dimension = 1000

Figure 8. Box plots of the optimal solutions which evaluated from standard benchmark function f_4 and f_5 (1=CPFOA, 2=CEFOA, 3=FOA, 4=LGMS, 5=IFFO, 6=MFOA, 7=MSFOA).

5.4 The Second Experiment: Comparison of the CPFOA and Meta-Heuristics Algorithms

This section presents a comparison of the CPFOA to the six meta-heuristics algorithms: PSO, DE, GSA, HS, BA, FA and including CEFOA. The average (*ave*) and standard deviation (*std*) of the final objective values as well as *h* and *SR* values produced by the competitive algorithms performing 18 benchmark functions with the dimension of 30, 50 and 1000 are shown in Table 9, 10, and 11, respectively. To conclude the table, the totals of *h* and average of *SR* values are shown in the last rows.

Table 9 and Table 10, CPFOA shows the best performance with the total of *h* is of 11/6/1; for the uni-modal functions f_1 - f_{10} , it has 5 wins, 4 ties, and 1 loss, for the multi-modal functions f_{11} - f_{18} , it has 6 wins, 2 ties, and 0 loss. Moreover, from the last row of Table 9 and 10 CPFOA has the highest the average *SR* value (0.94).

For the large scale problem set, the results are shown in Table 11. CPFOA still shows a very promising performance with the total of *h* is of 9/8/1; for the uni-modal functions f_1 - f_{10} , it has 3 wins, 6 ties, and 1 loss, for the multi-modal functions f_{11} - f_{18} , it has 6 wins, 2 ties, and 0 loss. In addition, from the last row of Table 11, CPFOA has the highest the average *SR* value (0.83).

Table 9. Solution quality comparisons of the CPFOA and the meta-heuristic algorithms performing 18 benchmark functions (Dimension = 30). The h values signify the results of the Rank-sum test. The total h is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The SR is the success rate. For each function, the lowest of ave and std values, and the highest of SR value are highlighted with the boldfaces.

Function	Criteria	PSO	DE	GSA	HS	BA	FA	CEFOA	CPFOA
f_1	ave	-0.99	-1	-1	-9.83E-1	-1	-9.99E-1	-1	-1
	std	3.75E-3	0.00	0.00	4.11E-3	1.51E-6	6.34E-4	0.00	0.00
	SR	0.03	0.73	1	0	1	0	1	1
	h	-	=	=	-	=	-	=	NA
f_2	ave	7.94E-2	2.12E-1	8.34E-2	1.99E-1	9.95E1	3.59E-2	1.02E-4	1.00E-3
	std	4.07E-2	8.95E-2	3.02E-2	5.29E-2	5.92E1	1.86E-2	1.98E-4	9.21E-4
	SR	0	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	+	NA
f_3	ave	8.28E2	6.89E2	9.49E2	2.11E4	5.29E4	1.16E3	7.29E-14	0.00
	std	1.05E3	4.97E2	3.11E2	5.78E3	2.27E4	4.47E2	1.98E-12	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f_4	ave	1.59E1	2.65E1	7.15E1	2.25E1	5.18E1	8.37E-2	1.98E-9	0.00
	std	7.79E1	6.47E1	2.41E1	2.34E	7.41E1	1.84E-2	3.46E-8	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f_5	ave	1.87E2	1.18E1	1.95E2	1.81E1	5.41E33	1.51E1	2.67E-9	0.00
	std	1.26E2	3.90E1	4.77E1	4.22E	2.27E34	2.13E1	1.06E-6	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f_6	ave	1.65E2	7.90E1	7.94E-3	3.21E2	2.12E4	5.71E-3	1.95E-12	0.00
	std	2.30E2	1.98E1	4.35E-2	8.92E1	5.47E3	2.43E-3	2.08E-10	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f_7	ave	0.00	0.00	0.00	0.00	0.00	0.00	1.45E-12	0.00
	std	0.00	0.00	0.00	0.00	0.00	0.00	1.94E-11	0.00
	SR	1	1	1	1	1	1	1	1
	h	=	=	=	=	=	=	-	NA
f_8	ave	3.20E1	5.23E-1	3.10E-15	8.23E1	2.51E2	3.89E-1	0.00	0.00
	std	5.52E1	1.79E1	1.90E-15	1.97E1	1.38E2	4.01E-1	0.00	0.00
	SR	0	0.03	1	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f_9	ave	-3.37E2	-4.37E2	-4.50E2	-1.40E2	1.93E4	-4.50E2	-4.50E2	-4.50E2
	std	1.14E2	3.08E1	3.17E-14	5.78E1	7.05E3	2.30E-3	0.00	0.00
	SR	0.3	0.96	0	0	0	0	1	1
	h	-	-	-	-	-	=	=	NA

f10	ave	3.81E2	2.46E2	5.14E2	2.00E4	6.53E4	6.84E2	-4.50E2	-4.50E2
	std	7.24E2	4.14E2	3.39E2	6.05E3	3.58E4	4.32E2	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f11	ave	4.41E1	2.47E	3.85E-2	5.15E1	1.62E1	4.11E-2	1.60E-10	8.88E-16
	std	1.53E1	2.24E	2.11E-1	5.67E-1	9.90E-1	1.46E-2	4.01E-10	0
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f12	ave	8.51E-1	7.52E-3	3.25E-3	9.89E-1	2.17E1	3.16E-2	9.15E-14	0.00
	std	7.59E-1	1.43E-2	2.88E-3	2.42E-1	5.31E1	2.14E-2	1.55E-12	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f13	ave	1.53E1	3.34E-1	2.75E1	3.81E1	2.01E2	3.59E-3	8.89E-16	0.00
	std	1.07E1	5.58E-1	6.78E1	6.52E-1	6.38E1	9.05E-4	4.70E-15	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f14	ave	-1.70E1	-1.35E1	-2.25E1	-2.63E1	-1.02E1	-2.32E1	-2.73E1	-2.75E1
	std	2.05E1	3.40E	1.86E1	5.13E-1	3.20E1	1.47E	1.05	2.80E-2
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f15	ave	6.35E4	9.45E13	8.22E5	5.21E4	1.63E6	-1.02E3	-1.89E4	-3.12E3
	std	5.76E4	9.75E13	1.19E5	1.92E4	5.80E5	2.25E3	3.60E3	5.79E1
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f16	ave	1.02E1	4.74E1	1.00E1	5.90E1	1.28E1	1.10E1	0.00	0.00
	std	7.62E-1	1.35E1	1.04E1	4.98E-1	4.30E-1	6.30E-1	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f17	ave	7.13E1	1.44E12	2.95E1	2.11E1	9.26E1	2.78E1	1.46E-8	0.00
	std	1.99E1	3.41E11	8.19E1	3.99E1	3.40E1	8.13E1	4.42E-8	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f18	ave	6.26E-54	0.00	3.72E-4	2.05E-4	1.73E-1	3.46E-7	0.00	0.00
	std	1.24E-53	0.00	8.54E-4	2.45E-4	1.34E-1	3.68E-7	0.00	0.00
	SR	1	1	0	0	0	1	1	1
	h	-	=	-	-	-	-	=	NA
total	+	0	0	0	0	0	0	1	11
	-	17	15	16	17	16	16	11	1
	h	=	1	3	2	1	2	2	6
Average SR		0.12	0.20	0.16	0.05	0.11	0.11	0.72	0.94

537
538
539

Table 10. Solution quality comparisons of the CPFOA and the meta-heuristic algorithms performing 18 benchmark functions (Dimension = 50). The h values signify the results of the Rank-sum test. The total h is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The SR is the success rate. For each function, the lowest of ave and std values, and the highest of SR value are highlighted with the boldfaces.

Function	Criteria	PSO	DE	GSA	HS	BA	FA	CEFOA	CPFOA
f^1	ave	-8.91E-1	-9.88E-1	-1	-8.13E-1	-1	-9.88E-1	-1	-1
	std	7.14E-2	1.46E-2	2.96E-6	2.74E-2	4.27E-6	3.79E-3	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f^2	ave	4.59E-1	4.13E-1	3.31E-1	1.82E1	2.21E1	6.48E-2	9.52E-5	1.03E-3
	std	2.99E-1	1.78E-1	1.10E-1	4.16E-1	9.78E1	3.35E-2	2.08E-4	8.76E-4
	SR	0	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	+	NA
f^3	ave	9.62E3	6.75E3	3.24E3	9.61E4	1.48E5	5.41E3	1.02E-13	0.00
	std	3.58E3	2.23E3	9.46E2	1.24E4	5.98E4	1.46E3	2.10E-12	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f^4	ave	3.67E1	5.99E1	1.23E1	4.36E1	5.79E1	1.13E1	1.99E-9	0.00
	std	7.64E	2.74E1	2.31E1	2.49E1	9.36E	6.65E-1	3.50E-8	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f^5	ave	7.84E2	5.14E1	3.90E2	1.03E2	3.82E56	8.28E1	1.97E-29	0.00
	std	2.12E2	6.07E1	3.43E1	1.63E1	1.63E57	5.11E1	1.01E-6	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f^6	ave	1.71E3	1.23E2	2.14E2	3.93E3	3.32E4	4.27E-2	1.02E-12	0.00
	std	1.08E3	1.14E2	1.44E2	5.43E2	7.92E3	1.11E-2	1.88E-10	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f^7	ave	0.00	0.00	0.00	0.00	0.00	0.00	1.95E-12	0.00
	std	0.00	0.00	0.00	0.00	0.00	0.00	1.48E-11	0.00
	SR	1	1	1	1	1	1	1	1
	h	=	=	=	=	=	=	-	NA
f^8	ave	1.47E2	1.57E1	1.29E1	2.11E2	1.13E3	2.99E1	0.00	0.00
	std	8.35E1	1.89E1	1.26E1	3.94E1	6.72E2	1.21E1	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f^9	ave	1.55E3	-3.05E2	-2.37E2	3.59E3	3.59E4	-4.50E2	-4.50E2	-4.50E2
	std	1.09E3	1.37E2	1.77E2	7.50E2	1.41E4	9.56E-3	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	=	=	NA

