Hindawi
Complexity
Volume 2022, Article ID 6627409, 19 pages
https://doi.org/10.1155/2022/6627409

Research Article

Chaotic Fruit Fly Algorithm for Solving Engineering Design Problems

M. A. El-Shorbagy [1,2]

[1]*Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia*
[2]*Department of Basic Engineering Science, Faculty of Engineering, Menoufia University, Shebin El-Kom 32511, Egypt*

Correspondence should be addressed to M. A. El-Shorbagy; mohammed_shorbagy@yahoo.com

Received 12 February 2022; Revised 16 March 2022; Accepted 1 April 2022; Published 21 April 2022

Academic Editor: Akif Akgul

The aim of this article is to present a chaotic fruit fly algorithm (CFFA) as an optimization approach for solving engineering design problems (EDPs). In CFFA, the fruit fly algorithm (FFA), which is recognized for its durability and efficiency in addressing optimization problems, was paired with the chaotic local search (CLS) method, which allows for local exploitation. CFFA will be set up to work in two phases: in the first, FFA will be used to discover an approximate solution, and in the second, chaotic local search (CLS) will be used to locate the optimal solution. As a result, CFFA can address difficulties associated with the basic FFA such as falling into local optima, an imbalance between exploitation and exploration, and a lack of optimum solution acquisition (i.e., overcoming the drawback of premature convergence and increasing the local exploitation capability). The chaotic logistic map is employed in the CLS because it has been demonstrated to be effective in improving the quality of solutions and giving the best performance by many studies. The proposed algorithm is tested by the set of CEC’2005 special sessions on real parameter optimization and many EDPs from the most recent test suite CEC’2020. The results have demonstrated the superiority of the proposed approach to finding the global optimal solution. Finally, CFFA′s results were compared to those of earlier research, and statistical analysis using Friedman and Wilcoxon’s tests revealed its superiority and capacity to tackle this type of problem.

1. Introduction

The engineering design problems (EDPs) are extremely significant from both the manufacturing and scientific perspective, where it is a very important and challenging area, especially in the field of engineering for getting designs that have efficient form and are more accurate. Generally, these problems are treated as nonlinear constrained optimization problems (NCOPs). NCOPs are very difficult, and the problem feasible region may be a thin subset of the search domain [1].

Traditionally, NCOPs are solved by some efficient methods such as recursive quadratic programming, projection method, generalized reduced gradient method, penalty method, and a multiplier method [2]. These methods are not efficient since they may only compute local optima, and it is very hard to apply these methods to problems as its feasible region is not convex or the objective function is not differentiable [3].

Because of the drawbacks of traditional optimization approaches, the meta-heuristic optimization algorithm for tackling NCOPs emerged. Meta-heuristic algorithms are considered the best optimization algorithms, where they have several advantages such as resilience, performance reliability, simplicity, ease of implementation, and so on. Meta-heuristic algorithms are divided into several categories, including:

(1) Evolutionary-based algorithms: These algorithms are based on evolutionary theory.
(2) Swarm-based algorithms: These algorithms mimic the social behavior and collective decision-making of different social groups. The reason for achieving a specific goal in these algorithms is typically based on bio-community intelligence and collective action.

(3) The rules of natural physics have been used for the emergence of physics-based algorithms.

(4) Algorithms influenced by human social behavior: Recently, optimization algorithms inspired by human social behavior have been suggested in the literature.

Table 1 contains examples of several classifications offered in the literature.

Swarm intelligence-based algorithms are regarded as one of the most essential types of meta-heuristic algorithms. These algorithms emulated the behavior and features of swarms' systems, for which Gerardo Beni and Jing Wang coined the term "swarm intelligence" (SI) in 1989 [62], the notion of which is critical in computer science and artificial intelligence. As a result, they have been dubbed swarm intelligence algorithms (SIAs). Swarm intelligence algorithms (SIAs) are connected to the study of swarms, or colonies of social creatures, where studies of social behavior in swarms of organisms influenced the development of many effective optimization algorithms. The simulation of bird flocks, for example, resulted in the particle swarm optimization (PSO) method, while studies of ant behavior in the construction of the ant colony optimization (ACO) algorithm [62]. SIAs, on the other hand, are simple in concepts, have a low probability to fall into local optima, and require simple information about the optimization problem without requiring that the objective function or the constraints are derivable or continuous [63]. Due to the drawbacks of traditional approaches, SIAs were commonly employed to solve engineering design problems (EDPs).

Many SIAs are presented today to tackle complex optimization problems. Although they can find promising solutions to optimization problems, they frequently become caught in local optima when the problem is complicated and contains several local optima. Creating hybrid SIAs has the potential to dramatically improve this issue. Most of the time, hybrid SIAs are more resilient and efficient than the basic versions since they may benefit from the advantages of different algorithms that are used in hybrid SIAs [64, 65]. Many researchers sought to design hybrid SIAs to produce more efficient global optimization algorithms. The most popular hybrid SIAs are hybrid cultural-trajectory-based search [66], hybrid of the ant colony and firefly algorithms (HAFA) [67], hybrid harmony search-cuckoo search (HS/CS) algorithm [68], hybrid particle swarm optimization-genetic algorithm (PSO/GA) [69], hybrid krill herd-biogeography-based optimization (KHBBO) algorithm [70], hybrid cat swarm optimization (CSO) [71], hybrid tissue membrane systems (TMS) and the evolution strategy with covariance matrix adaptation (CMA-ES) [72], hybrid grasshopper optimization algorithm-local search (GOA/LS) [73], krill herd-differential evolution (KHDE) [74], hybrid grasshopper optimization algorithm-genetic algorithm (GOA/GA) [75], hybrid bat algorithm with harmony search (BHS) [76], etc.

Recently, hybrid SIAs have become the most widely used method for solving EDPs such as a penalty-guided artificial bee colony (ABC) algorithm [77], hybrid Nelder–Mead simplex search and particle swarm optimization [78], Gaussian quantum-behaved particle swarm optimization [79], hybrid Lévy flight-chaotic local search-whale optimization algorithm (LF-CLS-WOA) [80], self-adaptive strategy-based firefly algorithm [81], hybrid genetic algorithm-particle swarm optimization-sequential quadratic programming (GA-PSO-SQP) [82], sine-cosine grey wolf optimizer [83], and improved moth-flame optimization algorithm (IMFO) [84]. It is now obvious that engineering design problems are a significant problem that scholars are focusing on to offer new hybrid methods for solving it and determining the best solutions.

Fruit fly algorithm (FFA) is a novel SI approach based on the foraging behaviors of fruit flies that competes with current swarm algorithms like particle swarm optimization (PSO). However, the FFA still has certain drawbacks, such as its necessitating long CPU times, which are impractical from an engineering standpoint, and limited convergence accuracy, which makes it easy to get stuck at a local optimal value during the evolution process [85]. As a result, the application of chaos theory to overcome these shortcomings is being researched. In recent years, the chaos theory has been applied to several fields of optimization science. As a new method of global optimization, chaos algorithms have garnered a lot of attention. The characteristics inherent in chaos can enhance algorithms of optimization by avoiding local solutions and enhancing convergence to reach a global solution.

Many researchers [86–97] proposed merging chaos theory and optimization algorithms to overcome these limitations, increase solution quality, and reach the ideal solution. For example, in [86], the chaos algorithm was included in the evolutionary process of the fundamental FFA to tackle the difficulties of poor convergence accuracy and quickly relapsing into the local extremum in the fundamental FFA. That is, in the case of local convergence, the chaos algorithm was used to search for the global optimum in the convergent area's outer space, leap out of the local extremum, and continue to optimize. Also, in [88], the conventional FFA was improved by including a new parameter that was integrated by chaotic to solve global optimization; overall study findings reveal that FFA with Chebyshev map outperforms FFA without Chebyshev map in terms of global optimality reliability and algorithm success rate. In addition, a novel version of FFA with Gaussian mutation operator and chaotic local search strategy (MCFFA) was proposed in [90]. To avoid premature convergence and enhance the algorithm's exploitative tendencies, the Gaussian mutation operator was first included in the basic FFA (MFFA). The chaotic local search approach was then used to improve the swarm of agents′ capacity to search locally (CFFA). MCFFA was used to handle issues involving benchmark functions with various properties and feature selection. MCFFA effectively increased FFA's performance and achieves optimal classification accuracy, according to the findings. Furthermore, chaotic fruit fly optimization [92] was presented as a novel learning technique for early detection and effective evaluation of sepsis, where two new mechanisms, chaotic population initiation

Table 1: Meta-heuristic algorithms classification [4].

Category	Algorithm name	References
Evolutionary-based algorithms [5–8]	Genetic algorithm (GA)	[5, 6]
	Differential evolution (DE)	[7]
	Evolutionary strategy (ES)	[8]
Swarm-based algorithms [9–41]	Particle swarm optimization (PSO)	[9–11]
	Ant colony optimization (ACO)	[12]
	Fruit fly algorithm (FFA)	[13]
	Bacterial foraging (BF)	[14]
	Glowworm swarm optimization (GSO)	[15]
	Grey wolf optimizer (GWO)	[16]
	Whale optimization algorithm (WOA)	[17]
	Firefly algorithm (FA)	[18]
	Moth-flame optimization (MFO)	[19]
	Salp swarm optimization (SSA)	[20]
	Grasshopper optimization algorithm (Goa)	[21]
	Artificial bee colony algorithm (ABCA)	[22]
	Bat algorithm (BA)	[23]
	Monkey algorithm (MA)	[24]
	Cuckoo search algorithm (CSA)	[25]
	Spherical search algorithm (SSA)	[26]
	Social spider optimization (SSO)	[27]
	Marine predators algorithm (MPA)	[28]
	Crow search algorithm (CSA)	[29]
	Krill herd algorithm (KHA)	[30]
	Chimp optimization algorithm (COA)	[31]
	Squirrel search algorithm (SCA)	[32]
	Flower pollination algorithm (FPA)	[33]
	Manta ray foraging optimization (MRFO)	[34]
	Sailfish optimizer (SO)	[35]
	Emperor penguin optimizer (EPO)	[36]
	Spotted hyena optimizer (SHO)	[37]
	Slime mould algorithm (SMA)	[38]
	Coyote optimization algorithm	[39]
	Harris hawks optimization (HHO)	[40]
	Colony predation algorithm (CPA)	[41]
Human behavior-based algorithms [42–48]	Group teaching optimization (GTO)	[42]
	Imperialist competitive algorithm (ICA)	[43]
	Teaching-learning based optimization (TLBO)	[44]
	League champion algorithm (LCA)	[45]
	Political optimizer (PO)	[46]
	Poor and rich optimization (PRO)	[47]
	Hunger games search (HGS)	[48]
Physics-based algorithms [49–61]	Gravitational search algorithm (GSA)	[49]
	Simulated annealing (SA)	[50]
	Artificial electric field optimization (AEFO)	[51]
	Sine-cosine algorithm (SCA)	[52, 53]
	Magnetic optimization algorithm (MOA)	[54]
	Turbulent flow of water-based optimization (TFWBO)	[55]
	Henry gas solubility optimization (HGSO)	[56]
	Archimedes optimization algorithm (AOA)	[57]
	Fireworks algorithm (FA)	[58]
	Mine blast algorithm (MBA)	[59]
	weIghted meaN oF vectOrs (INFO)	[60]
	RUNge Kutta optimizer (RUN)	[61]

and chaotic local search strategy, were added to the original FFA. The positive results showed that the approach developed may be a valuable diagnostic tool for clinical decision assistance. A novel support vector machine (SVM) optimization approach, on the other hand, was given in [95], that is based on an upgraded chaotic fruit fly algorithm (FFA) with a mutation strategy to execute SVM parameter setting turning and feature selection simultaneously. The chaotic particle in the enhanced FFA initializes the fruit fly swarm location and substitutes the distance expression for the fruit fly to find the food source. This strategy has been proved to be more resilient and successful than other well-known optimization methods, especially when it comes to tackling medical diagnosis and credit card problems. Finally, in [97], a new method for parameter identification of a bidirectional inductive power transfer (BIPT) system was proposed utilizing a chaotic-enhanced FFA, which used a chaotic sequence to improve the original FFA′s global optimization capabilities. Simulations demonstrated that the suggested approach is efficient for measuring noise and changes in operating conditions, making it ideal for practical use.

