scientific reports

Check for updates

OPEN

Chameleon swarm algorithm with Morlet wavelet mutation for superior optimization performance

Vipan Kusla[1], Gurbinder Singh Brar[2]✉, Harpreet Kaur[2], Ramandeep Sandhu[2], Chander Prabha[3], Md. Mehedi Hassan[4]✉, Shahab Abdulla[5], Md Rittique Alam[6], Samah Alshathri[7] & Walid El-Shafai[8,9]✉

Metaheuristic algorithms play a vital role in addressing a wide range of real-world problems by overcoming hardware and computational constraints. The Chameleon Swarm Algorithm (CSA) is a modern metaheuristic algorithm that uses how chameleons act. To improve the capabilities of the CSA, this work proposes a modified version of the Chameleon Swarm Algorithm to find better optimal solutions applicable to various application areas. The effectiveness of the proposed algorithm is assessed using 97 typical benchmark functions and three real-world engineering design problems. To validate the efficacy of the proposed algorithm, it has been compared to a number of well-known and widely-used classical algorithms, the Gravitational Search Algorithm, the Earthworm Optimization. The proposed modified Chameleon Swarm Algorithm using Morlet wavelet mutation and Lévy flight (mCSAMWL) is superior to existing algorithms for both unimodal and multimodal functions, as demonstrated by Friedman's mean rank test as well as three real world engineering design problems. Five performance metrics—average energy consumption, total energy consumption, total residual energy, dead node and cluster head frequency are taken into consideration when evaluating the performances against state-of-the-art algorithms. For nine different simulation scenarios, the proposed algorithm mCSAMWL outperforms the Atom Search Optimization (ASO), Hybrid Particle Swarm Optimization and Grey Wolf Optimization (PSO-GWO), Bald Eagle Search Algorithm (BES), the African Vulture Optimization Algorithm (AVOA), and the Chameleon Swarm Algorithm (CSA) in terms of average energy consumption and total energy consumption by 50.9%, 52.6%, 45%, 42.4%, 50.1% and 51.4%, 53.3%, 45.6%, 42.4%, 50.7%.

Keywords Morlet wavelet, Lévy flight, Benchmark functions, Wireless sensor network (WSN), Cluster head

Metaheuristic algorithms are effective solutions that can be utilized for a wide variety of engineering challenges that are encountered in the real world[1]. Compared to deterministic approaches, metaheuristic algorithms have excelled in recent decades due to their adaptability, ability to prevent local optima, and gradient-free framework. Deterministic approaches get identical results for the same problem. This behavior may result in local optimum trapping, which is a drawback of deterministic techniques[2]. Local optima trapping denotes an algorithm becoming stuck in local solutions. As a result, it is unable to find a global solution. Because of their inconsistent performance, deterministic methods can no longer be relied upon for solving practical optimization problems with several possible solutions. Most of these algorithms are derived from observations of natural phenomena, such as the intelligence of swarms of particles, the logic of biologically inspired algorithms, the physics of

[1]Department of Computer Science and Engineering, Sant Longowal Institute of Engineering and Technology, Longowal, Sangrur, Punjab 148106, India. [2]School of Computer Science & Engineering, Lovely Professional University, Jalandhar, Punjab 144411, India. [3]Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab 140401, India. [4]Computer Science and Engineering Discipline, Khulna University, Khulna 9208, Bangladesh. [5]UniSQ College, University of Southern Queensland, Toowoomba, QLD 4305, Australia. [6]Department of Computer Science, American International University-Bangladesh, Dhaka 1229, Bangladesh. [7]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia. [8]Automated Systems and Soft Computing Lab (ASSCL), Computer Science Department, Prince Sultan University, Riyadh 11586, Saudi Arabia. [9]Department of Electronics and Electrical Communications Engineering, Faculty of Electronic Engineering , Menoufia University, Menouf 32952, Egypt. ✉email: maatibrar@gmail.com; mehedihassan@ieee.org; welshafai@psu.edu.sa; eng.waled.elshafai@gmail.com; walid.elshafai@el-eng.menofia.edu.eg

everyday objects, etc. Evolutionary algorithms mimic nature. The fittest candidate survives in evolutionary algorithms. These algorithms start with a population of solutions surviving in a fitness-evaluated environment. Then, genetic crossover and mutation help the parent population pass on its environmental adaptations to the offspring. Finally, iterative generations are used to find the best environmental solutions. Genetic Algorithm, Biogeography-Based Optimization (BBO), Genetic Programming (GP), Differential Evolution (DE), and Evolution Strategies (ES), etc., are the evolutionary algorithms.