f10	ave	8.40E3	5.87E3	2.76E3	9.69E4	1.57E5	5.78E3	-4.50E2	-4.50E2
	std	2.70E3	1.64E3	9.91E2	1.32E4	8.08E4	1.45E3	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f11	ave	9.26E1	5.52E1	1.63E1	1.05E1	1.67E1	1.30E-1	1.69E-10	8.88E-16
	std	2.20E1	2.97E1	6.29E-1	5.63E-1	8.83E-1	2.52E-2	2.13E-10	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f12	ave	7.70E1	3.29E-1	2.09E-1	7.14E	4.20E1	2.05E-1	1.05E-15	0.00
	std	3.72E1	2.79E-1	3.37E-1	9.85E-1	7.84E1	1.18E-1	1.95E-13	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f13	ave	1.87E1	3.40E1	1.28E2	3.75E1	3.48E2	1.28E-2	1.09E-17	0.00
	std	8.63E1	3.85E1	1.43E1	5.20E1	1.13E2	3.12E-3	8.70E-14	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f14	ave	2.40E1	-1.43E1	-3.63E1	-4.09E1	-1.67E1	-3.79E1	-2.58E1	-4.45E1
	std	2.72E1	2.42E1	2.48E1	1.24E1	4.74E1	2.33E1	1.52	1.56
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f15	ave	1.95E6	2.99E5	4.62E7	2.89E6	2.01E7	1.54E4	-1.69E4	-8.60E3
	std	8.42E5	1.64E5	5.81E6	5.24E5	6.01E6	1.72E4	2.64E3	2.02E2
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f16	ave	1.98E1	9.93E1	1.71E1	1.38E1	2.23E1	1.97E1	0.00	0.00
	std	9.25E-1	1.07E1	1.46E1	6.27E-1	5.77E-1	6.80E-1	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f17	ave	1.72E2	3.07E2	6.12E1	8.20E1	1.36E2	5.70E1	1.06E-8	0.00
	std	3.34E1	7.58E1	1.15E1	8.57E1	4.96E1	9.44E1	2.27E-8	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f18	ave	7.22E-54	0.00	1.19E-3	2.86E-4	1.73E-1	3.47E-7	0.00	0.00
	std	2.23E-53	0.00	4.25E-3	4.91E-4	1.44E-1	3.65E-7	0.00	0.00
	SR	1	1	0	0	0	1	1	1
	h	-	=	-	-	-	-	=	NA
total	+	0	0	0	0	0	0	1	11
	-	17	16	17	17	17	16	11	1
	=	1	2	1	1	1	2	6	6
Average SR		0.11	0.11	0.05	0.05	0.05	0.11	0.72	0.94

545

546

Table 11. Solution quality comparisons of the CPFOA and the meta-heuristic algorithms performing 18 benchmark functions (Dimension = 1000). The *h* values signify the results of the Rank-sum test. The total *h* is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm, respectively. The *SR* is the success rate. For each function, the lowest of *ave* and *std* values, and the highest of *SR* value are highlighted with the boldfaces.

Function	Criteria	PSO	DE	GSA	HS	BA	FA	CEFOA	CPFOA
<i>f</i> ¹	<i>ave</i>	-1.36E-28	-1.01E-14	-4.42E-13	-7.28E-38	-3.33E-7	-6.34E-5	-1	-1
	<i>std</i>	7.05E-28	5.51E-14	1.09E-12	6.62E-38	1.44E-6	4.02E-5	0.00	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	=	NA
<i>f</i> ²	<i>ave</i>	5.96E4	2.43E5	5.38E3	9.42E4	6.24E3	1.35E3	0.00	2.53E-4
	<i>std</i>	6.61E3	6.28E3	5.64E2	2.48E3	3.44E3	1.63E2	0.00	2.55E-4
	<i>SR</i>	0	0	0	0	0	0	1	0
	<i>h</i>	-	-	-	-	-	-	+	NA
<i>f</i> ³	<i>ave</i>	8.10E6	1.44E7	6.35E6	4.53E7	7.27E7	3.43E6	0.00	0.00
	<i>std</i>	1.40E6	2.70E6	3.01E6	6.72E6	3.54E7	5.22E5	0.00	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	=	NA
<i>f</i> ⁴	<i>ave</i>	8.79E1	9.96E1	3.28E1	9.66E1	7.78E1	4.85E1	1.80E-9	0.00
	<i>std</i>	1.15E	1.52E-1	1.53E	1.70E-1	7.27E	1.69E	1.06E-8	0.00
	<i>SR</i>	0	0	0	0	0	0	0	1
	<i>h</i>	-	-	-	-	-	-	-	NA
<i>f</i> ⁵	<i>ave</i>	2.86E4	4.08E4	8.95E3	3.21E4	2.31E4	9.60E3	0.00	0.00
	<i>std</i>	7.51E2	5.12E2	2.28E2	3.69E2	3.05E3	2.82E2	0.00	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	=	NA
<i>f</i> ⁶	<i>ave</i>	1.39E6	2.90E6	1.25E5	1.73E6	7.91E5	1.50E5	1.74E-12	0.00
	<i>std</i>	5.28E4	7.53E5	4.92E3	2.62E4	1.86E5	1.21E4	3.28E-10	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	-	NA
<i>f</i> ⁷	<i>ave</i>	0.00	0.00	0.00	0.00	0.00	0.00	1.15E-13	0.00
	<i>std</i>	0.00	0.00	0.00	0.00	0.00	0.00	1.72E-11	0.00
	<i>SR</i>	1	1	1	1	1	1	1	1
	<i>h</i>	=	=	=	=	=	=	-	NA
<i>f</i> ⁸	<i>ave</i>	1.70E6	1.72E6	1.52E5	2.07E6	8.87E5	1.81E5	0.00	0.00
	<i>std</i>	7.82E4	1.47E6	1.12E4	3.91E4	2.61E5	1.31E4	0.00	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	=	NA
<i>f</i> ⁹	<i>ave</i>	1.36E6	2.81E6	1.23E5	1.71E6	8.18E5	1.51E5	-4.50E2	-4.50E2
	<i>std</i>	5.93E4	8.27E5	5.06E3	3.18E4	2.17E5	9.18E3	0.00	0.00
	<i>SR</i>	0	0	0	0	0	0	1	1
	<i>h</i>	-	-	-	-	-	-	=	NA

f10	ave	7.95E6	1.46E7	7.66E6	4.71E7	5.64E7	3.22E6	-4.50E2	-4.50E2
	std	1.19E6	3.66E6	3.73E6	5.54E6	3.04E7	5.05E5	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f11	ave	1.99E01	1.86E1	1.09E1	2.03E1	1.78E1	1.28E1	7.64E-11	8.88E-16
	std	9.78E-2	7.49E-1	1.69E-1	4.16E-2	7.21E-1	2.59E-1	2.00E-10	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f12	ave	1.59E3	9.62E2	4.57E2	1.74E3	1.34E3	5.44E2	1.15E-14	0.00
	std	7.39E1	4.90E1	2.46E1	2.56E1	1.50E2	2.32E1	1.51E-12	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f13	ave	1.26E4	2.61E4	2.05E4	1.55E4	7.71E3	1.35E3	2.09E-16	0.00
	std	8.42E2	6.83E3	2.83E2	2.82E2	2.22E3	7.89E1	3.79E-15	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	-	NA
f14	ave	-1.82E2	-8.67E1	-5.10E2	-1.11E2	-1.87E2	-4.84E2	-2.32E1	-9.23E2
	std	1.51E1	3.13E1	1.58E1	5.52E	4.06E1	1.35E1	1.05	6.45
	SR	0	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	-	NA
f15	ave	1.11E14	5.34E13	3.04E14	1.87E14	8.44E13	1.12E13	-1.29E4	-1.02E6
	std	7.06E12	5.40E12	6.30E12	3.98E12	2.31E13	8.49E11	2.54E3	1.64E3
	SR	0	0	0	0	0	0	0	0
	h	-	-	-	-	-	-	-	NA
f16	ave	4.79E2	2.79E2	4.82E2	4.84E2	4.70E2	4.48E2	0.00	0.00
	std	5.99E1	1.30E1	1.43E	1.49E	7.36E	3.35E	0.00	0.00
	SR	0	0	0	0	0	0	1	1
	h	-	-	-	-	-	-	=	NA
f17	ave	1.14E4	8.31E3	6.63E3	1.24E4	8.06E3	6.10E3	1.14E-8	0.00
	std	3.50E2	3.09E2	1.62E2	1.27E2	5.30E2	1.78E2	3.27E-8	0.00
	SR	0	0	0	0	0	0	0	1
	h	-	-	-	-	-	-	-	NA
f18	ave	5.09E-54	0.00	4.99E-3	1.97E-4	2.27E-1	2.88E-7	0.00	0.00
	std	2.44E-53	0.00	1.84E-2	2.35E-4	2.03E-1	3.19E-7	0.00	0.00
	SR	1	1	0	0	0	0	1	1
	h	-	=	-	-	-	-	=	NA
total	+	0	0	0	0	0	0	1	9
	-	17	16	17	17	17	17	9	1
	=	1	2	1	1	1	1	8	8
Average SR		0.11	0.11	0.05	0.05	0.05	0.05	0.77	0.83