From the above, it is clear that introducing chaos to improve FFA has received wide attention from many researchers. From this motivation, this paper proposed a chaotic fruit fly algorithm (CFFA). CFFA combines the fruit fly algorithm (FFA) with a chaotic local search (CLS) method to expedite optimum seeking and find the optimal solution. In addition, combining FFA global search and procedures of CLS offers the benefits of both methods of optimization, while compensating for their disadvantages to ensure the proposed algorithm's robustness. The main contributions to this paper are:

(1) For solving EDPs, a new algorithm called chaotic fruit fly algorithm (CFFA) is presented and tested.
(2) Demonstrating that combining the fruit fly algorithm (FFA) with a chaotic local search (CLS) strategy in CFFA accelerates optimum seeking and finds the EDPs' best solutions.
(3) Testing the robustness and reliability of CFFA and the ability for finding global solutions by using the test suite (CEC'2005) and many EDPs from the most recent test suite (CEC'2020).
(4) Validating by the numerical analysis results that the proposed algorithm has high performance and prove that statistically.

The following is how this paper is structured: the formulation of the nonlinear constrained optimization problem is discussed in Section 2. The proposed methodology is presented in Section 3. The computational experiment is shown in Section 4. Finally, the conclusion is provided in Section 5.

2. Nonlinear Constrained Optimization Problem

Mathematically, the generic nonlinear constrained optimization problem (NCOP) is expressed as:

Minimize/maximize: $f(x)$.

Subject to:

$$\begin{aligned} g_j &\leq 0 \text{ for } j = 1, \ldots, m, \\ h_e &\leq 0 \text{ for } e = 1, \ldots, l, \end{aligned} \tag{1}$$

where $f, g_1, \ldots, g_j, h_1, \ldots, h_e$ are functions defined on $\mathbb{R}^n$, x is a subset of $\mathbb{R}^n$, and is a vector of n components $x_1, \ldots, x_n$. The above problem must be solved for the values of the variables $x_1, \ldots, x_n$ that satisfy the restrictions and minimize or maximize the function f. The function f is the objective function or criterion function. An unconstrained problem is one in which there are no constraints. If there are constraints, the problem is called a constrained problem, and each of the constraints $g_j \leq 0 \quad \forall\, j = 1, \ldots, m$ is called an inequality constraint, and each of the constraints $h_e \leq 0 \quad \forall\, e = 1, \ldots, l$ is called equality constrain [98].

At solving the optimization problem, we are looking for a global solution and not stock on a local solution. An optimal solution (either maximum or minimum) within a neighboring set of candidate solutions is referred to as a local solution of an optimization problem. A global optimum solution is the best solution among all feasible solutions, not simply those within a neighboring set of candidate solutions [98]. Definition 1 introduces the difference between a local solution and a global solution. Figure 1 illustrates this definition.

Definition 1. Let $x = (x_1, x_2, \ldots, x_n)$ be a feasible solution to a minimization problem with objective function $f(x)$ [98]. Then, x is:

(i) A global minimum if $f(X) \leq f(Y)$ for every feasible point $y = (y_1, y_2, \ldots, y_n)$.
(ii) A local minimum if $f(X) \leq f(Y)$ for all feasible points $y = (y_1, y_2, \ldots, y_n)$ sufficiently close to x.

3. The Proposed Methodology

In this section, we provide a brief overview of both the fruit fly algorithm (FFA) and the chaos theory. The proposed algorithm is then thoroughly described.

3.1. Fruit fly Algorithm.

The FFA [99] is a fruit fly-inspired swarm-based intelligence approach that mimics the fruit fly′s foraging behavior. Fruit flies use their keen sense of smell and eyesight to locate food sources. During foraging, fruit flies can detect the aromas of food sources from a long distance away, and swarms fly towards the food source with the highest concentration of the scent. When the fruit fly gets near enough to the food source, it may use its better vision to pinpoint the exact position of the food supply.

The procedure of foraging is emulated in the FFA by exploring the solution space iteratively. The search technique is divided into two parts: smell-based search and vision-based search. The FFA technique may be characterized as follows, according to the fruit fly′s characteristics:

Step 1: The algorithm′s parameters are set, as well as the swarms′ center position.

Step 2: Smell-based search.

Step 2.1: Determine a suitable position for the food supply towards the center of the swarms at random for each fruit fly.
Step 2.2: The concentration of smell at each site of the fruit fly is determined.

Step 3: Search-based vision.

Step 3.1: With the greatest concentration of smell, the most likely location is determined.

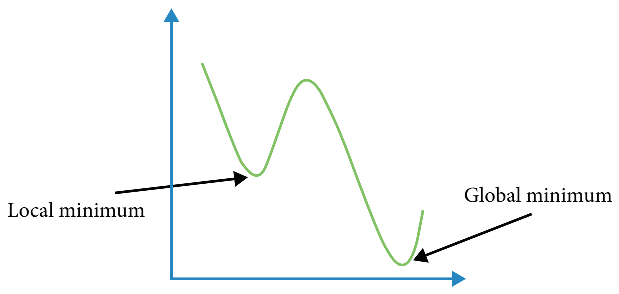

Figure 1: Global minimum and local minimum.

Step 3.2: The fruit fly swarms flock to this location, and the location of the swarm centers is updated.

Step 4: The algorithm is terminated if the stopping condition is fulfilled; otherwise, repeat steps 2 and 3.

3.2. Chaos Theory. By employing extremely unpredictable chaotical sequences, chaos theory (CS) enhances swarm intelligence algorithms [100] and increases convergence and variety of solutions. CS is seen as irregular behavior in nonlinear systems. These maps are meant to represent particles moving in a restricted range of nonlinear dynamic systems, with no knowledge of how the particles move.

To improve solution quality, many researchers proposed combining the CS and optimization algorithms, such as hybrid chaos-PSO [101, 102], chaotic genetic algorithm [103], the combined evolutionary algorithm with chaos [104], chaotic differential bee colony optimization algorithm [105], chaotic DE algorithm [106], chaotic WOA [107], chaotic artificial bee colony (ABC) [108], chaotic harmony search algorithm [109], and chaotic artificial neural networks [110]. There are many well-known chaotic maps, such as the sinusoidal map, Chebyshev map, singer map, tent map, sine map, circle map, Gauss map, logistic map, to be found in the literature [94].

3.3. Chaotic Fruit fly Algorithm. In this section, the chaotic fruit fly algorithm (CFFA) is proposed, which is an integration between the fruit fly algorithm (FFA) and chaos local search (CLS) strategy. The suggested approach is divided into two parts. In the first, FFA is used to discover an approximate solution. Then, in the second stage, CLS is used to speed up convergence, increase solution quality, and reach the optimal solution. The description of the essential idea of the suggested method is as follows:

3.3.1. Phase I: FFA.

Step 1 **(Initialization).** Define the fly group population size $i = 1, \ldots, N$, the iteration termination condition $T_{\max}$, and the starting fruit fly swarm center position $(X_{\text{axis}}, Y_{\text{axis}})$.

Step 2 **(Determination of individual locations).** The position of each fruit fly (X_i, Y_i) is assigned at random as:

$$\begin{aligned} X_i &= X_{\text{axis}} + \text{Random Value}, \\ Y_i &= Y_{\text{axis}} + \text{Random Value}. \end{aligned} \tag{2}$$

Step 3. The judgment value of smell concentration S_i is set as the reciprocal of the distance between the fruit fly and the origin (Dist_i):

$$\begin{aligned} S_i &= \frac{1}{\text{Dist}_i} \\ &= \frac{1}{\sqrt{X_i^2 + Y_i^2}}, \end{aligned} \tag{3}$$

Step 4 **(Repairing infeasible solutions).** A repair approach [111] will be used to deal with the constraint violation at each generation and before the solutions $S_i \forall i = 1, \ldots, N$ are assessed, which will segregate and repair any infeasible solution in the population. The proposed algorithm′s repairing procedure provides a new feasible solution y instead of an infeasible one q on a segment defined by two points: an initial feasible reference point R and any infeasible solution q. A user-specified parameter $\mu \in [0, 1]$ can be used to expand this segment equally on both sides. Therefore, the new feasible solution is produced as:

$$y_1 = \gamma q + (1 - \gamma) R. \tag{4}$$

If y_1 is infeasible, the feasible individual is produced by:

$$y_2 = \gamma R + (1 - \gamma) q. \tag{5}$$

where $\gamma = (1 + 2\mu)\delta - \mu$ and $\delta \in [0, 1]$ is a random number. Figure 2 depicts a schematic representation of a probable sample location for the produced solution.

Step 5 **(Evaluation).** The judgment function of smell concentration (fitness function) of the corresponding position is determined by substituting S_i in the objective function as:

$$\text{Smell}_i = \text{function}(S_i). \tag{6}$$

Step 6 **(Determine the best).** Calculate the minimal concentration of smell and its corresponding location as follows:

$$[\text{best Smell}, \text{best Index}] = \min(\text{Smell}) \quad \forall i. \tag{7}$$

Step 7**(Update swarm center location).** The swarm center position is replaced with the minimum smell location:

$$\begin{aligned} \text{Smell best} &= \text{best Smell}, \\ X_{\text{axis}} &= X(\text{best Index}), \\ Y_{\text{axis}} &= Y(\text{best Index}). \end{aligned} \tag{8}$$

Step 8. Do optimization by repeating Steps 2–6 to determine if the current smell concentration is better

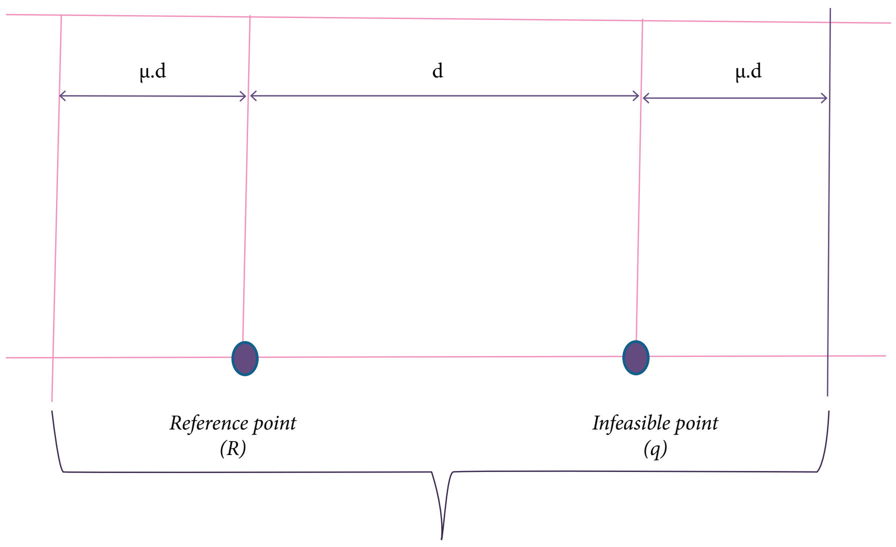

Figure 2: Probable sample location for the produced solution by repairing infeasible solutions.

than the previous smell concentration; if so, go to Step 7. Otherwise, proceed to Step 2 and iterate again.