5.5 The Third Experiment: CPFOA Optimizes the CEC'2017 Problems

The CEC'2017 problems are harder than the problem in the previous two sub-section. Our CPFOA, the recently proposed CEFOA, and 6 meta-heuristics including BLPSO, CLPSO, CoDE, JADE, MGWO, MFPA are evaluated based on this problem, with the dimension of 50 and 100. The winning algorithms are judged from average (*ave*) and standard deviation (*std*) of the final objective values, where lower is better. For an easy inspection, the *aves* of the row are sorted from low to high—rank first is the best, rank last is the worst. As can be seen from Table 12, the lowest of *aves*, *stds* and *Ranks* values are highlighted with the boldfaces.

The total *h* values are obtained from the p-values of 29 functions which indicate that all the p-values are less than 0.05. From Table 12 (dimension = 50), CPFOA ranks sixth, based on the total *h* values and the average of *Rank*. The first to fifth are CoDE, BLPSO, JADE, MFPA and MGWO respectively. From Table 13 (with dimension = 100), CPFOA get a better rank, it ranks forth, based on the total *h* values and the average of *Rank*. The first to third are BLPSO, CoDE and JADE, respectively. These results mean that CPFOA works efficiently when the number of dimension is high.

We conclude from the benchmarking results that CPFOA outperforms FOA variants and can compete with the other meta-heuristic algorithms in finding solutions of unconstrained optimization problems.

Table 12. Solution qualities comparisons of the CPFOA and seven meta-heuristic algorithms based on CEC’2017 benchmark functions (Dimension = 50). The *h* values signify the results of the Rank-sum test. The total *h* is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm when it is competed with CPFOA, respectively. The *ave* values of the row are sorted from low to high—rank first is the best, rank last is the worst. For each function, the lowest of *Rank*, *ave* and *std* values are highlighted with the boldfaces.

Function	Criteria	BLPSO	CLPSO	CoDE	JADE	MGWO	MFPA	CEFOA	CPFOA
f1	ave	3.46E3	3.35E9	4.87E3	5.16E3	1.15E10	1.12E9	1.36E11	3.00E3
	std	3.31E3	5.62E8	6.67E3	5.68E3	4.29E9	1.16E9	8.56E7	2.92E3
	h	-	-	-	-	-	-	-	NA
	Rank	2	6	3	4	7	5	8	1
f2	Excluded from the experiment								
f3	ave	1.83E5	2.70E5	6.42E3	1.45E5	1.28E5	1.31E5	1.86E14	1.55E5
	std	3.11E4	4.20E4	2.23E3	6.64E4	2.35E4	4.39E4	5.81E12	4.62E4
	h	-	-	+	+	+	+	-	NA
	Rank	6	7	1	4	2	3	8	5
f4	ave	5.90E2	1.68E3	5.26E2	5.46E2	1.52E3	9.03E2	5.73E4	5.82E2
	std	3.97E1	2.74E2	4.72E1	4.73E1	5.58E2	3.61E2	2.88E1	4.38E1
	h	-	-	+	+	+	-	-	NA
	Rank	4	7	1	2	6	5	8	3
f5	ave	7.40E2	9.20E2	6.04E2	7.40E2	7.46E2	8.38E2	1.37E3	8.27E2
	std	1.86E1	2.58E1	4.18E1	2.23E1	3.57E1	5.38E1	1.24E2	3.84E1
	h	+	-	+	+	+	-	-	NA
	Rank	2	7	1	3	4	6	8	5
f6	ave	6.00E2	6.23E2	6.00E2	6.01E2	6.73E2	6.52E2	7.48E2	6.62E2
	std	1.93E-2	4.15E	7.36E4	6.53E-1	5.25E	8.32E	6.92E-1	4.92E
	h	+	+	+	+	-	+	-	NA
	Rank	1	4	2	3	7	5	8	6
f7	ave	1.07E3	1.26E3	9.47E2	1.03E3	1.33E3	1.75E3	2.21E3	1.19E3
	std	1.77E1	3.11E1	6.31E1	3.15E1	6.04E1	1.97E2	2.83E	1.19E2
	h	+	-	+	+	-	-	-	NA
	Rank	3	5	1	2	6	7	8	4
f8	ave	1.04E3	1.22E3	9.20E2	1.04E3	1.45E3	1.17E3	1.71E3	1.15E3
	std	1.77E1	2.82E1	5.79E1	2.23E1	3.94E1	4.66E1	1.85E	4.26E1
	h	+	-	+	+	-	-	-	NA
	Rank	2	6	1	3	7	5	8	4
f9	ave	9.15E2	2.03E4	9.07E2	1.52E3	9.24E3	2.08E4	7.84E4	1.31E4
	std	1.03E1	4.13E3	1.14E1	7.02E2	4.03E3	2.06E3	1.89E3	9.08E2
	h	+	-	+	+	+	-	-	NA
	Rank	2	6	1	3	4	7	8	5

f10	ave	1.04E4	1.04E4	1.02E4	1.05E4	8.54E3	1.04E4	2.18E4	7.62E3
	std	4.66E2	5.09E2	1.97E3	6.26E2	2.57E3	1.15E3	7.80E1	1.03E3
	h	-	-	-	-	-	-	-	NA
	Rank	4	5	3	7	2	6	8	1
f11	ave	1.39E3	9.74E3	1.16E3	1.68E3	6.25E3	2.82E3	1.83E6	1.36E3
	std	4.72E1	2.22E3	1.75E1	9.98E2	3.04E3	1.32E3	2.81E5	6.03E1
	h	+	-	+	-	-	-	-	NA
	Rank	3	7	1	4	6	5	8	2
f12	ave	9.44E6	1.61E9	4.93E5	1.46E6	8.66E8	7.21E7	1.43E11	3.77E7
	std	2.82E6	4.19E8	2.98E5	1.25E6	8.01E8	1.22E8	1.41E8	2.85E7
	h	+	-	+	+	-	-	-	NA
	Rank	3	7	1	2	6	5	8	4
f13	ave	3.24E3	3.64E8	6.12E3	8.11E3	1.60E8	1.86E5	1.14E11	1.75E5
	std	2.18E3	1.53E8	5.02E3	6.07E3	1.55E8	4.46E5	2.56E8	1.33E5
	h	+	-	+	+	-	-	-	NA
	Rank	1	2	3	4	7	6	8	5
f14	ave	4.49E5	6.90E6	1.50E3	1.85E5	1.31E6	1.90E5	1.47E9	1.75E5
	std	3.99E5	3.87E6	9.04E1	4.26E5	1.50E6	1.44E5	1.71E6	1.05E5
	h	-	-	+	-	-	-	-	NA
	Rank	5	7	1	3	6	4	8	2
f15	ave	7.29E3	5.01E7	2.05E3	7.15E3	1.25E7	1.96E4	2.39E10	3.88E4
	std	6.37E3	3.62E7	5.75E2	6.22E3	2.70E7	1.66E4	1.77E7	3.31E4
	h	+	-	+	+	-	+	-	NA
	Rank	3	7	1	2	6	4	8	5
f16	Ave	2.97E3	4.44E3	2.57E3	3.55E3	3.85E3	3.76E3	2.47E4	3.70E3
	Std	2.34E2	3.02E2	3.36E2	2.73E2	5.01E2	5.01E2	2.84E1	4.94E2
	h	+	-	+	+	-	-	-	NA
	Rank	2	7	1	3	6	5	8	4
f17	ave	2.78E3	3.91E3	2.43E3	3.20E3	2.89E3	3.39E3	1.78E5	3.26E3
	std	1.76E2	3.42E2	1.75E2	1.81E2	2.80E2	4.54E2	1.27E3	2.71E2
	h	+	-	+	+	+	-	-	NA
	Rank	2	7	1	4	3	6	8	5
f18	ave	2.72E6	2.37E7	2.74E4	2.05E5	9.73E6	2.20E6	2.12E9	1.81E6
	std	9.79E5	1.50E7	2.16E4	6.14E5	1.33E7	2.14E6	1.30E7	1.46E6
	h	-	-	+	+	-	-	-	NA
	Rank	5	7	1	2	6	4	8	3
f19	ave	1.65E4	7.51E6	2.03E3	6.93E3	8.80E6	2.39E4	1.40E10	1.69E6
	std	8.24E3	6.60E6	1.56E2	9.09E3	1.39E7	3.14E4	9.06E6	1.21E6
	h	+	-	+	+	-	+	-	NA
	Rank	3	6	1	2	7	4	8	5

f20	ave	2.76E3	3.44E3	2.59E3	3.36E3	2.97E3	3.43E3	5.45E3	3.12E3
	std	1.79E2	2.17E2	2.26E2	2.37E2	3.15E2	3.43E2	2.99E1	3.22E2
	h	+	-	+	-	+	-	-	NA
	Rank	2	7	1	5	3	6	8	4
f21	ave	2.55E3	2.76E3	2.40E3	2.53E3	2.74E3	2.67E3	4.35E3	2.61E3
	std	1.93E1	2.53E1	3.96E1	2.23E1	4.05E1	7.08E1	2.75E	8.44E1
	h	+	-	+	+	-	-	-	NA
	Rank	3	7	1	2	6	5	8	4
f22	ave	8.91E3	1.22E4	1.02E4	1.24E4	9.76E3	1.16E4	2.12E4	9.54E3
	std	4.44E3	9.66E2	2.95E3	7.48E2	1.23E3	1.42E3	7.10E1	1.66E3
	h	+	-	-	-	-	-	-	NA
	Rank	1	6	4	7	3	5	8	2
f23	ave	2.99E3	3.25E3	2.81E3	2.98E3	3.60E3	3.23E3	9.67E3	3.14E3
	std	2.23E1	3.35E1	2.81E1	2.53E1	8.74E1	1.42E2	2.85E1	6.88E1
	h	+	-	+	+	-	-	-	NA
	Rank	3	6	1	2	7	5	8	4
f24	ave	3.16E3	3.44E3	2.97E3	3.13E3	3.39E3	3.34E3	6.85E3	3.26E3
	std	2.32E1	3.24E1	1.90E1	3.49E1	1.00E2	1.72E2	2.11E	6.37E1
	h	+	-	+	+	-	-	-	NA
	Rank	3	7	1	2	6	5	8	4
f25	ave	3.07E3	3.99E3	3.03E3	3.06E3	3.75E3	3.30E3	2.00E4	3.07E3
	std	2.58E1	1.59E2	3.35E1	2.55E1	2.67E2	1.50E2	1.63E1	2.99E1
	h	+	-	+	-	-	-	-	NA
	Rank	2	7	1	3	6	5	8	4
f26	ave	6.22E3	8.81E3	4.61E3	6.20E3	7.04E3	9.65E3	2.03E4	3.54E3
	std	2.30E2	4.13E2	2.45E2	2.81E2	7.07E2	1.33E3	8.30E	1.84E2
	h	-	-	+	-	-	-	-	NA
	Rank	3	6	2	5	4	7	8	1
f27	ave	3.47E3	4.06E3	3.24E3	3.35E3	3.71E3	3.68E3	1.93E4	3.68E3
	std	3.75E1	9.16E1	1.75E1	5.89E1	1.17E2	1.63E2	2.36E1	1.53E2
	h	+	-	+	+	-	=	-	NA
	Rank	3	7	1	2	5	4	8	6
f28	ave	3.34E3	5.66E3	3.29E3	3.32E3	4.80E3	4.35E3	2.03E4	3.33E3
	std	2.95E1	2.38E2	2.26E1	2.47E1	6.26E2	1.54E3	1.33E1	3.32E1
	h	-	-	+	+	-	-	-	NA
	Rank	4	7	1	2	6	5	8	3
f29	ave	4.05E3	6.14E3	3.52E3	4.30E3	4.83E3	5.31E3	6.75E6	5.97E3
	std	1.65E2	4.24E2	1.98E2	1.79E2	3.34E2	6.06E2	4.27E4	5.71E2
	h	+	-	+	+	+	+	-	NA
	Rank	2	7	1	3	4	5	8	6

<i>f</i> 30	<i>ave</i>	5.72E6	2.52E8	7.40E5	1.17E6	1.48E8	6.34E6	2.50E10	6.99E7
	<i>std</i>	2.07E6	8.30E7	9.01E4	3.62E5	1.26E8	7.35E6	3.98E7	1.92E7
	<i>h</i>	+	-	+	+	-	+	-	NA
	<i>Rank</i>	3	7	1	2	6	4	8	5
<i>total</i> <i>h</i>	+	21	1	26	21	7	6	0	
	-	8	28	2	8	21	22	29	
	=	0	0	0	0	0	1	0	
	<i>Average Rank</i>	2.82	6.34	1.37	3.13	5.31	5.10	8	5.72

602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634

Table 13. Solution qualities comparisons of the CPFOA and seven meta-heuristic algorithms based on CEC'2017 benchmark functions (Dimension = 100). The *h* values signify the results of the Rank-sum test. The total *h* is represented as #1/#2/#3, where #1, #2, and #3 represents the number of wins, ties, and loss for the algorithm when it is competed with CPFOA, respectively. The *ave* values of the row are sorted from low to high—rank first is the best, rank last is the worst. For each function, the lowest of *Rank*, *ave* and *std* values are highlighted with the boldfaces.

Function	Criteria	BLPSO	CLPSO	CoDE	JADE	MGWO	MFPA	CEFOA	CPFOA
f1	ave	5.89E3	4.48E10	1.38E4	2.15E7	5.48E10	3.58E10	2.98E11	1.84E5
	std	4.31E3	5.46E9	7.99E3	2.77E7	8.03E9	9.69E09	1.35E8	1.10E5
	h	+	-	+	-	-	-	-	NA
	Rank	1	6	2	4	7	5	8	3
f2	Excluded from the experiment								
f3	ave	7.60E4	6.73E5	1.18E5	4.74E5	4.28E5	4.09E5	1.52E14	3.39E5
	std	1.80E4	8.21E4	1.85E4	1.08E5	6.87E4	8.30E4	2.63E12	2.52E4
	h	+	-	+	-	-	-	-	NA
	Rank	1	6	2	5	4	7	8	3
f4	ave	5.20E2	9.40E3	6.83E2	8.10E2	5.88E3	4.56E3	1.60E5	8.07E2
	std	1.24E1	1.24E3	4.18E1	6.42E1	1.66E3	2.92E3	1.42E2	7.97E1
	h	+	-	-	-	-	-	-	NA
	Rank	1	7	2	4	6	5	8	3
f5	ave	7.08E2	1.72E3	7.80E2	1.28E3	1.21E3	1.50E3	2.38E3	1.28E3
	std	1.40E1	4.47E1	7.76E1	5.85E1	6.49E1	1.08E2	1.13E1	6.71E1
	h	+	-	-	=	+	-	-	NA
	Rank	1	7	2	4	3	6	8	5
f6	ave	6.00E2	6.57E2	6.00E2	6.12E2	6.44E2	6.67E2	7.40E2	6.66E2
	std	5.28E-2	5.47E	3.33E2	2.49E	4.17E	5.92E	3.54E-1	3.07E
	h	+	+	+	+	+	-	-	NA
	Rank	1	5	2	3	4	7	8	6
f7	ave	1.42E3	2.50E3	1.41E3	1.86E3	2.13E3	3.92E3	4.36E3	3.22E3
	std	8.49E1	1.18E2	1.99E2	9.85E1	1.75E2	3.91E2	1.96E1	1.80E2
	h	+	+	+	+	+	-	-	NA
	Rank	2	5	1	3	4	7	8	6
f8	ave	9.58E2	2.03E3	1.09E3	1.57E3	1.51E3	1.84E3	2.84E3	1.69E3
	std	1.90E1	4.86E1	1.17E2	4.71E1	8.61E1	1.04E2	5.63E	8.06E1
	h	+	-	+	+	+	-	-	NA
	Rank	1	7	2	4	3	6	8	5
f9	ave	6.66E3	8.58E4	4.80E3	1.72E4	4.16E4	3.14E4	1.16E5	2.59E4
	std	2.95E2	9.69E3	2.15E3	7.96E3	1.27E4	4.81E3	1.23E3	7.19E2
	h	+	-	+	+	-	-	-	NA
	Rank	2	7	1	3	6	5	8	4