Step 9. If the stopping criteria are met, the proposed algorithm is stopped. Otherwise, do optimization, and repeat Steps 2 to 8.

For either of the following two conditions, the proposed algorithm is stopped:

(i) Reaching the full predetermined number of generations $T_{\max}$.

(ii) When the individuals of the population converge, i.e., when solutions in the population are identical.

3.3.2. Phase II: Chaotic Local Search. Optimization by using the above-formulated FFA yields an approximate solution $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$. To discover the optimal solution, chaotic local search (CLS) can disturb and explore the local region of the solution x^*. The following is a more extensive description of CLS:

Step 1 **(Determine the boundary range of CLS).** The range of CLS $[a_d, b_d]$, $d = 1, 2, \ldots, n$ for x^* is determined by $x_d^* - \varepsilon < a_d, x_d^* + \varepsilon > b_d \quad \forall d = 1, \ldots, n$, where ε is a specified radius of CLS and set $m = 1$, where m is the CLS iterations $m = 1, 2, \ldots, M$.

Step 2. **Create chaotic variables:** A chaotic number σ^m is generated by the logistic map as:

$$\sigma^m = 4 \times \sigma^{m-1} \times \left(1 - \sigma^{m-1}\right),\ \sigma^0 \in (0, 1), \sigma^0 \notin \{0, 0.25, 0.5, 0.75, 1\}. \tag{9}$$

According to the findings in [94], the logistic map increases the quality of solutions and delivers the best performance. Therefore, it was employed in this study.

Step 3. **Generate a new solution:** By using the chaos variable σ^m and the variance range $[a_d, b_d]$, the new solution is generated as:

$$(x_d^*)^m = a_d + (b_d - a_d)\sigma^m \quad \forall d = 1, \ldots, n. \tag{10}$$

Step 4. **Check feasibility:** If the new solution $(x^*)^m$ is feasible, update the approximate solution x^* as follows: if $f(x^*)^m < f(x^*)$, then set $x^* = (x^*)^m$, otherwise, set $m = m + 1$ and go to Step 2.

Step 5. Stopping CLS: If $m = M$, stop the CLS and put out x^* as the optimal solution. Otherwise, go to Step 2. Figure 3 depicts the suggested algorithm's flowchart.

4. Computational Experiment

In this section, CFFA is evaluated by the set of CEC'2005 special sessions on real parameter optimization to evaluate the performance of the proposed method for global optimization [112]. In addition, the CFFA′s applicability in real-world applications is evaluated in this section using three constrained engineering design problems from the most current test suite CEC'2020 [113]. These problems are common challenges that have been explored by other researchers. The suggested approach is compared to current meta-heuristic algorithms such as simulated annealing (SA) [114], continuous genetic algorithm (CGA) [115], grey wolf optimizer (GWO) [116], moth-flame optimization (MFO) [19], whale optimization algorithm (WOA) [117], Lévy-flight moth-flame optimization (LMFO) [118], water-cycle moth-flame optimization (WCMFO) [119], chimp optimization algorithm (ChOA) [31], arithmetic

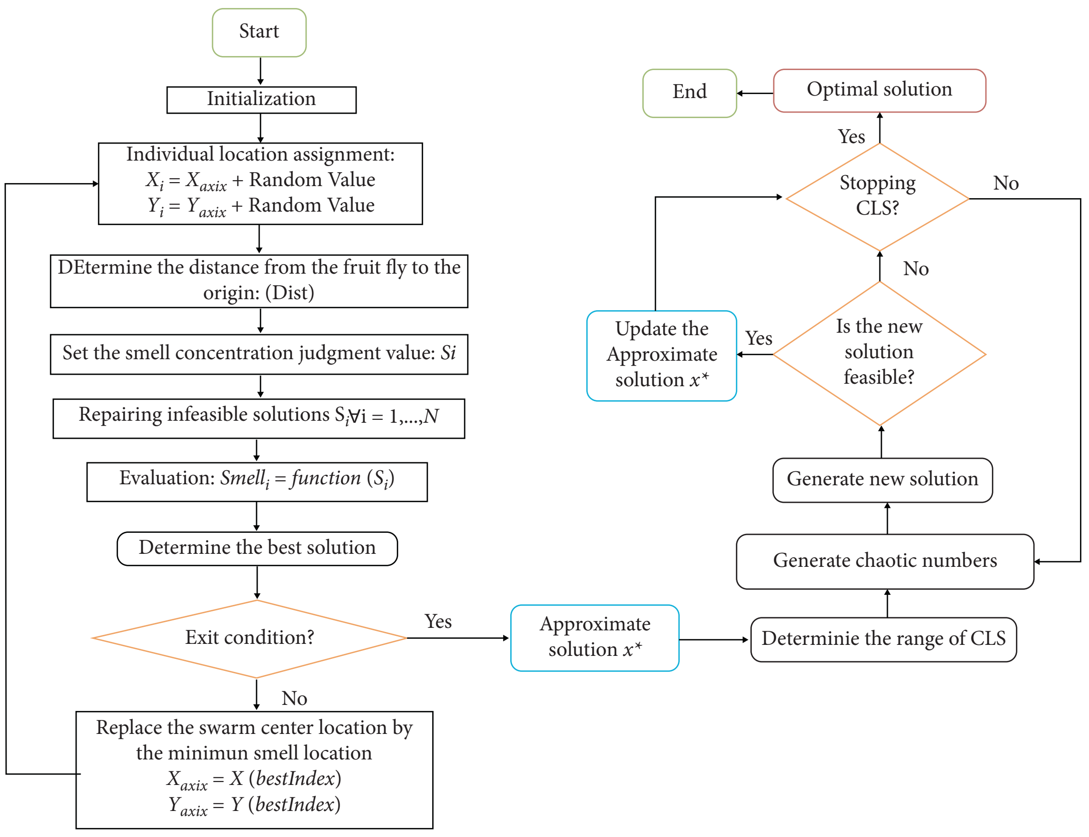
Start
Initialization
Individual location assignment:
$X_i = X_{axix}$ + Random Value
$Y_i = Y_{axix}$ + Random Value
DEtermine the distance from the fruit fly to the origin: (Dist)
Set the smell concentration judgment value: *Si*
Repairing infeasible solutions $S_i \forall i = 1,\ldots,N$
Evaluation: $Smell_i = function\ (S_i)$
Determine the best solution
Exit condition?
Yes
No
Replace the swarm center location by the minimun smell location
$X_{axix} = X\ (bestIndex)$
$Y_{axix} = Y\ (bestIndex)$
Approximate solution x^*
Determinie the range of CLS
Generate chaotic numbers
Generate new solution
Is the new solution feasible?
Yes
Update the Approximate solution x^*
No
Stopping CLS?
No
Yes
Optimal solution
End

Figure 3: The suggested algorithm′s flowchart.

optimization algorithm (AOA) [120], sine-cosine moth-flame optimization (SMFO) [121], and improved moth-flame optimization (IMFO) [122].

The suggested method was coded in MATLAB (R2012b) and tested on a PC with an Intel(R) Core(TM) i7-6600U processor running at 2.60 GHz, 16 GB of RAM, and a Windows 10 operating system. Table 2 shows the parameter settings of comparing algorithms as they were in their original articles. The algorithms were running 20 times with the population size (N) 20 and maximum iterations $T_{\max} = (D \times 10^4)/N$ to ensure a fair comparison. The nonparametric Friedman test, on the other hand, is used to analyze the results statistically. Also, the Wilcoxon signed-rank test is employed to ensure that valid comparisons between all algorithms are made.

4.1. Computational Experiment for CEC'2005. In this part, 25 unconstrained test problems of dimension 10 from the CEC'2005 special session on real parameter optimization are used to evaluate CFFA. The following are the specifics of these functions:

(i) 5 unimodal functions:

F_1 : Shifted sphere, F_2 : Shifted Schwefel's, F_3 : Shifted rotated high conditioned elliptic, F_4 : Shifted Schwefel's with noise in fitness, and F_5 : Schwefel's with global optimum on bounds.

(ii) 20 multimodal functions:

7 basic functions $\longrightarrow$ F_6 : Shifted Rosenbrock's, F_7 : Shifted rotated Griewank without bounds, F_8 : Shifted rotated Ackley's with global optimum on bounds, F_9 : Shifted Rastrigin's, F_{10} : Shifted rotated Rastrigin's, F_{11} : Shifted rotated Weierstrass, and F_{12} : Schwefel's.

2 expanded functions $\longrightarrow$ F_{13} : Expanded extended Griewank's plus Rosenbrock's (F_8F_2) and F_{14} : Shifted rotated expanded Scaffers F_6.

11 hybrid functions. Each one (F_{15} to F_{25}) is created by combining ten of the fourteen preceding functions (different in each case).

All functions are displaced to guarantee that their optimum is never discovered in the search space′s center. Furthermore, the optima cannot be identified inside the initialization range in two functions, and the search scope is not limited.

These test functions are solved by PSO [123], IPOP-CMA-ES [124], CHC [125], SSGA [126], SS-BLX [127], SS-Arit [128], DE-Bin [129], DE-Exp [129], SaDE [130], and the proposed algorithm CFFA. For each test function, all of the algorithms were performed 50 times. Each run ends when the obtained error is less than 10^{-8} or at the maximum number of evaluations (10^{-5}), whichever comes first. Table 3 presents a comparison of the average error achieved by CFFA and 9 continuous optimization techniques. Table 3 confirms that, on average, CFFA produces better solutions than all nine continuous optimization techniques.

Table 2: The CFFA and other competing algorithms′ parameter settings.

Parameter Settings for Algorithms	
SA	$T_0 = 10$.
CGA	$IPMut = 0.9$, $PX\text{cross} = 0.5$.
GWO	The parameter a is linearly decreased from 2 to 0.
MFO	$b = 1$, a is decreased linearly from −1 to −2.
WOA	α variable decreases linearly from 2 to 0, $b = 1$.
LMFO	$\beta = 1.5$, μ and ν are normal distributions, Γ is the gamma function
WCMFO	The number of rivers and sea = 4.
ChOA	f decreases linearly from 2 to 0.
AOA	$\mu = 0.5$, $\alpha = 5$.
SMFO	r_4 = random number between interval (0, 1).
I-MFO	$\delta_1 = 2.02$, $\delta_2 = 1.08$, NF = random number between 1 and D.
CFFA	The specified radius ε is 0.1, $\sigma^0 = 0.001$, and $M = 100$

Table 3: The average error of the 25 CEC'2005 benchmark functions as determined by CFFA and comparing algorithms.

Function	PSO	IPOP-CMA-ES	CHC	SSGA	SS-BLX	SS-Arit	DE-Bin	DE-Exp	SaDE	CFFA
F_1	1.234E-4	0	2.464	8.420E-9	34.02	1.064	7.716E-9	8.260E-9	8.416E-9	0
F_2	0.02595	0	0.0118	8.719E-5	1.730	5.282	8.342E-9	8.181E-9	8.208E-9	0
F_3	51740	0	269900	79480	184400	253500	42.33	99.35	6560	20.8036
F_4	2.488	2932	91.9	2.585E-3	6.228	5.755	7.686E-9	8.350E-9	8.087E-9	0
F_5	409.5	8.104E-10	264.1	134.3	2.185	14.43	8.608E-9	8.514E-9	8.640E-9	2.600E-5
F_6	731	0	1416000	6.171	114.5	494.5	7.956E-9	8.391E-9	0.01612	0.635
F_7	26.78	1267	1269	1271	1966	1908	1266	1265	1263	4.831
F_8	20.43	20.01	20.34	20.37	20.35	20.36	20.33	20.38	20.32	14.54
F_9	14.38	28.41	5.886	7.286E-9	4.195	5.960	4.546	8.151E-9	8.330E-9	0.000
F_{10}	14.04	23.27	7.123	17.12	12.39	21.79	12.28	11.18	15.48	4.541
F_{11}	5.590	1.343	1.599	3.255	2.929	2.858	2.434	2.067	6.796	3.094
F_{12}	636.2	212.7	706.2	279.4	150.6	241.1	106.1	63.09	56.34	5.732
F_{13}	1.503	1.134	82.97	67.13	32.45	54.79	1.573	64.03	70.70	1.052
F_{14}	3.304	3.775	2.073	2.264	2.796	2.970	3.073	3.158	3.415	2.501
F_{15}	339.8	193.4	275.1	292	113.6	128.8	372.2	294	84.23	0
F_{16}	133.3	117	97.29	105.3	104.1	113.4	111.7	112.5	122.7	83.85
F_{17}	149.7	338.9	104.5	118.5	118.3	127.9	142.1	131.2	138.7	107.3
F_{18}	851.2	557	879.9	806.3	766.8	657.8	509.7	448.2	532	479.2
F_{19}	849.7	529.2	879.8	889.9	755.5	701	501.2	434.1	519.5	458.1
F_{20}	850.9	526.4	896	889.3	746.3	641.1	492.8	418.8	476.7	335.1
F_{21}	913.8	442	815.8	852.2	485.1	500.5	524	542	514	394.5
F_{22}	807.1	764.7	774.2	751.9	682.8	694.1	771.5	772	765.5	632.7
F_{23}	1028	853.9	1075	1004	574	582.8	633.7	582.4	650.9	594.7
F_{24}	412	610.1	295900	236	251.3	201.1	206	202	200	210.5
F_{25}	509.9	1818	1764	1747	1794	1804	1744	1742	1738	274.3

4.1.1. The Nonparametric Friedman Test for CEC′2005 Results. The Friedman test is used to statistically rank the significance of algorithms [131]. Table 4 summarizes the outcomes obtained by this test. According to this statistical analysis, the CFFA ranks top, and because the obtained P value is less than 0.05 ($\alpha = 0.000$), there are substantial variations in the performances of the CFFA and the other algorithms tested. Figure 4 includes the chart that depicts the ranking of the CFFA and competitor algorithms. The smallest bar on the graph represents the best algorithm, while the largest represents the worst. The chart reveals that the CFFA obtained the shortest bar with a mean rank equal to 2.12, while PSO obtained the largest bar with a mean rank equal to 7.76. As a result, the chart reveals that the CFFA beats other algorithms by obtaining the first rank (shortest bar).

4.1.2. The Nonparametric Wilcoxon Signed-Rank Test for CEC'2005 Results. To demonstrate the substantial differences between the CFFA and the other algorithms, the Wilcoxon signed-rank test is performed [132]. The Wilcoxon signed-rank test results are shown in Table 5. $R+$ is the sum of positive ranks, whereas R− is the sum of negative ranks. Table 5 shows that CFFA beats other algorithms by achieving $R+$ values larger than R− values in all comparisons. As a consequence, we may conclude that the suggested CFFA is a significant algorithm that outperforms the others.

4.2. Computational Experiment for Engineering Design Problems. The proposed method and rival algorithms compete in this evaluation to solve three different problems: a gas transmission compressor design problem, a three-bar

Table 4: Friedman test results for the 25 CEC′2005 benchmark functions.

Test Statistics	
N	25
Chi-square	66.126
df	9
Asymp. Sig.	0.000
Ranks	
Algorithm	Mean rank
PSO	7.76
IPOP-CMA-ES	5.36
CHC	7.20
SSGA	6.60
SS-BLX	5.80
SS-Arit	6.40
DE-Bin	4.60
DE-Exp	4.28
SaDE	4.88
CFFA	2.12

truss design problem, and a tension/compression spring design problem.

(1) P_1: Design of gas transmission compressor problem. The basic purpose of the gas transmission compressor design challenge is to minimize the objective function utilizing four design variables which are length between compressor stations $L = x_1$, compression ratio that denotes the inlet pressure to the compressor $r = x_2$, and inner diameter of the pipe $D = x_3$. Figure 5 and (11) depict and formulate this problem.

$$\begin{gathered}\text{Min } f(x) = 8.61 \times 10^5 x_1^{0.5} x_2 x_3^{-2/3} x_4^{-0.5} + 3.69 \times 10^4 x_3 + \frac{7.72 \times 10^8 x_2^{0.219}}{x_1} - \frac{765.43 \times 10^6}{x_1}, \\ \text{Subject to: } x_4 x_2^{-2} + x_2^{-2} - 1 \le 0, 20 \le x_1 \le 50,\ 1 \le x_2 \le 10, 20 \le x_3 \le 45, 0.1 \le x_4 \le 60.\end{gathered} \tag{11}$$

(2) P_2: Three-bar truss problem. The three-bar truss design is an engineering optimization problem to evaluate the optimal cross-sectional areas $A_1 = A_3 = x_1$ and $A_2 = x_2$ such that the volume of the statically loaded truss structure $f(x)$ is minimized while stress constraints σ are taken into consideration. The mathematical model of this problem is formulated using three constraints and two variables. Figure 6 and (12) show the formulation and schematic of this problem.

$$\begin{aligned}&\text{Min } f(x) = H\left(x_2 + 2\sqrt{2}\,x_1\right), \\ &\text{Subject to: } \frac{x_2}{2x_1x_2 + \sqrt{2}x_1^2} P - \sigma \le 0, \frac{x_2 + \sqrt{2}x_1}{2x_1x_2 + \sqrt{2}x_1^2} P - \sigma \le 0, \frac{1}{x_1 + \sqrt{2}x_2} P - \sigma \le 0, \\ &H = 100cm, P = \frac{2KN}{cm^2}, \sigma = \frac{2KN}{cm^2}, 0 \le x_1, x_2 \le 1.\end{aligned} \tag{12}$$

(3) P_3: Tension/compression spring design problem. The tension/compression spring design challenge′s purpose is to lower the tension/compression spring′s weight by considering three variables and four constraints. Wire diameter $(d = x_1)$, mean coil diameter $(D = x_2)$, and the number of active coils $(N = x_3)$ are the variables (as indicated in Figure 7). (13) describes the problem and its constraints.

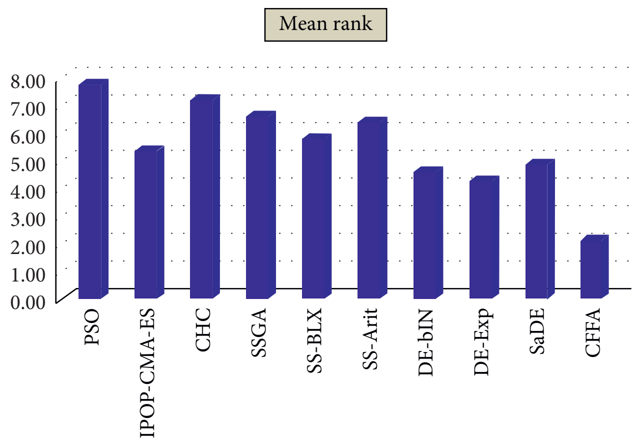

FIGURE 4: The Friedman test's mean -ranking on CFFA and its 9 competitors.

TABLE 5: The results of Wilcoxon's signed-rank test for the 25 CEC'2005 benchmark functions.

Test Statistics				Ranks		
				N	Mean Rank	Sum of Ranks
SaDE – CFFA		R$^-$	3[a]	7.00	21.00	a. SaDE < CFFA
Z	−3.807[ab]	**R$^+$**	22[b]	13.82	304.00	b. SaDE > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[c]			c. SaDE = CFFA
ab. Based on negative ranks.		**Total**	25			
DE-Exp - CFFA		**R$^-$**	7[d]	10.29	72.00	d. DE-Exp < CFFA
Z	−2.435[ab]	**R$^+$**	18[e]	14.06	253.00	e. DE-Exp > CFFA
Asymp. Sig. (2-Tailed)	0.015	**Ties**	0[f]			f. DE-Exp = CFFA
ab. Based on negative ranks.		**Total**	25			
DE-Bin - CFFA		**R$^-$**	4[g]	7.00	28.00	g. DE-Bin < CFFA
Z	−3.619[ab]	**R$^+$**	21[h]	14.14		h. DE-Bin > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[i]			i. DE-Bin = CFFA
ab. Based on negative ranks.		**Total**	25			
SS-Arit - CFFA		**R$^-$**	3[j]	6.00	18.00	j. SS-Arit < CFFA
Z	−3.888[ab]	**R$^+$**	22[k]	13.95	307.00	k. SS-Arit > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[l]			l. SS-Arit = CFFA
ab. Based on negative ranks.		**Total**	25			
SS-BLX - CFFA		**R$^-$**	2[m]	6.00	12.00	m. SS-BLX < CFFA
Z	−4.049[ab]	**R$^+$**	23[n]	13.61	313.00	n. SS-BLX > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[0]			o. SS-BLX = CFFA
ab. Based on negative ranks.		**Total**	25			
SSGA - CFFA		**R$^-$**	1[p]	6.00	6.00	p. SSGA < CFFA
Z	−4.211[ab]	**R$^+$**	24[q]	13.29	319.00	q. SSGA > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[r]			r. SSGA = CFFA
ab. Based on negative ranks.		**Total**	25			
CHC - CFFA		**R$^-$**	3[s]	3.67	11.00	s. CHC < CFFA
Z	−4.076[ab]	**R$^+$**	22[t]	14.27	314.00	t. CHC > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[u]			u. CHC = CFFA
ab. Based on negative ranks.		**Total**	25			
IPOP-CMA-ES - CFFA		**R$^-$**	4[v]	4.25	17.00	v. IPOP-CMA-ES < CFFA
Z	−3.680[ab]	**R$^+$**	19[w]	13.63	259.00	w. IPOP-CMA-ES > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	2[x]			x. IPOP-CMA-ES = CFFA
ab. Based on negative ranks.		**Total**	25			
PSO - CFFA		**R$^-$**	0[y]	0.00	0.00	y. PSO < CFFA
Z	−4.372[ab]	**R$^+$**	25[z]	13.00	325.00	z. PSO > CFFA
Asymp. Sig. (2-Tailed)	0.000	**Ties**	0[aa]			aa. PSO = CFFA
ab. Based on negative ranks.		**Total**	25			

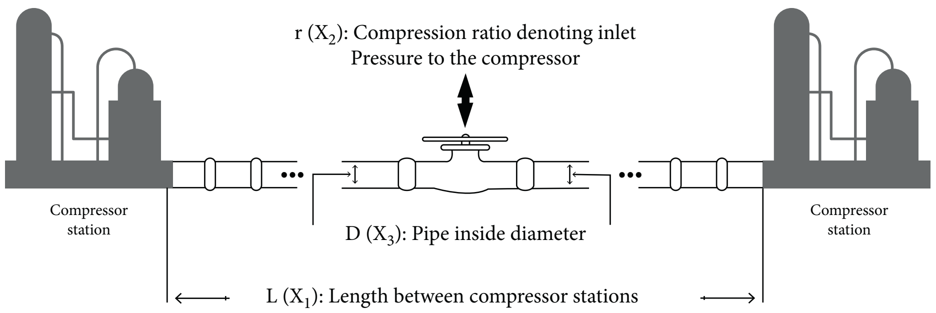
r (X₂): Compression ratio denoting inlet
Pressure to the compressor
Compressor station
Compressor station
D (X₃): Pipe inside diameter
L (X₁): Length between compressor stations

FIGURE 5: Gas transmission compressor problem design.