f10	ave	5.15E3	2.48E4	2.46E4	2.64E4	1.79E4	2.25E4	3.67E4	1.53E4
	std	3.99E2	8.11E2	5.31E3	2.00E3	3.95E3	2.42E3	2.83E1	1.92E3
	h	+	-	-	-	-	-	-	NA
	Rank	1	6	5	7	3	4	8	2
f11	ave	8.02E3	2.01E5	3.01E3	4.94E4	7.66E4	6.97E4	2.69E13	1.61E4
	std	5.25E3	3.75E4	5.79E2	3.48E4	1.56E4	2.98E4	2.13E11	5.71E3
	h	+	-	+	-	-	-	-	NA
	Rank	2	7	1	4	6	5	8	3
f12	ave	5.83E8	1.65E10	6.73E6	2.76E7	1.19E10	4.06E9	2.61E11	2.58E8
	std	1.11E6	3.32E9	2.24E6	1.63E7	5.61E9	5.07E9	1.48E8	8.72E7
	h	-	-	+	+	-	-	-	NA
	Rank	3	6	1	7	5	4	8	2
f13	ave	1.10E9	1.34E9	5.63E3	7.24E3	1.24E9	4.17E7	6.57E10	4.58E4
	std	3.20E2	2.65E8	3.58E3	4.23E3	9.03E8	1.26E8	3.14E7	1.10E4
	h	-	-	+	+	-	-	-	NA
	Rank	5	7	1	2	6	4	8	3
f14	ave	8.84E4	4.97E7	1.25E5	3.94E5	8.71E6	4.66E6	1.48E9	1.83E6
	std	1.21E5	2.45E7	8.51E4	3.14E5	5.61E6	3.76E6	3.81E6	1.67E6
	h	+	-	+	+	-	-	-	NA
	Rank	1	7	2	3	6	5	8	4
f15	ave	8.62E3	2.98E8	3.89E3	6.12E3	2.59E8	2.44E6	4.14E10	5.04E4
	std	4.74E3	1.12E8	3.13E3	5.00E3	2.74E8	4.59E6	1.84E7	2.03E4
	h	+	-	+	+	-	-	-	NA
	Rank	2	7	1	3	6	5	8	4
f16	ave	2.51E3	1.16E4	4.88E3	7.52E3	6.53E3	6.96E3	3.59E4	6.88E3
	std	2.37E2	7.62E2	5.58E2	4.67E2	7.98E2	9.67E2	2.60E1	7.77E2
	h	+	-	+	-	+	-	-	NA
	Rank	1	7	2	6	3	5	8	4
f17	ave	1.91E3	1.03E4	4.18E3	5.87E3	5.22E3	6.25E3	1.81E8	5.41E3
	std	7.38E1	1.35E3	3.79E2	3.93E2	1.05E3	6.67E2	4.13E5	9.63E2
	h	+	-	+	-	+	-	-	NA
	Rank	1	7	2	5	3	6	8	4
f18	ave	4.62E5	6.87E7	5.44E5	9.65E5	7.13E6	6.68E6	1.50E9	2.67E6
	std	2.67E5	2.38E7	2.53E5	1.04E6	4.03E6	3.99E6	3.13E6	1.09E6
	h	+	-	+	+	-	-	-	NA
	Rank	1	7	2	3	6	5	8	4
f19	ave	1.17E4	2.96E8	4.31E3	5.13E3	2.29E8	5.54E6	4.18E10	7.23E6
	std	2.16E4	1.21E8	2.56E3	3.01E3	2.75E8	1.57E7	2.63E8	4.07E6
	h	+	-	+	+	-	+	-	NA
	Rank	3	7	1	2	6	4	8	5

f20	ave	2.29E3	6.63E3	4.13E3	6.45E3	5.56E3	5.70E3	3.95E4	5.21E3
	std	6.04E1	3.98E2	4.35E2	4.47E2	1.16E3	6.24E2	2.60E1	6.24E2
	h	+	-	+	-	-	-	-	NA
	Rank	1	7	2	6	4	5	8	3
f21	ave	2.42E3	3.61E3	2.60E3	3.11E3	3.06E3	3.62E3	1.11E4	3.42E3
	std	1.66E1	5.60E1	1.32E2	6.28E1	1.14E2	1.76E2	1.71E1	9.76E1
	h	+	-	+	+	+	-	-	NA
	Rank	1	6	2	4	3	7	8	5
f22	ave	2.50E3	2.77E4	2.74E4	2.84E4	2.10E4	2.50E4	4.08E4	1.83E4
	std	1.10E3	7.58E2	4.25E3	1.57E3	3.05E3	2.53E3	2.55E1	1.96E3
	h	+	-	-	-	-	-	-	NA
	Rank	1	6	5	7	3	4	8	2
f23	ave	2.78E3	4.24E3	3.38E3	3.54E3	3.64E3	4.04E3	1.64E4	3.85E3
	std	1.70E1	6.88E1	1.37E2	6.23E1	9.47E1	1.75E2	6.78E1	1.02E2
	h	+	-	+	+	+	-	-	NA
	Rank	1	7	2	3	4	6	8	5
f24	ave	2.95E3	5.13E3	3.53E3	4.09E3	4.39E3	4.91E3	1.68E4	4.80E3
	std	1.71E1	1.40E2	6.10E1	7.70E1	1.69E2	2.68E2	9.04E	2.17E2
	h	+	-	+	+	+	-	-	NA
	Rank	1	7	2	3	4	6	8	5
f25	ave	2.89E3	9.67E3	3.34E3	3.51E3	6.87E3	6.55E3	3.59E4	3.47E3
	std	6.65E	8.41E2	5.07E1	6.74E1	9.11E2	1.35E3	2.37E1	7.69E1
	h	+	-	+	-	-	-	-	NA
	Rank	1	7	2	4	6	5	8	3
f26	ave	4.75E3	2.25E4	8.22E3	1.40E4	1.68E4	2.32E4	6.64E4	1.72E4
	std	3.89E2	1.14E3	5.22E2	7.31E2	1.49E3	3.27E3	1.38E1	5.58E3
	h	+	-	+	+	+	-	-	NA
	Rank	1	6	2	3	4	7	8	5
f27	ave	3.25E3	5.62E3	3.39E3	3.61E3	4.20E3	4.27E3	2.57E4	3.98E3
	std	6.17E	2.53E2	2.95E1	8.49E1	1.85E2	2.14E2	4.52E1	3.08E2
	h	+	+	+	+	-	-	-	NA
	Rank	1	4	2	3	6	7	8	5
f28	ave	3.24E3	1.63E4	3.48E3	3.64E3	9.33E3	1.01E4	4.36E4	3.53E3
	std	1.42E1	1.15E3	3.65E1	7.99E1	1.29E3	3.29E3	4.56E	4.81E1
	h	+	-	+	-	-	-	-	NA
	Rank	1	7	2	4	5	6	8	3
f29	ave	3.71E3	1.51E4	5.71E3	8.02E3	9.07E3	8.77E3	8.91E6	9.10E3
	std	1.10E2	1.31E3	4.75E2	4.43E2	9.20E2	8.70E2	3.04E4	8.89E2
	h	+	-	+	+	+	+	-	NA
	Rank	1	7	2	3	4	5	8	6

f30	ave	1.49E5	1.08E9	1.07E4	6.65E4	1.21E9	5.20E7	6.12E10	5.82E7
	std	1.13E5	2.53E8	2.36E3	3.81E4	1.22E9	6.24E7	2.46E7	3.18E7
	h	+	-	+	+	-	+	-	NA
	Rank	3	6	1	2	7	4	8	5
total	+	27	3	25	17	11	3	0	
	-	2	26	4	11	18	26	29	
h	=	0	0	0	1	0	0	0	
Average Rank		1.48	6.48	1.93	3.93	4.72	5.41	8	4.03

6. Applications

6.1 Bio-Medical Real Life Classification Problems

In this section, the proposed CPFOA is applied for training the Multi-Layer Perceptron (MLP) by solving five datasets. The datasets are a synthesis dataset, 3-bits XOR, a small dataset, Iris, and three bio-medical datasets; Balloon, Breast cancer and Heart. The details of these datasets and parameter setting including the MLP structure for solving these dataset are taken from the literature [53,54]. The brief details of dataset for the implementation and comparison performance of algorithms are presented in Table 14. CPFOA is compared with six meta-heuristic algorithms; JADE, BLPSO, CLPSO, GWO, MGWO and HAGWO [54]. HAGWO was successfully test with these five datasets. It is a hybrid nature-inspired optimization technique that has been constructed using a hybridization of Mean Grey Wolf Optimizer (MGWO) and Whale Optimizer Algorithm (WOA). The parameters of GWO and HAGWO are set as they are in [53,54]. Each algorithm is coded and run in MATLAB environment. The convergent graphs based on five datasets are shown in Figure 9, where the red line represents the graph of CPFOA. From Figure 9, CPFOA can produce the lowest MSE in every dataset. To confirm our claim, the statistical results in term of minimum objective function value, maximum objective function value are extracted and are shown in Table 15. A lower objective function value is better. In addition, the average and the standard deviation of the classification rate are shown in Table 15 as well. A higher classification rate with a lower standard deviation is better.