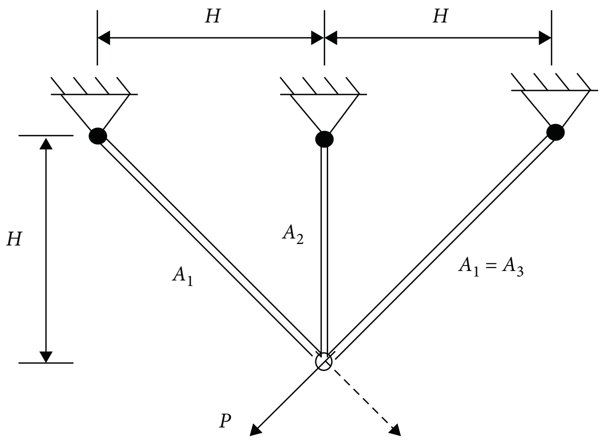

FIGURE 6: Three-bar truss problem design.

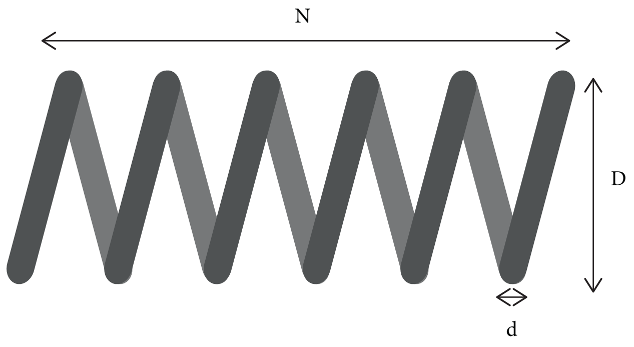

FIGURE 7: Design of tension/compression spring problem.

$$\text{Subject to: } 1-\frac{x_2^3x_3}{71785x_1^4}\leq 0\,\frac{4x_2^2-x_1x_2}{12566\left(x_1^3x_2-x_1^4\right)}+\frac{1}{5108x_1^2}-1\leq 01-\frac{140.45x_1}{x_2^2x_3}\leq 0,\frac{x_1+x_2}{1.5}-1\leq 0, \tag{13}$$

$$0.05\leq x_1\leq 2, 0.25\leq x_2\leq 1.3, 2\leq x_3\leq 15.$$

4.2.1. Engineering Design Problems Results. The suggested CFFA and comparative algorithms are compared. The results of these experiments are summarized in Table 6, which demonstrates that the CFFA technique outperforms other algorithms in obtaining a good approximation to the best values for low-weight variables.

Figures 8–10 also illustrate the convergence curves of the best function values obtained by CFFA before and after CLS for the gas transmission compressor design problem, the three-bar truss design problem, and the tension/compression spring design problem, respectively. Figures 8–10 show

Table 6: Results of the engineering design problems.

Algorithms	Gas transmission compressor: $\mathbf{T}_{\mathbf{max}} = 2000$					Three-bar truss problem: $\mathbf{T}_{\mathbf{max}} = 1000$			Tension/compression spring: $\mathbf{T}_{\mathbf{max}} = 1500$			
	Optimal values				Optimal weight	Optimal values		Optimal weight	Optimal values			Optimal weight
	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$		$\mathbf{x}_1$	$\mathbf{x}_2$		d	D	N	
SA	46.76	1.62	25.79	0.55	4390311	0.768630	0.474232	264.82456	0.075935	0.993094	3.879891	0.033670
CGA	49.97	20.01	31.47	49.83	17350230	0.792428	0.397752	263.90770	0.071031	1.019975	1.726076	0.019749
GWO	20.00	7.81	20.00	60.00	2964974	0.787771	0.410872	263.89619	0.051231	0.345699	11.970135	0.012676
MFO	50.00	1.18	24.57	0.39	2964902	0.789186	0.406806	263.89603	0.053064	0.390718	9.542437	0.012699
WOA	50.00	1.18	24.86	0.39	2965002	0.787713	0.410977	263.89653	0.050451	0.327675	13.219341	0.012694
LMFO	49.46	1.18	24.64	0.39	2965456	0.791713	0.399909	263.92114	0.050000	0.317154	14.107156	0.012771
WCMFO	50.00	1.18	24.61	0.39	2964897	0.788472	0.408822	263.89589	0.051509	0.352411	11.545969	0.012666
ChOA	50.00	1.19	24.24	0.41	2966828	0.787802	0.410724	263.89653	0.051069	0.341746	12.251078	0.012702
AOA	50.00	1.23	20.00	0.51	3014615	0.792789	0.396906	263.92526	0.050000	0.310475	15.000000	0.013195
SMFO	23.66	1.09	23.66	0.19	3052254	0.792044	0.398859	263.90973	0.050000	0.314692	14.696505	0.013136
I-MFO	50.00	1.18	24.60	0.39	2964896	0.788792	0.407919	263.89585	0.051710	0.357217	11.259785	0.012665
FFA	49.8660	1.1832	23.5344	0.3999	**2966272**	0.779348	0.435289	**263.96190**	0.0538558	0.405526	10.106720	**0.014240**
CFFA	49.9999	1.1782	24.5936	0.3882	**2964895**	0.788661	0.408286	**263.89584**	0.051813	0.359726	11.114753	**0.012665**

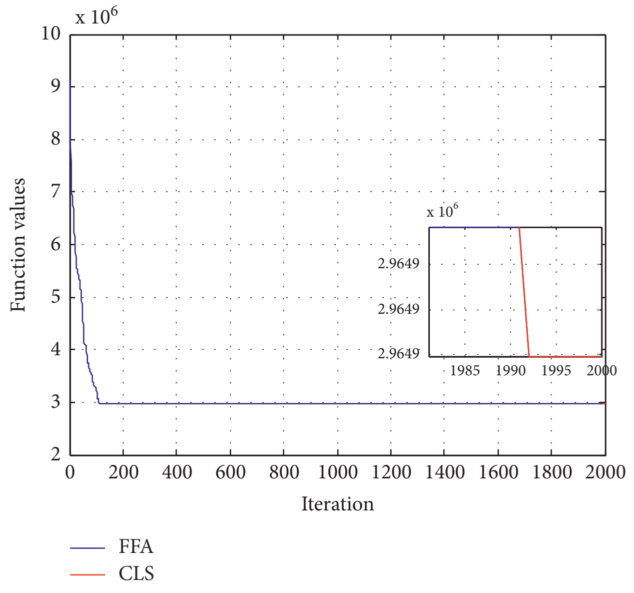

Figure 8: CFFA′s convergence curve of gas transmission compressor design problem before and after CLS.

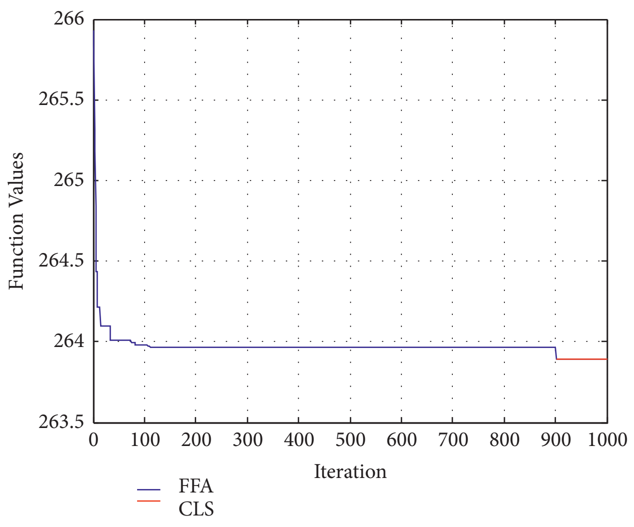

Figure 9: CFFA′s convergence curve of the three-bar truss design problem before and after CLS.

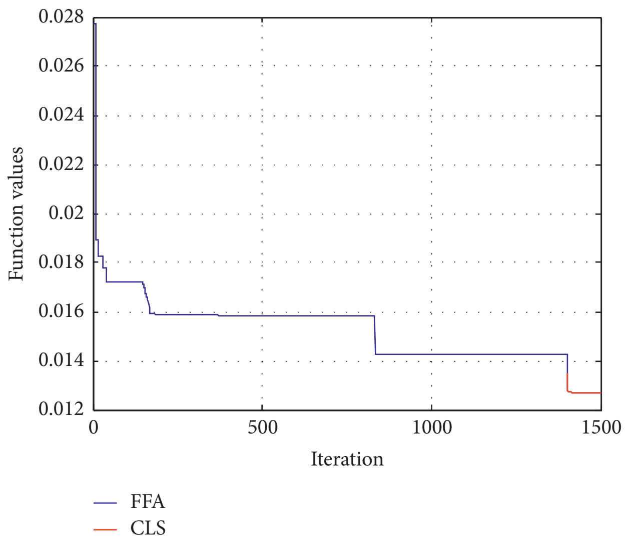

Figure 10: CFFA's convergence curve of tension/compression spring design problem before and after CLS.

how the basic FFA (before CLS/blue line) stuck in the local minimum for a long time. Figures, on the other hand, show how CLS disturbs and explore the local region of the approximate solution obtained by FFA and how it accelerates convergence, enhances solution quality, and finds the optimal solution (after CLS/red line). So, we can conclude that the convergence curves showed the importance of the introduction of the chaotic local search (CLS) on fruit fly algorithm (FFA) which improves the FFA results of FFA and help it to exit from the local optimal solution (blue line) and access to the globally optimal solution (red line).

(1) The nonparametric (Friedman & Wilcoxon signed-rank) tests for engineering design problems results. The findings of the Friedman test for engineering design problems are displayed in Table 7. The CFFA ranks top in this statistical study, and because the calculated p -value is less than 0.05 ($\alpha = 0.001$), there are significant differences between the CFFA and the other comparing algorithms. Figure 11 also shows a chart that shows the CFFA and rival algorithms' rankings. As previously stated, the best algorithm is represented by the shortest bar on the graph, while the poorest is represented by the biggest bar. The CFFA has the smallest bar, with a mean rank of 1.17, while SA has the largest bar, with a mean rank of 11.67. The basic FFA, on the other hand, obtains the 10th rank among all algorithms. As a consequence, the chart shows that the CFFA outperforms other algorithms by having the first rank (shortest bar).