The results from Table 15 are as follow:

- (i) XOR dataset contains 8 training and 8 testing samples; each sample has 3 input attributes and 1 output. The outputs of the XOR dataset are as same as the input values. After encoding, the dimension of a fruit fly is 36. As can be from Table 15 that the highest accuracy of 100% could be obtained by GWO, MGWO, HAGWO and CPFOA. However, CPFOA is the best as it produces the lowest of the Best MSE Value (1.97E-09), Worst MSE Value (2.43E-04), ave (2.59E-05), and std (5.45E-05) compared with those of the others algorithms.
- (ii) Iris dataset contains 150 samples; each sample has 4 attributes and 3 classes. The training and the testing sets are the same. This problem is harder than XOR as it has three output classes. After encoding, the dimension of a fruit fly is 75. The results show that CPFOA can produce the highest classification rate (93.733%), showed in the boldface. The second and third best competitors are two improved versions of GWO, HAGWO and MGWO, that could obtain an accuracy of 93.00%, and 91.334%,

- 675 respectively. The original GWO ranks forth with the accuracy of (91.333%).
676 Furthermore, CPFOA produces the lowest average MSE (5.39E-02) compared with
677 those of the six competitive algorithms.
678 (iii) Balloon dataset contains 18 training, 18 testing samples; each of samples has 4 input
679 attributes and 2 output classes. After encoding, the dimension of a fruit fly is 55.
680 Obviously seen from Table 15 that this dataset is quite simple because the several
681 algorithms can produce the highest accuracy of 100%, except CLPSO's accuracy is of
682 89.75%. However, CPFOA produces the lowest of Best MSE Value (1.87E-20), Worst
683 MSE Value (2.82E-08), *ave* (1.51E-09) and *std* (6.29E-09) of MSE, compared with those
684 of the six competitive algorithms.
685 (iv) Breast cancer dataset contains 599 training, 100 testing samples, each of which has 9
686 input attributes and 2 output classes. After encoding, the dimension of a fruit fly is
687 209. Therefore, the computational time per iteration is quit long. However, the
688 results confirm that CPFOA can produce the highest accuracy and the lowest values
689 of Best MSE Value (1.21E-03), Worst MSE Value (1.70E-03), *ave* (1.67E-03) and *std*
690 (1.19E-04) of the MSE.
691 (v) Heart dataset, is the hardest problem among the four classification problems,
692 contains 80 training, 187 testing samples, each of sample has 22 input attributes and
693 2 classes. After encoding, the dimension of a fruit fly is 1081. It is a large scale
694 problem. As can be seen from Table 15, several algorithms could produce low
695 accuracies; the accuracy of JADE, BLPSO, CLPSO, GWO, MGWO and HAGWO are
696 77.50%, 67.20%, 68.125%, 75.00 %, 88.375%. HAGWO produces a very low accuracy
697 of 47.625%. However, the CPFOA showed a very promising result as it can reach the
698 highest accuracy of 90.333%. Therefore, CPFOA outperforms the other algorithms.

699 To summarize, CPFOA has the ability and is suitable for applying to train the Multi-Layer
700 Perceptron (MLP) in order to solve real life classification problems.

701
702
703
704
705
706

Figure 9. Convergence curves of XOR, Iris, Balloon, Cancer and Heart dataset that compare JADE, BLPSO, CLPSO, GWO, MGWO, HAGWO and CPFOA.

707

708

709 **Table 14.** The classification datasets [53,54].

Classification datasets	Number of attributes	Number of training samples	Number of test samples	Number of classes
3-bits XOR	3	8	8 as training samples	2
Balloon	4	16	6 as training samples	2
Iris	4	150	150 as training samples	3
Breast cancer	9	599	100	2
Heart	22	80	187	2

710 **Table 15.** Experimental results of Bio-Medical real life problems. The column having the ∇ symbol means
711 lower is better, whereas the column having the Δ symbol means higher is better. The best value of each
712 measure is highlighted with the boldfaces.

(i) XOR dataset problem					
Algorithm	Best MSE value ∇	Worst MSE value ∇	ave MSE value ∇	std MSE value ∇	Classification Rate Δ
JADE	2.03E-06	1.61E-03	5.04E-04	5.43E-04	76.875%
BLPSO	8.81E-04	1.42E-02	4.89E-03	3.79E-03	58.125%
CLPSO	3.17E-09	8.71E-02	3.36E-02	2.17E-02	60.833%
GWO	8.27E-06	1.33E-01	1.56E-02	3.75E-02	100%
MGWO	5.06E-05	2.16E-01	3.48E-02	6.34E-02	100%
HAGWO	4.27E-02	2.30E-01	2.90E-03	4.69E-02	100%
CPFOA	1.97E-09	2.43E-04	2.59E-05	5.45E-05	100%
(ii) Iris dataset problem					
Algorithm	Best MSE value ∇	Worst MSE value ∇	ave MSE value ∇	std MSE value ∇	Classification Rate Δ
JADE	2.02E-02	3.10E-02	2.52E-01	2.89E-03	75.266%
BLPSO	3.51E-02	8.17E-02	5.68E-02	1.45E-02	43%
CLPSO	1.36E-01	3.67E-01	2.59E-01	6.35E-02	47.333%
GWO	6.67E-01	8.76E-01	6.71E-01	2.49E-02	91.333%
MGWO	6.67E-01	8.13E-01	6.79E-01	1.54E-02	91.334%
HAGWO	6.67E-01	8.81E-01	6.73E-01	2.81E-02	93.00%
CPFOA	1.81E-02	3.42E-01	5.39E-02	1.01E-01	93.733%
(iii) Balloon dataset problem					
Algorithm	Best MSE value ∇	Worst MSE value ∇	ave MSE value ∇	std MSE value ∇	Classification Rate Δ
JADE	4.44E-18	3.37E-08	2.24E-09	7.62E-09	100%
BLPSO	1.42E-11	9.61E-08	1.37E-08	2.42E-08	100%
CLPSO	7.45E-06	3.18E-03	9.71E-04	8.77E-04	89.75%
GWO	3.51E-17	1.17E-01	6.40E-03	2.61E-02	100%
MGWO	2.25E-15	5.56E-02	7.10E-03	1.73E-02	100%
HAGWO	1.64E-05	1.80E-01	1.43E-02	4.38E-02	100%
CPFOA	1.87E-20	2.82E-08	1.51E-09	6.29E-09	100%

(iv) Breast cancer dataset problem					
Algorithm	Best MSE value [▽]	Worst MSE value [▽]	ave MSE value [▽]	std MSE value [▽]	Classification Rate ^Δ
JADE	3.79E-03	7.98E-03	6.28E-03	1.30E-03	67.90%
BLPSO	4.56E-03	8.34E-03	6.63E-03	1.07E-03	92.40%
CLPSO	2.26E-02	3.90E-02	3.39E-02	3.91E-03	60%
GWO	1.40E-03	4.64E-02	6.50E-03	9.30E-03	99.00%
MGWO	1.70E-03	3.87E-02	6.60E-03	9.60E-03	99.11%
HAGWO	1.30E-03	4.64E-02	2.60E-03	4.20E-03	100%
CPFOA	1.21E-03	1.70E-03	1.67E-03	1.19E-04	100%
(v) Heart dataset problem					
Algorithm	Best MSE value [▽]	Worst MSE value [▽]	ave MSE value [▽]	std MSE value [▽]	Classification Rate ^Δ
JADE	8.49E-02	1.13E-01	9.86E-02	8.35E-03	77.50%
BLPSO	1.65E-01	2.33E-01	2.01E-01	2.30E-02	67.20%
CLPSO	1.98E-01	2.48E-01	2.23E-01	1.80E-02	68.125%
GWO	4.69E-02	8.13E-02	6.44E-02	1.07E-02	75.00 %
MGWO	4.69E-02	7.81E-02	5.88E-02	8.69E-03	88.375%
HAGWO	9.02E-02	1.42E-01	1.12E-01	1.78E-02	47.625%
CPFOA	5.63E-02	1.06E-01	8.17E-02	1.14E-02	90.333%

6.2 T-S Fuzzy System Parameter Extraction

CPFOA is applied to estimate the T-S fuzzy system parameters. This T-S fuzzy system parameter estimation had been done in several literary works, and seven of existing works are shown in Table 16 to compare with CPFOA. The dataset is Box and Jenkins gas furnace data set [55]. It consists of 296 input and output measurements of a gas-furnace process. It is the collection of recorded data from a combustion process of a methane-air mixture. At each sampling time k , the input $x(k)$ is the gas flow rate and the output $y(k)$ is the output CO₂ concentration.

To compare CPFOA with the other novel algorithms, we borrow a procedure from the literature namely “Parameter estimation of Takagi–Sugeno fuzzy system using heterogeneous cuckoo search algorithm” [56] by choosing $u(k)$, $u(k-1)$, $y(k-1)$ and $y(k-2)$ as the input variables of the model. For preparing the data, the first 148 input–output data were utilized as training data, and the last 148 were the testing data, respectively. The individual parameters setting of simulation and HeCoS are set as same as they are in [56] as follows: $P_a=0.15$, $\alpha=1.3$, $\beta=1.3$, $\delta=0.9$, the number of rules in a model is 3, the maximum iteration (Max_iter) = 2000 and the number of agents ($popsiz$ e) = 40. After the encoding, the total parameters for a T-S fuzzy system model is 39.

The training results of CPFOA are visualized in Figure 10, and the testing results are shown in Figure 11. The blue line is the real data and the red dashed line is the output of model trained by CPFOA. From those two figures, we have observed that the real data and the output from the model are not different by much.

Base on literature [56], the HeCoS optimizing *iTaSuM* claimed that it is the best model with the training and testing MSE values of 0.0271 and 0.138, respectively. These training and testing MSE values are not the lowest values, but the number of rules of the T-S fuzzy system is less than the other six approaches. From Table 16, CPFOA does not obviously outperform HeCoS, in optimizing

iTaSuM. Two systems have the same number of parameters. However, if we focus on the testing MSE, CPFOA can produce a system with a lower testing MSE than HeCoS can.

The parameters of rules produced by CPFOA are presented in Table 15 The parameters shown in Table 6 of [56] are produced by HeCoS. The Euclidean distance between the parameters from two tables is 25.378, which is quite high. This means that Takagi–Sugeno fuzzy system is a highly nonlinear model. Therefore, for achieving a model with a lower testing MSE, a very high efficient optimization algorithm is required. And CPFOA can fit with this requirement.