The Wilcoxon signed-rank test findings for engineering design problems, on the other hand, are presented in Table 8. In all comparisons, CFFA outperforms other algorithms, as shown in Table 8, by reaching R^+ values greater than R^- values. We can see that the statistical results of engineering design problems do not differ from the results of problems CEC'2005, as the presented method CFFA outperformed the rest of the other algorithms. As a consequence, we may conclude that the suggested CEGA is a significant algorithm that outperforms the others in the computational experiment.

Table 7: Friedman test′ results for the engineering design problems.

Test Statistics			
N		3	
Chi-square		31.25	
df		11	
Asymp. Sig.		**0.001**	
	Ranks		
Method	**Mean rank**	**Method**	**Mean rank**
SA	12.67	**CGA**	11.00
GWO	4.67	**MFO**	4.67
WOA	5.83	**LMFO**	8.33
WCMFO	3.00	**ChOA**	7.50
AOA	10.33	**SMFO**	9.67
I-MFO	1.83	**FFA**	10.33
CFFA	1.17		

4.3. Discussions. Table 3 displayed the average error for all algorithms for the CEC′2005 benchmark functions, whereas Table 6 showed the best solution for all algorithms for the engineering design issues. Tables 3 and 6 indicated that CFFA beat other algorithms in terms of producing better results. Statistically, the Fridman test, as shown in Tables 4 and 7, demonstrated that the Asymp. Sig. (P value) is less than 0.05, suggesting that there are differences in the results obtained by all algorithms. Furthermore, as demonstrated in Tables 4 and 7 and Figures 4 and 11, CFFA beat the other algorithms by obtaining the lower mean rank. Tables 5 and 8, on the other hand, presented the Wilcoxon signed-rank test findings to investigate the major differences between the comparison methods. They proved that CFFA outperformed other algorithms by achieving more positive rank values (R^+) than negative rank values (R^-) in each of CEC'2005 benchmark functions and engineering design problems. The convergence curves also showed the importance of the proposed method, as the introduction of the chaotic local search (CLS) on fruit fly algorithm (FFA) proved its importance and the ability of the CLS to improve the results of

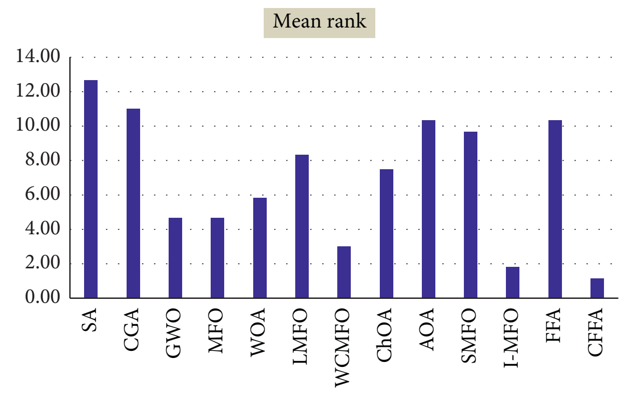

Figure 11: The Friedman test's mean-ranking on CFFA and its 11 competitors.

Table 8: The results of Wilcoxon's signed-rank test for the engineering design problems.

Test statistics					Ranks	
				N	Mean rank	Sum of ranks
SA - CFFA		R^-	0^{a}	0.00	0.00	a. SA < CFFA
Z	-1.604^{ak}	**R^+**	3^{b}	2.00	6.00	b. SA > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{c}			c. SA = CFFA
ak. Based on negative ranks		**Total**	3			
CGA - CFFA		**R^-**	0^{d}	0.00	0.00	d. CGA < CFFA
Z	-1.604^{ak}	**R^+**	3^{e}	2.00	6.00	e. CGA > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{f}			f. CGA = CFFA
ak. Based on negative ranks		**Total**	3			
GWO - CFFA		**R^-**	0^{g}	0.00	0.00	g. GWO < CFFA
Z	-1.604^{ak}	**R^+**	3^{h}	2.00	6.00	h. GWO > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{i}			i. GWO = CFFA
ak. Based on negative ranks		**Total**	3			
MFO - CFFA		**R^-**	0^{j}	0.00	0.00	j. MFO < CFFA
Z	-1.604^{ak}	**R^+**	3^{k}	2.00		6.00
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{l}			l. MFO = CFFA
ak. Based on negative ranks		**Total**	3			
WOA - CFFA		**R^-**	0^{m}	0.00	0.00	m. WOA < CFFA
Z	-1.604^{ak}	**R^+**	3^{n}	2.00	6.00	n. WOA > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{o}			o. WOA = CFFA
ak. Based on negative ranks		**Total**	3			
LMFO - CFFA		**R^-**	0^{p}	0.00	0.00	p. LMFO < CFFA
Z	-1.604^{ak}	**R^+**	3^{q}	2.00	6.00	q. LMFO > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{r}			r. LMFO = CFFA
ak. Based on negative ranks		**Total**	3			
WCMFO - CFFA		**R^-**	0^{s}	0.00	0.00	s. WCMFO < CFFA
Z	-1.604^{ak}	**R^+**	3^{t}	2.00	6.00	t. WCMFO > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{u}			u. WCMFO = CFFA
ak. Based on negative ranks		**Total**	3			
ChOA - CFFA		**R^-**	0^{v}	0.00	0.00	v. ChOA < CFFA
Z	-1.604^{ak}	**R^+**	3^{w}	2.00	6.00	w. ChOA > CFFA
Asymp. Sig. (2-Tailed)		0.109	**Ties**			x. ChOA = CFFA
ak. Based on negative ranks		**Total**	3			
AOA - CFFA		**R^-**	0^{y}	0.00	0.00	y. AOA < CFFA
Z	-1.604^{ak}	**R^+**	3^{z}	2.00	6.00	z. AOA > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{aa}			aa. AOA = CFFA
ak. Based on negative ranks		**Total**	3			
SMFO - CFFA		**R^-**	0^{ab}	0.00	0.00	ab. SMFO < CFFA
Z	-1.604^{ak}	**R^+**	3^{ac}	2.00	6.00	ac. SMFO > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0^{ad}			ad. SMFO = CFFA
ak. Based on negative ranks		**Total**	3			

Table 8: Continued.

Test statistics				Ranks		
				N	Mean rank	Sum of ranks
I-MFO - CFFA		**R⁻**	0[ae]	0.00	0.00	ae. I-MFO < CFFA
Z	−1.342[ak]	**R⁺**	2[af]	1.50	3.00	af. I-MFO > CFFA
Asymp. Sig. (2-Tailed)	0.180	**Ties**	1[ag]			ag. I-MFO = CFFA
ak. Based on negative ranks		**Total**	3			
FFA - CFFA		**R⁻**	0[ah]	0.00	0.00	ah. FFA < CFFA
Z	−1.604[ak]	**R⁺**	3[ai]	1.50	3.00	ai. FFA > CFFA
Asymp. Sig. (2-Tailed)	0.109	**Ties**	0[aj]			aj. FFA = CFFA
ak. Based on negative ranks		**Total**	3			

FFA and help it to exit from the local optimal solution (blue line) and access to the globally optimal solution (red line).

From the above, CFFA showed several advantages, which we mention as follows:

(i) CFFA is a versatile and adaptable strategy for solving a broad variety of optimization issues.

(ii) Due to the combination of the advantages of the CLS and FFA, CFFA has a good solution quality.

(iii) Unlike traditional approaches, CFFA searches across a population of points to find the globally optimal solution.

(iv) Because CFFA only employs objective function information, it can handle any realistic optimization issue, including noncontinues, nonsmooth, and nondifferentiable functions.

(v) Computational trials have demonstrated the superiority of CFFA above those published in the literature where it outperforms other comparison approaches substantially.

(vi) The importance of the CFFA findings was demonstrated using Wilcoxon and Friedman tests.

(vii) Finally, the results of the engineering design problems show that the proposed CFFA is suitable for addressing real-world issues such as problems of cost-effective load transfer, resource allocation, wind farm turbine optimization, unit commitment, and real-time applications.

Finally, without prejudice, the proposed technique CFFA, like previous meta-heuristics algorithms, has the potential drawback of not ensuring an increase in computing speed or accuracy when addressing any optimization problem. Because meta-heuristics methods are random approaches, the CFFA's computational efficacy and solution quality are dependent on the problem's nature and complexity.

5. Conclusion

A chaotic fruit fly algorithm (CFFA) to solve engineering design problems (EDPs) was proposed in this paper. The fruit fly algorithm (FFA), recognized for its resilience and efficacy in addressing optimization problems, was merged with the chaotic local search (CLS) method, which is known for its ability to identify the global optimal solution. CFFA was used in two stages. In the first, FFA was used to get an approximate solution. The optimal solution was then found using chaotic local search (CLS) in the second phase. The proposed approach was tested utilizing a set of CEC'2005 special sessions on actual parameter optimization as well as, three restricted engineering design problems from the most recent test suite, CEC'2020. The experimental outcomes demonstrated the superiority of the proposed technique to finding the global optimal solution and reveal that the suggested CFFA may be utilized to address real-world engineering problems. Furthermore, the convergence curves of the best function values obtained by CFFA before and after CLS showed how CLS disturbed and explored the local region of the approximate solution and how it was utilized to speed convergence, improve solution quality, and find the ideal solution. Finally, the statistical efficiency of the CFFA was investigated by the Friedman test and Wilcoxon signed-rank test, which revealed that the proposed CFFA outperformed other algorithms.

A multi-objective version of CFFA can be developed in future works to solve continuous multi-objective issues. Furthermore, adapting CFFA to a discrete version for handling discrete optimization challenges like the community discovery problem is a promising direction.

Data Availability

All data used to support the findings of this study are included in the article.

Conflicts of Interest

The authors declare that this article′s content has no conflicts of interest.

Acknowledgments

The authors extend their appreciation to the Deputyship for Research and Innovation, Ministry of Education in Saudi Arabia, for funding this research work through the project number (IF-PSAU-2021/01/18049).

References

[1] B.-B. Michael, *Nonlinear Optimization with Engineering Applications*, Springer Optimization and Its Applications, Springer-Verlag, Switzerland, 2008.

[2] M. A. El-Shorbagy, *Hybrid Particle Swarm Algorithm for Multi-Objective Optimization, Master of Engineering Thesis*, Menoufia Univ, Egypt, 2010.

[3] Z. Michalewicz, "Evolutionary computation techniques for nonlinear programming problems," *International Transactions in Operational Research*, vol. 1, no. 2, pp. 223–240, 1994.

[4] T. L. Ayyarao, N. S. S. RamaKrishna, R. M. Elavarasam et al., "War Strategy Optimization Algorithm: A New Effective Metaheuristic Algorithm for Global Optimization," *IEEE Access*, vol. 10, 2022.

[5] I. M. El-Desoky, M. A. El-Shorbagy, S. M. Nasr, Z. M. Hendawy, and A. A. Mousa, "A hybrid genetic algorithm for job shop scheduling problems," *Int. J. Adv. Eng. Technol. Comput. Sci*, vol. 3, no. no. 1, pp. 6–17, 2016.

[6] M. A. El-Shorbagy, A. Y. Ayoub, I. M. El-Desoky, A. A. Mousa, and A. A. Mousa, "A novel genetic algorithm based k-means algorithm for cluster analysis," in *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)*, pp. 92–101, Springer, Cham, NY, USA, 2018.

[7] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[8] D. Simon, *Evolutionary Optimization Algorithms*, John Wiley & Sons, Hoboken, New Jersey, 2013.

[9] M. A. El-Shorbagy and A. E. Hassanien, "Particle swarm optimization from theory to applications," *International Journal of Rough Sets and Data Analysis*, vol. 5, no. 2, pp. 1–24, 2018.

[10] M. A. El-Shorbagy, "Weighted method based trust region-particle swarm optimization for multi-objective optimization," *American Journal of Applied Mathematics*, vol. 3, no. 3, p. 81, 2015.

[11] A. M. Abd Allah and M. A. El-Shorbagy, "Enhanced Particle Swarm Optimization Based Local Search for Reactive Power Compensation Problem," *Applied Mathematics*, vol. 3, 2012.

[12] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, Cambridge, MA, 2004.

[13] W. T. Pan, *Fruit Fly Optimization Algorithm*, Tsang Hai publishing, Taibei, China, 2011.

[14] W. Zhao and L. Wang, "An effective bacterial foraging optimizer for global optimization," *Information Sciences*, vol. 329, pp. 719–735, 2016.

[15] M. Marinaki and Y. Marinakis, "A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands," *Expert Systems with Applications*, vol. 46, pp. 145–163, 2016.

[16] H. Xu, X. Liu, and J. Su, "An improved grey wolf optimizer algorithm integrated with Cuckoo Search," in *Proceedings of the 2017 9th IEEE international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS)*, September 2017.

[17] F. S. Gharehchopogh and H. Gholizadeh, "A comprehensive survey: whale Optimization Algorithm and its applications," *Swarm and Evolutionary Computation*, vol. 48, pp. 1–24, 2019.

[18] M. Elsisy, D. Hammad, and M. El-Shorbagy, "Solving interval quadratic programming problems by using the numerical method and swarm algorithms," *Complexity*, vol. 2020, 11 pages, Article ID 6105952, 2020.

[19] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015.

[20] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.

[21] M. A. El-Shorbagy and A. M. El-Refaey, "COVID-19: mathematical growth vs. precautionary measures in China, KSA, and the USA," *Informatics in Medicine Unlocked*, vol. 28, Article ID 100834, 2022.

[22] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687–697, 2008.

[23] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284, pp. 65–74, 2010.

[24] Y. Zhou, X. Chen, and G. Zhou, "An improved monkey algorithm for a 0-1 knapsack problem," *Applied Soft Computing*, vol. 38, pp. 817–830, 2016.

[25] M. Shehab, A. T. Khader, M. Laouchedi, and O. A. Alomari, "Hybridizing cuckoo search algorithm with bat algorithm for global numerical optimization," *The Journal of Supercomputing*, vol. 75, no. 5, pp. 2395–2422, 2019.

[26] A. Kumar, R. K. Misra, D. Singh, S. Mishra, and S. Das, "The spherical search algorithm for bound-constrained global optimization problems," *Applied Soft Computing*, vol. 85, Article ID 105734, 2019.

[27] E. Cuevas, M. Cienfuegos, D. Zaldívar, and M. Pérez-Cisneros, "A swarm optimization algorithm inspired in the behavior of the social-spider," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6374–6384, 2013.

[28] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: a nature-inspired metaheuristic," *Expert Systems with Applications*, vol. 152, p. 113377, 2020.

[29] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers & Structures*, vol. 169, pp. 1–12, 2016.

[30] A. L. a. Bolaji, M. A. Al-Betar, M. A. Awadallah, A. T. Khader, and L. M. Abualigah, "A comprehensive review: krill Herd algorithm (KH) and its applications," *Applied Soft Computing*, vol. 49, pp. 437–446, 2016.

[31] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, Article ID 113338, 2020.

[32] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: squirrel search algorithm," *Swarm and Evolutionary Computation*, vol. 44, pp. 148–175, 2019.

[33] X.-S. Yang, "Flower pollination algorithm for global optimization," *Unconventional Computation and Natural Computation*, vol. 7445, pp. 240–249, 2012.

[34] W. Zhao, Z. Zhang, and L. Wang, "Manta ray foraging optimization: an effective bio-inspired optimizer for engineering applications," *Engineering Applications of Artificial Intelligence*, vol. 87, Article ID 103300, 2020.

[35] S. Shadravan, H. R. Naji, and V. K. Bardsiri, "The Sailfish Optimizer: a novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 80, pp. 20–34, 2019.

[36] G. Dhiman and V. Kumar, "Emperor penguin optimizer: a bio-inspired algorithm for engineering problems," *Knowledge-Based Systems*, vol. 159, pp. 20–50, 2018.

[37] G. Dhiman and V. Kumar, "Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering

applications," *Advances in Engineering Software*, vol. 114, pp. 48–70, 2017.

[38] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020.

[39] J. Pierezan and L. Dos Santos Coelho, "Coyote optimization algorithm: a new metaheuristic for global optimization problems," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.

[40] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[41] J. Tu, H. Chen, M. Wang, A. H. Gandomi, and Gandomi, "The colony predation algorithm," *Journal of Bionics Engineering*, vol. 18, no. 3, pp. 674–710, 2021.

[42] Y. Zhang and Z. Jin, "Group teaching optimization algorithm: a novel metaheuristic method for solving global optimization problems," *Expert Systems with Applications*, vol. 148, Article ID 113246, 2020.

[43] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, pp. 4661–4667, Singapore, September 2007.

[44] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-Learning-Based Optimization: an optimization method for continuous non-linear large scale problems," *Information Sciences*, vol. 183, no. 1, pp. 1–15, 2012.

[45] A. Husseinzadeh Kashan, "An efficient algorithm for constrained global optimization and application to mechanical engineering design: league championship algorithm (LCA)," *Computer-Aided Design*, vol. 43, no. 12, pp. 1769–1792, 2011.

[46] Q. Askari, I. Younas, and M. Saeed, "Political Optimizer: a novel socio-inspired meta-heuristic for global optimization," *Knowledge-Based Systems*, vol. 195, Article ID 105709, 2020.

[47] S. H. Samareh Moosavi and V. K. Bardsiri, "Poor and rich optimization algorithm: a new human-based and multi populations algorithm," *Engineering Applications of Artificial Intelligence*, vol. 86, pp. 165–181, 2019.

[48] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, "Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts," *Expert Systems with Applications*, vol. 177, Article ID 114864, 2021.

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

[50] P. J. M. van Laarhoven and E. H. L. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications*, pp. 7–15, Springer Netherlands, Dordrecht, 1987.

[51] Anita and A. Yadav, "AEFA: artificial electric field algorithm for global optimization," *Swarm and Evolutionary Computation*, vol. 48, pp. 93–108, 2019.

[52] Y. Abo-Elnaga and M. A. El-Shorbagy, "Multi-sine cosine algorithm for solving nonlinear bilevel programming problems," *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, p. 421, 2020.

[53] M. A. El-Shorbagy, M. A. Farag, A. A. Mousa, and I. M. El-Desoky, "A hybridization of sine cosine algorithm with steady state genetic algorithm for engineering design problems," in *Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications*, A. E. Hassanien, A. Azar, T. Gaber, R. Bhatnagar, and M. F. Tolba, Eds., vol. 921, pp. 1–13, AMLTA 2019, AISC 921, Springer, Berlin, Germany, 2020.

[54] M. H. Tayarani-N and M. R. Akbarzadeh-T, "Magnetic optimization algorithms a new synthesis," in *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 2659–2664, 2008.

[55] M. Ghasemi, I. F. Davoudkhani, E. Akbari, A. Rahimnejad, S. Ghavidel, and L. Li, "A novel and effective optimization algorithm for global optimization and its engineering applications: turbulent Flow of Water-based Optimization (TFWO)," *Engineering Applications of Artificial Intelligence*, vol. 92, Article ID 103666, 2020.

[56] F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, and S. Mirjalili, "Henry gas solubility optimization: a novel physics-based algorithm," *Future Generation Computer Systems*, vol. 101, pp. 646–667, 2019.

[57] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, and W. Al-Atabany, "Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems," *Applied Intelligence*, vol. 51, no. 3, pp. 1531–1551, 2021.

[58] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," *Lecture Notes in Computer Science*, vol. 6145, pp. 355–364, 2010.

[59] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems," *Applied Soft Computing*, vol. 13, no. 5, pp. 2592–2612, 2013.

[60] I. Ahmadianfar, A. A. Heidari, S. Noshadian, H. Chen, A. H. Gandomi, and Gandomi, "INFO: an efficient optimization algorithm based on weighted mean of vectors," *Expert Systems with Applications*, vol. 195, Article ID 116516, 2022.

[61] I. Ahmadianfar, A. A. Heidari, A. H. Gandomi, X. Chu, and H. Chen, "RUN beyond the metaphor: an efficient optimization algorithm based on Runge Kutta method," *Expert Systems with Applications*, vol. 181, Article ID 115079, 2021.

[62] G. Beni and J. Wang, *Swarm Intelligence in Cellular Robotic Systems". Proceed. NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy*, pp. 703–712, Springer, Berlin, Heidelberg, 1993.

[63] G. C. Onwubolu and B. V. Babu, *New Optimization Techniques in Engineering*, p. 141, Springer Science & Business Media, Berlin/Heidelberg, Germany, 2004.

[64] X. Z. Gao, X. Wang, T. Jokinen, S. J. Ovaska, A. Arkkio, and K. Zenger, "A hybrid optimization method for wind generator design," *Int J Innov Comput Inf Control*, vol. 8, pp. 4347–4373, 2012.

[65] S. Khalilpourazari and S. Khalilpourazary, "Optimization of production time in the multi-pass milling process via a Robust Grey Wolf Optimizer," *Neural Computing & Applications*, vol. 29, 2016.

[66] M. Z. Ali, N. H. Awad, P. N. Suganthan, R. M. Duwairi, and R. G. Reynolds, "A novel hybrid Cultural Algorithms framework with trajectory-based search for global numerical optimization," *Information Sciences*, vol. 334-335, pp. 219–249, 2016.

[67] R. Goel and R. Maini, "A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems," *Journal of Computational Science*, vol. 25, pp. 28–37, 2018, þ.

[68] G.-G. Wang, A. H. Gandomi, X. Zhao, and H. C. E. Chu, “Hybridizing harmony search algorithm with cuckoo search for global numerical optimization,” *Soft Computing*, vol. 20, no. 1, pp. 273–285, 2016.

[69] W. F. Abd-El-Wahed, A. A. Mousa, and M. A. El-Shorbagy, “Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems,” *Journal of Computational and Applied Mathematics*, vol. 235, no. 5, pp. 1446–1453, 2011.

[70] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, “An effective krill herd algorithm with migration operator in biogeography-based optimization,” *Applied Mathematical Modelling*, vol. 38, no. 9-10, pp. 2454–2462, 2014b.

[71] V. I. Skoullis, I. x. Tassopoulos, and G. N. Beligiannis, “Solving the high school timetabling problem using a hybrid cat swarm optimization based algorithm,” *Applied Soft Computing*, vol. 52, pp. 277–289, 2017.

[72] C. Liu and L. Fan, “A hybrid evolutionary algorithm based on tissue membrane systems and CMA-ES for solving numerical optimization problems,” *Knowledge-Based Systems*, vol. 105, pp. 38–47, 2016.

[73] M. A. El-Shorbagy and A. Y. Ayoub, “Integrating grasshopper optimization algorithm with local search for solving data clustering problems,” *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, pp. 783–793, 2021.