Figure 10. Comparison of *iTaSuM* optimized by CPFOA and the real system for Box and Jenkins gas furnace training data. (A): The real values and the CPFOA outputs for the data set. (B) The error between the real values and the CPFOA outputs.

Figure 11. Comparison of *iTaSuM* optimized by CPFOA and the real system for Box and Jenkins gas furnace testing data. (A): The real values and the CPFOA outputs for the data set. (B) The error between the real values and the CPFOA outputs.

Table 15. The parameters in each rules of CPFOA in the Box-Jenkins system.

Rule	Parameter	Vectors														
1	C_1	62.3044	61.0918	2.9017	-2.4265	σ_1	46.3485	47.7164	49.0855	49.1978	W_1, b_1	0.4595	-4.2473	2.9734	0.3492	4.1753
2	C_2	61.1341	62.0122	2.6807	-0.9053	σ_2	48.1691	49.2993	49.9984	48.7294	W_2, b_2	1.4795	-3.8469	-2.4239	-4.9634	-2.6189
3	C_3	61.3956	61.4808	2.6451	2.8111	σ_3	48.8828	47.9141	46.7963	49.9989	W_3, b_3	3.1562	5.9989	-0.1805	1.7542	-3.0621

Table 16. Comparison of different models for the Box-Jenkins system.

No.	Models	No. of rules	No. of parameters	Training MSE	Testing MSE
1	Lin et al. [57]	4	44	0.0710	0.261
2	Kim et al. [58]	2	38	0.0340	0.244
3	Tsekouras [59]	7	91	0.0220	0.236
4	Li et al. [60]	3	57	0.0159	0.126
5	Li et al. [61]	3	57	0.0150	0.147
6	Cheung et al. [62]	3	21	0.0590	0.152
7	iTaSuM [56]	3	39	0.0271	0.138
8	CPFOA	3	39	0.0349	0.130

7. Conclusion

This paper addresses a problem of low exploration ability of FOA and its variants. The cooperation of the multileader and the probabilistic random walk strategies is formed as the proposed CPFOA algorithm. CPFOA can be smoothly transited from the exploration stage to intensification stage, accordingly. The experimental results show that the performance of CPFOA does improve over the original FOA and the FOA variants in finding the optimal solution.

The main characteristics of CPFOA in finding the optimal solution can be summarized as follow:

- The CPFOA uses the probabilistic of random walk algorithm with adaptive normalization is the main procedure.
- The CPFOA uses the multileader strategy to further enhance the exploration ability.
- The population having two types of behaviors can prevent the search from becoming trapped in a local optimum, whereas only one population behavior in the existing FOA variants can lead that algorithms to be trapped easily.
- The CPFOA demonstrated its promising performance in solving unconstrained function optimization problems, especially when the dimension of problem is high.

We evaluated the CPFOA's performance in 18 well-known standard benchmark functions, and 30 CEC'2017 functions. The experimental results from the benchmark functions clearly illustrated that the CPFOA outperforms both the original and FOA variants in terms of the convergence speed, the success rate, and the solution accuracy, in finding the optimal solution. The experimental results conducted on CEC'2017 functions show that CPFOA can achieve a very promising accuracy when compared with the well-known competitive algorithms.

CPFOA is applied for training the MLPs to classified real life datasets, the results demonstrate an achieving a higher level of accuracy in classification of the proposed CPFOA trainer than the competitive algorithms.

Moreover, CPFOA is applied for T-S fuzzy system parameter extraction of Box and Jenkins gas furnace data set. The results demonstrate that CPFOA can achieve a very promising accuracy of the T-S fuzzy system in modeling when compared with the best known competitive algorithms.

The future work is applying CPFOA to optimize the multi-objective function problems, which are a very challenging problem.

Author Contributions: The authors have contributed equally in this research and in the writing of this paper. Wirote Apinantanakon designed the experiments, conceptualization, investigation, methodology, software and writing – original draft of this paper. Khamron Sunat supervised, performed the numerical experiments, conceptualization, investigation, methodology, software and writing – original draft of this paper. Sirapat Chiewchanwattana analyzed the numerical results, conceptualization, investigation, methodology, software and writing – original draft of this paper. All authors have read and approved the final version of the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on, 4-6 Oct 1995; pp. 39-43.
2. Dorigo, M.; Maniezzo, V.; Colnari, A. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **1996**, *26*, 29-41, doi:10.1109/3477.484436.
3. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization* **2007**, *39*, 459-471, doi:10.1007/s10898-007-9149-x.
4. Yang, X.S.; Suash, D. Cuckoo Search via Levy flights. In Proceedings of 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), 9-11 Dec. 2009; pp. 210-214.
5. Yang, X.-S. A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, González, J.R., Pelta, D.A., Cruz, C., Terrazas, G., Krasnogor, N., Eds. Springer Berlin Heidelberg: Berlin, Heidelberg, 2010; 10.1007/978-3-642-12538-6_6pp. 65-74.
6. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm and Evolutionary Computation* **2018**, <https://doi.org/10.1016/j.swevo.2018.02.013>, doi:<https://doi.org/10.1016/j.swevo.2018.02.013>.
7. Ma, H.; Shen, S.; Yu, M.; Yang, Z.; Fei, M.; Zhou, H. Multi-population techniques in nature inspired optimization algorithms: A comprehensive survey. *Swarm and Evolutionary Computation* **2018**, <https://doi.org/10.1016/j.swevo.2018.04.011>, doi:<https://doi.org/10.1016/j.swevo.2018.04.011>.
8. Pan, W.-T. A new evolutionary computation approach: Fruit Fly Optimization Algorithm. *2011 Conference of Digital Technology and Innovation Management, Taipei* **2011**.
9. Storn, R.; Price, K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* **1997**, *11*, 341-359, doi:10.1023/A:1008202821328.
10. Goldberg, D.E.; Holland, J.H. Genetic Algorithms and Machine Learning. *Machine Learning* **1988**, *3*, 95-99, doi:10.1023/A:1022602019183.
11. Shan, D.; Cao, G.; Dong, H. LGMS-FOA: An Improved Fruit Fly Optimization Algorithm for Solving Optimization Problems. *Mathematical Problems in Engineering* **2013**, *2013*, 9, doi:10.1155/2013/108768.
12. Pan, Q.-K.; Sang, H.-Y.; Duan, J.-H.; Gao, L. An improved fruit fly optimization algorithm for continuous function optimization problems. *Knowledge-Based Systems* **2014**, *62*, 69-83, doi:<https://doi.org/10.1016/j.knsys.2014.02.021>.
13. Pan, W.-T. Using modified fruit fly optimisation algorithm to perform the function test and case studies. *Connect. Sci* **2013**, *25*, 151-160, doi:10.1080/09540091.2013.854735.
14. Yuan, X.; Dai, X.; Zhao, J.; He, Q. On a novel multi-swarm fruit fly optimization algorithm and its application. *Applied Mathematics and Computation* **2014**, *233*, 260-271, doi:<https://doi.org/10.1016/j.amc.2014.02.005>.

15. Li, H.-z.; Guo, S.; Li, C.-j.; Sun, J.-q. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm. *Knowledge-Based Systems* **2013**, *37*, 378-387, doi:https://doi.org/10.1016/j.knosys.2012.08.015.
16. Hu, R.; Wen, S.; Zeng, Z.; Huang, T. A short-term power load forecasting model based on the generalized regression neural network with decreasing step fruit fly optimization algorithm. *Neurocomputing* **2017**, *221*, 24-31, doi:https://doi.org/10.1016/j.neucom.2016.09.027.
17. Niu, D.; Wang, H.; Chen, H.; Liang, Y. The General Regression Neural Network Based on the Fruit Fly Optimization Algorithm and the Data Inconsistency Rate for Transmission Line Icing Prediction. *Energies* **2017**, *10*, doi:10.3390/en10122066.
18. Zainal Abidin, Z.; Arshad, M.R.; Ngah, U. *A simulation based fly optimization algorithm for swarms of mini autonomous surface vehicles application*; 2011; Vol. 40.
19. Lei, X.; Ding, Y.; Fujita, H.; Zhang, A. Identification of dynamic protein complexes based on fruit fly optimization algorithm. *Knowledge-Based Systems* **2016**, *105*, 270-277, doi:https://doi.org/10.1016/j.knosys.2016.05.019.
20. Cao, G.; Wu, L. Support vector regression with fruit fly optimization algorithm for seasonal electricity consumption forecasting. *Energy* **2016**, *115*, 734-745, doi:https://doi.org/10.1016/j.energy.2016.09.065.
21. Kanarachos, S.; Griffin, J.; Fitzpatrick, M.E. Efficient truss optimization using the contrast-based fruit fly optimization algorithm. *Computers & Structures* **2017**, *182*, 137-148, doi:https://doi.org/10.1016/j.compstruc.2016.11.005.
22. Du, T.-S.; Ke, X.-T.; Liao, J.-G.; Shen, Y.-J. DSLC-FOA : Improved fruit fly optimization algorithm for application to structural engineering design optimization problems. *Applied Mathematical Modelling* **2018**, *55*, 314-339, doi:https://doi.org/10.1016/j.apm.2017.08.013.
23. Wu, L.; Liu, Q.; Tian, X.; Zhang, J.; Xiao, W. A new improved fruit fly optimization algorithm IAFOA and its application to solve engineering optimization problems. *Knowledge-Based Systems* **2018**, *144*, 153-173, doi:https://doi.org/10.1016/j.knosys.2017.12.031.
24. Han, S.-Z.; Pan, W.-T.; Zhou, Y.-Y.; Liu, Z.-L. Construct the prediction model for China agricultural output value based on the optimization neural network of fruit fly optimization algorithm. *Future Generation Computer Systems* **2018**, *86*, 663-669, doi:https://doi.org/10.1016/j.future.2018.04.058.
25. Zhang, X.; Lu, X.; Jia, S.; Li, X. A novel phase angle-encoded fruit fly optimization algorithm with mutation adaptation mechanism applied to UAV path planning. *Applied Soft Computing* **2018**, *70*, 371-388, doi:https://doi.org/10.1016/j.asoc.2018.05.030.
26. Zhang, Y.; Cui, G.; Wu, J.; Pan, W.-T.; He, Q. A novel multi-scale cooperative mutation Fruit Fly Optimization Algorithm. *Knowledge-Based Systems* **2016**, *114*, 24-35, doi:https://doi.org/10.1016/j.knosys.2016.09.027.
27. Han, X.; Liu, Q.; Wang, H.; Wang, L. Novel fruit fly optimization algorithm with trend search and co-evolution. *Knowledge-Based Systems* **2018**, *141*, 1-17, doi:https://doi.org/10.1016/j.knosys.2017.11.001.
28. Krishnanand, K.N.; Ghose, D. Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. *Multiagent Grid Syst.* **2006**, *2*, 209-222.
29. Rajabioun, R. Cuckoo Optimization Algorithm. *Applied Soft Computing* **2011**, *11*, 5508-5518, doi:https://doi.org/10.1016/j.asoc.2011.05.008.
30. Chen, Z.; Tang, H. Cockroach Swarm Optimization. 10.1109/ICCET.2010.5485993.
31. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper Optimisation Algorithm: Theory and application. *Advances in Engineering Software* **2017**, *105*, 30-47, doi:https://doi.org/10.1016/j.advengsoft.2017.01.004.

32. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software* **2017**, *114*, 163-191, doi:https://doi.org/10.1016/j.advengsoft.2017.07.002.
33. Dai, H.; Zhao, G.; Lu, J.; Dai, S. Comment and improvement on "A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example". *Knowledge-Based Systems* **2014**, *59*, 159-160, doi:https://doi.org/10.1016/j.knsys.2014.01.010.
34. Niu, J.; Zhong, W.; Liang, Y.; Luo, N.; Qian, F. Fruit fly optimization algorithm based on differential evolution and its application on gasification process operation optimization. *Knowledge-Based Systems* **2015**, *88*, 253-263, doi:https://doi.org/10.1016/j.knsys.2015.07.027.
35. Babalik, A.; İşcan, H.; Babaoğlu, İ.; Gündüz, M. An improvement in fruit fly optimization algorithm by using sign parameters. *Soft Computing* **2017**, 10.1007/s00500-017-2733-1, doi:10.1007/s00500-017-2733-1.
36. Mirjalili, S. The Ant Lion Optimizer. *Advances in Engineering Software* **2015**, *83*, 80-98, doi:https://doi.org/10.1016/j.advengsoft.2015.01.010.
37. Yang, X.-S. *Nature-inspired Metaheuristic Algorithms*; Luniver Press, 2010.
38. Gupta, S.; Deep, K. A novel Random Walk Grey Wolf Optimizer. *Swarm and Evolutionary Computation* **2018**, https://doi.org/10.1016/j.swevo.2018.01.001, doi:https://doi.org/10.1016/j.swevo.2018.01.001.
39. Mirjalili, S.; Gandomi, A.H. Chaotic gravitational constants for the gravitational search algorithm. *Applied Soft Computing* **2017**, *53*, 407-419, doi:https://doi.org/10.1016/j.asoc.2017.01.008.
40. Pan, W.-T. A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example. *Knowledge-Based Systems* **2012**, *26*, 69-74, doi:https://doi.org/10.1016/j.knsys.2011.07.001.
41. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Information Sciences* **2009**, *179*, 2232-2248, doi:https://doi.org/10.1016/j.ins.2009.03.004.
42. Lee, K.S.; Geem, Z.W. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering* **2005**, *194*, 3902-3933, doi:https://doi.org/10.1016/j.cma.2004.09.007.
43. Fister, I.; Fister, I.; Yang, X.-S.; Brest, J. A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation* **2013**, *13*, 34-46, doi:https://doi.org/10.1016/j.swevo.2013.06.001.
44. Chen, X.; Tianfield, H.; Mei, C.; Du, W.; Liu, G. *Biogeography-based learning particle swarm optimization*; 2017; Vol. 21, pp. 7519-7541.
45. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *Trans. Evol. Comp* **2006**, *10*, 281-295, doi:10.1109/tevc.2005.857610.
46. Wang, Y.; Cai, Z.; Zhang, Q. Differential Evolution With Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Transactions on Evolutionary Computation* **2011**, *15*, 55-66, doi:10.1109/TEVC.2010.2087271.
47. Singh, N.; Singh, S. A Modified Mean Gray Wolf Optimization Approach for Benchmark and Biomedical Problems. *Evolutionary Bioinformatics* **2017**, *13*, 1176934317729413, doi:10.1177/1176934317729413.
48. Zhang, J.; Sanderson, A.C. JADE: Adaptive Differential Evolution With Optional External Archive. *IEEE Transactions on Evolutionary Computation* **2009**, *13*, 945-958, doi:10.1109/TEVC.2009.2014613.
49. Nabil, E. A Modified Flower Pollination Algorithm for Global Optimization. *Expert Systems with Applications* **2016**, *57*, 192-203, doi:https://doi.org/10.1016/j.eswa.2016.03.047.
50. El-Sherbiny, A.; Elhosseini, M.A.; Haikal, A.Y. A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm. *Applied Soft Computing* **2018**, *73*, 24-38, doi:https://doi.org/10.1016/j.asoc.2018.08.028.

51. Liu, Z.; Liu, X.; Cai, X. A new hybrid aerodynamic optimization framework based on differential evolution and invasive weed optimization. *Chinese Journal of Aeronautics* **2018**, *31*, 1437-1448, doi:https://doi.org/10.1016/j.cja.2018.05.002.
52. Zhao, X.; Yao, Y.; Yan, L. Learning algorithm for multimodal optimization. *Computers & Mathematics with Applications* **2009**, *57*, 2016-2021, doi:https://doi.org/10.1016/j.camwa.2008.10.008.
53. Mirjalili, S. How effective is the Grey Wolf optimizer in training multi-layer perceptrons. *Applied Intelligence* **2015**, *43*, 150-161, doi:10.1007/s10489-014-0645-7.
54. Singh, N.; Hachimi, H. A New Hybrid Whale Optimizer Algorithm with Mean Strategy of Grey Wolf Optimizer for Global Optimization. *Mathematical and Computational Applications* **2018**, *23*, 14.
55. Box, G.E.P.; Jenkins, G. *Time Series Analysis, Forecasting and Control*; Holden-Day, Inc.: 1990; pp. 500.
56. Ding, X.; Xu, Z.; Cheung, N.J.; Liu, X. Parameter estimation of Takagi-Sugeno fuzzy system using heterogeneous cuckoo search algorithm. *Neurocomputing* **2015**, *151*, 1332-1342, doi:https://doi.org/10.1016/j.neucom.2014.10.063.
57. Yinghua, L.; Cunningham, G.A. A new approach to fuzzy-neural system modeling. *IEEE Transactions on Fuzzy Systems* **1995**, *3*, 190-198, doi:10.1109/91.388173.
58. Euntai, K.; Minkee, P.; Seungwoo, K.; Mignon, P. A transformed input-domain approach to fuzzy modeling. *IEEE Transactions on Fuzzy Systems* **1998**, *6*, 596-604, doi:10.1109/91.728458.
59. Tsekouras, G.E. On the use of the weighted fuzzy c-means in fuzzy modeling. *Advances in Engineering Software* **2005**, *36*, 287-300, doi:https://doi.org/10.1016/j.advengsoft.2004.12.001.
60. Li, C.; Zhou, J.; Fu, B.; Kou, P.; Xiao, J. T-S Fuzzy Model Identification With a Gravitational Search-Based Hyperplane Clustering Algorithm. *IEEE Transactions on Fuzzy Systems* **2012**, *20*, 305-317, doi:10.1109/TFUZZ.2011.2173693.
61. Li, C.; Zhou, J.; Xiao, J.; Xiao, H. Hydraulic turbine governing system identification using T-S fuzzy model optimized by chaotic gravitational search algorithm. *Engineering Applications of Artificial Intelligence* **2013**, *26*, 2073-2082, doi:https://doi.org/10.1016/j.engappai.2013.04.002.
62. Cheung, N.J.; Ding, X.; Shen, H. OptiFel: A Convergent Heterogeneous Particle Swarm Optimization Algorithm for Takagi-Sugeno Fuzzy Modeling. *IEEE Transactions on Fuzzy Systems* **2014**, *22*, 919-933, doi:10.1109/TFUZZ.2013.2278972.