[74] G.-G. Wang, A. H. Gandomi, A. H. Alavi, and G.-S. Hao, “Hybrid krill herd algorithm with differential evolution for global numerical optimization,” *Neural Computing & Applications*, vol. 25, no. 2, pp. 297–308, 2014.

[75] M. A. El-Shorbagy and A. M. El-Refaey, “Hybridization of grasshopper optimization algorithm with genetic algorithm for solving system of non-linear equations,” *IEEE Access*, vol. 8, pp. 220944–220961, 2020.

[76] G. Wang and L. Guo, “A novel hybrid bat algorithm with harmony search for global numerical optimization,” *Journal of Applied Mathematics*, vol. 2013, 21 pages, Article ID 696491, 2013.

[77] H. Garg, “Solving structural engineering design optimization problems using an artificial bee colony algorithm,” *Journal of Industrial and Management Optimization*, vol. 10, no. 3, pp. 777–794, 2014.

[78] E. Zahara and Y.-T. Kao, “Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems,” *Expert Systems with Applications*, vol. 36, no. 2, pp. 3880–3886, 2009.

[79] L. d. S. Coelho, “Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems,” *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, 2010.

[80] H. Chen, Y. Xu, M. Wang, and X. Zhao, “A balanced whale optimization algorithm for constrained engineering design problems,” *Applied Mathematical Modelling*, vol. 71, pp. 45–59, 2019.

[81] R. Tao, Z. Meng, and H. Zhou, “A self-adaptive strategy based firefly algorithm for constrained engineering design problems,” *Applied Soft Computing*, vol. 107, Article ID 107417, 2021.

[82] Y. Belkourchia, L. Azrar, and M. Z. Es-Sadek, “A hybrid optimization algorithm for solving constrained engineering design problems,” in *Proceedings of the 2019 5th International Conference on Optimization and Applications (ICOA)*, pp. 1–7, IEEE, Kenitra, Morocco, April 2019.

[83] S. Gupta, K. Deep, H. Moayedi, L. K. Foong, and A. Assad, “Sine cosine grey wolf optimizer to solve engineering design problems,” *Engineering with Computers*, vol. 37, no. 4, pp. 3123–3149, 2021.

[84] Y. Li, X. Zhu, and J. Liu, “An improved moth-flame optimization algorithm for engineering problems,” *Symmetry*, vol. 12, no. 8, p. 1234, 2020.

[85] C. Xiao, K. Hao, and Y. Ding, “An improved fruit fly optimization algorithm inspired from cell communication mechanism,” *Mathematical Problems in Engineering*, vol. 2015, 15 pages, Article ID 492195, 2015.

[86] H. Chengzhong and L. Junying, “Adaptive chaos fruit fly optimization algorithm,” *Journal of Computer Applications*, vol. 33, no. 05, p. 1313, 2013.

[87] A. A. Mousa, M. A. El-Shorbagy, I. Mustafa, and H. Alotaibi, “Chaotic search based equilibrium optimizer for dealing with nonlinear programming and petrochemical application,” *Processes*, vol. 9, no. 2, 2021.

[88] M. Mitić, N. Vuković, M. Petrović, and Z. Miljković, “Chaotic fruit fly optimization algorithm,” *Knowledge-Based Systems*, vol. 89, pp. 446–458, 2015.

[89] M.-Y. Cheng and K.-Yu Huang, “Genetic algorithm-based chaos clustering approach for nonlinear optimization,” *Journal of Marine Science and Technology*, vol. 18, no. 3, p. 15, 2010.

[90] X. Wang, Z. Wang, J. Weng, C. Wen, H. Chen, and X. Wang, “A new effective machine learning framework for sepsis diagnosis,” *IEEE Access*, vol. 6, Article ID 48300, 2018.

[91] S. Gao, Yu Yang, Y. Wang, J. Wang, J. Cheng, and M. C. Zhou, “Chaotic local search-based differential evolution algorithms for optimization,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 6, pp. 3954–3967, 2019.

[92] X. Zhang, Y. Xu, C. Yu et al., “Gaussian mutational chaotic fruit fly-built optimization and feature selection,” *Expert Systems with Applications*, vol. 141, Article ID 112976, 2020.

[93] A. Baykasoğlu and B. O. Fehmi, “Adaptive firefly algorithm with chaos for mechanical design optimization problems,” *Applied Soft Computing*, vol. 36, pp. 152–164, 2015.

[94] M. A. El-Shorbagy, A. A. Mousa, and S. M. Nasr, “A chaos-based evolutionary algorithm for general nonlinear programming problems,” *Chaos, Solitons & Fractals*, vol. 85, pp. 8–21, 2016.

[95] F. Ye, X. Y. Lou, and L. F. Sun, “An improved chaotic fruit fly optimization based on a mutation strategy for simultaneous feature selection and parameter optimization for SVM and its applications,” *PLoS One*, vol. 12, no. 4, Article ID e0173516, 2017.

[96] A. A. Mousa, M. A. El-Shorbagy, and M. A. Farag, “Steady-state sine cosine genetic algorithm based chaotic search for nonlinear programming and engineering applications,” *IEEE Access*, vol. 8, Article ID 212036, 2020.

[97] X. Yuan, Y. Liu, Y. Xiang, and X. Yan, “Parameter identification of BIPT system using chaotic-enhanced fruit fly optimization algorithm,” *Applied Mathematics and Computation*, vol. 268, pp. 1267–1281, 2015.

[98] S. S. Rao, *Engineering Optimization: Theory and Practice*, A Wiley-Interscience publication, New Jersey, fourth edition, 2009.

[99] W. T. Pan, “A new fruit fly optimization algorithm: taking the financial distress model as an example,” *Knowl.-Based Syst.*vol. 26, pp. 69–74, 2013.

[100] D. Yang, G. Li, and G. Cheng, “On the efficiency of chaos optimization algorithms for global optimization,” *Chaos, Solitons & Fractals*, vol. 34, no. 4, pp. 1366–1375, 2007.

[101] M. A. E. Shorbagy and A. A. Mousa, "Chaotic particle swarm optimization for imprecise combined economic and emission dispatch problem," *Review of Information Engineering and Applications*, vol. 4, no. 1, pp. 20–35, 2017.

[102] B. Alatas, E. Akin, and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons & Fractals*, vol. 40, no. 4, pp. 1715–1734, 2009.

[103] M. Jampour, "Chaotic genetic algorithm based on lorenz chaotic system for optimization problems," *Intelligent Systems and Applications*, vol. 5, no. 5, pp. 19–24, 2013.

[104] J. Xiao, "Improved quantum evolutionary algorithm combined with chaos and its application," *Advances in Neural Networks - ISNN 2009*, vol. 5553, pp. 704–713, 2009.

[105] P. Lu, J. Zhou, H. Zhang, R. Zhang, and C. Wang, "Chaotic differential bee colony optimization algorithm for dynamic economic dispatch problem with valve-point effects," *International Journal of Electrical Power & Energy Systems*, vol. 62, pp. 130–143, 2014.

[106] L. d. S. Coelho, H. V. H. Ayala, and V. C. Mariani, "A self-adaptive chaotic differential evolution algorithm using gamma distribution for unconstrained global optimization," *Applied Mathematics and Computation*, vol. 234, pp. 452–459, 2014.

[107] D. Oliva, M. Abd El Aziz, and A. Ella Hassanien, "Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm," *Applied Energy*, vol. 200, pp. 141–154, 2017.

[108] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[109] B. Alatas, "Chaotic harmony search algorithms," *Applied Mathematics and Computation*, vol. 216, no. 9, pp. 2687–2699, 2010.

[110] Z. Aram, S. Jafari, J. Ma, J. C. Sprott, S. Zendehrouh, and V.-T. Pham, "Using chaotic artificial neural networks to model memory in the brain," *Communications in Nonlinear Science and Numerical Simulation*, vol. 44, pp. 449–459, 2017.

[111] A. A. Mousa, M. A. El-Shorbagy, W. Abd-El-Wahed, and F. Abd-El-Wahed, "Local search based hybrid particle swarm optimization algorithm for multiobjective optimization," *Swarm and Evolutionary Computation*, vol. 3, pp. 1–14, 2012.

[112] P. Suganthan, N. Hansen, J. Liang et al., "Problem Definitions and Evaluation Criteria for the CEC'2005 Special Sessionon Real Parameter Optimization," Nanyang Technological University, Tech. Rep, 2005, http://www.ntu.edu.sg/home/epnsugan/index_files/cec-%2005/Tech-Report-May-30-05.pdf.

[113] A. Kumar, G. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, and S. Das, "A test-suite of non-convex constrained optimization problems from the real-world and some baseline results," *Swarm and Evolutionary Computation*, vol. 56, Article ID 100693, 2020.

[114] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[115] R. Chelouah and P. Siarry, "A continuous genetic algorithm designed for the global optimization of multimodal functions," *Journal of Heuristics*, vol. 6, no. 2, pp. 191–213, 2000.

[116] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[117] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[118] Z. Li, Y. Zhou, S. Zhang, and J. Song, "Lévy-flight moth-flame algorithm for function optimization and engineering design problems," *Mathematical Problems in Engineering*, vol. 2016, 22 pages, Article ID 1423930, 2016.

[119] S. Khalilpourazari and S. Khalilpourazary, "An efficient hybrid algorithm based on Water Cycle and Moth-Flame Optimization algorithms for solving numerical and constrained engineering optimization problems," *Soft Computing*, vol. 23, no. 5, pp. 1699–1722, 2019.

[120] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, Article ID 113609, 2021.

[121] C. Chen, X. Wang, H. Yu, M. Wang, and H. Chen, "Dealing with multi-modality using synthesis of Moth-flame optimizer with sine cosine mechanisms," *Mathematics and Computers in Simulation*, vol. 188, pp. 291–318, 2021.

[122] M. H. Nadimi-Shahraki, A. Fatahi, H. Zamani, S. Mirjalili, and L. Abualigah, "An improved moth-flame optimization algorithm with adaptation mechanism to solve numerical and mechanical engineering problems," *Entropy*, vol. 23, no. 12, p. 1637, 2021.

[123] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the 4th IEEE International Conference on Neural Networks*, Piscataway, New Jersey, December 1995.

[124] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, pp. 1769–1776, Edinburgh, UK, September 2005.

[125] L. J. Eshelman, "The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination," in *Foundations of Genetic Algorithms, Morgan Kaufmann*, G. J. E. Rawlins, Ed., pp. 265–283, San Mateo, California, 1991.

[126] C. Fernandes and A. Rosa, "A study of non-random matching and varying population size in genetic algorithm using a royal road function," in *Proceedings of the 2001 Congress on Evolutionary Computation*, pp. 60–66, Piscataway, New Jersey, 2001.

[127] F. Herrera, M. Lozano, and D. Molina, "Continuous scatter search: an analysis of the integration of some combination methods and improvement strategies," *European Journal of Operational Research*, vol. 169, no. 2, pp. 450–476, 2006.

[128] M. Laguna, R. Marti, and R. C Martí, *Scatter Search. Methodology and Implementation in C*, Springer Science & Business Media, Berlin, Germany, 2003.

[129] K. V. Price, M. Rainer, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, NY, USA, 2005.

[130] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, vol. 2pp. 1785–1791, Edinburgh, UK, September 2005.

[131] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[132] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power," *Information Sciences*, vol. 180, no. 10, 2010.