The notions and norms of physics are adhered to in physics-based procedures, in which person update their positions according to physical laws such as molecule dynamics, the force of inertia, the force of gravitation, etc., the Atom Search Optimization, the Simulated Annealing, the Artificial Electric Field Algorithm, and the Sine Cosine Optimization, etc. are all well-known methods based on physical principles. Natural metaheuristics inspired by the "collective intelligence" of swarms are referred to as "swarm intelligence". Collective knowledge is developed when a group of similar agents collaborate and learn from their surroundings. Colonies of ants, swarms of bees, dense flocks of birds, and many other groups of animals have been used as examples of collective intelligence. The coordinated flocking of birds inspired the concept of Particle Swarm Optimization. Fireflies' flashing habits inspired the firefly algorithm's development. The Bat Algorithm (BA) is a nature inspired algorithm that employs a sophisticated echolocation-based navigation system. The Ant Colony Optimization was based on the way that real-life ant colonies lay down pheromone trails. Cuckoo Search (CS) is an evolutionary optimization method inspired by the behavioral patterns of the cuckoo bird. Among the most prominent are: Fruit Fly Optimization Algorithm (FFOA), the Ant Colony Optimization (ACO), the Grasshopper Optimization Algorithm (GOA), the Salp Swarm Algorithm (SSA), the Whale Optimization Algorithm (WOA), the African Vulture Optimization Algorithm (AVOA), the Glowworm Swarm Optimization (GSO), and Cat Swarm Optimization (CSO) etc. Existing CSA studies have several problems, such as insufficient diversity, local optima problems, and imbalanced exploitation. Each previously mentioned optimization algorithm must also consider how to best explore and exploit a given search space. Exploration and exploitation[3] are the two distinct stages that make up the search process for an algorithm that is based on a population. The term "exploration" refers to the process of increasing the number of swarms in order to more thoroughly study every part of the search space, whereas the term "exploitation" describes the process of increasing the number of swarms to more thoroughly analyze any promising or intriguing locations that were discovered during the exploration phase. Stochastic behaviour makes it difficult to achieve equilibrium among the exploration and exploitation phases.

An effective optimization algorithm will strike a balance between exploring and exploiting the space. It is not guaranteed that an algorithm will be superior on all problems just because it performs well on some problems. It serves as inspiration for this research. The Chameleon Swarm Algorithm (CSA) is vulnerable to becoming trapped in local optima. The optimizer may be unable to locate the global solution because it is trapped in the local region. Generating new solutions is estimated using the solutions inherent to the previous iteration. Consequently, this may reduce the algorithm's convergence rate, resulting in solutions that do not effectively encompass the entire search space and premature convergence. Considering this as a motivation, the mCSAMWL algorithm is proposed in this study as an improved version of the CSA, to increase the search capability, optimize the balance between exploitation and exploration phases, and prevent early convergence of the local optimum. mCSAMWL's guiding principle is founded on the injection of two effective strategies into the original CSA: Morlet Wavelet mutation, and the Step Reducer Lévy flight. Swarm intelligence is the foundation of the Chameleon Swarm Algorithm (CSA). Recent advancements in optimization algorithms, particularly those utilizing Levy-based search techniques, have led to the development of several innovative approaches in the field such as the modified version of Dynamic Hunting Leadership (DHL) algorithm, have incorporated the Levy Flight technique to enhance convergence and solution precision. The mDHL algorithm[4], which also addresses the challenges of local optima and convergence delays, integrates this technique with localized development strategies to improve global exploration and exploitation. The DGS-SCSO optimizer[5], an enhanced version of Sand Cat Swarm Optimization (SCSO), incorporates Dynamic Pinhole Imaging and the Golden Sine Algorithm to mitigate issues like local optima entrapment and slow convergence. Similarly, AEFA-CSR[6], a hybrid of the Artificial Electric Field Algorithm with Cuckoo Search and Refraction Learning, improves convergence and solution precision, showing superior performance across benchmark functions and engineering problems.

Related work

CSA has attracted significant attention from researchers due to its simple architecture and ease of implementation. To further enhance its functionality, numerous concepts and approaches have been introduced. This section offers an overview of CSA's development and examines its applications in solving challenges across various domains. Sridharan[7] developed a Chameleon Swarm Optimization (CSO) with machine learning- based Sarcasm Detection and Classification (CSOML-SASC) model. Umamageswari et al.[8] introduced a framework using the Fuzzy C-Means (FCM) based Chameleon Swarm Algorithm (CSA) named FCM-CSA, which was used for plant leaf diseased part segmentation. RizkAllah and Hameed[9] suggested a Chameleon Swarm Algorithm (MCSA) that extracts parameters from solid oxide fuel cell models using a semi-empirical and memory-based approach. Anitha et al.[10] introduced a Modified Grey Wolf-based Chameleon Swarm Algorithm to minimize energy consumption and enable secure wireless sensor network communication. RizkAllah et al.[11] introduced a hybrid approach comprising the Chameleon Swarm Algorithm(CSA) and Mayfly Optimization (MO) named CSMO for solving the Combined Heat and Power Economic Dispatch (CHPED) problem. Mostafa et al.[12] proposed a modified mCSA algorithm that incorporates an Artificial Ecosystem-Based Optimization (AEO) consumption operator. Using a multi-objective chameleon swarm optimization algorithm and an advanced feature-selection method, Wang et al.[13] introduced a short term wind speed forecasting system. A Multi strategy Chameleon Swarm Algorithm called (MCSA) was developed by Hu et al.[14] using a Crossover-based Comprehensive Learning (CCL) strategy incorporating sinusoidal parameter tuning and fractional-order calculus. RMSCSA, which was based on the Refraction Mirror Learning (RML) method to promote variety and segmental variation of population diversity using S-type weight, was presented by Damin et al.[15]. To handle non-convex Economic Load Dispatch (ELD) problems, an Enhanced Chameleon Swarm Algorithm (ECSA) was developed by Braik et al.[16] that combines roulette wheel selection with Lévy flight approaches. A hybrid variant of CSA named CCECSA was suggested by Hu et al.[17] in which mutation operations and elite guidance strategies were used. Also, CSA incorporated the horizontal and vertical crossovers of CSO to solve the disc Wang-Ball curve (DWB) reduced-degree optimization models. Sun et al.[18] introduced an improved Chameleon Swarm Algorithm called CLCSA-LSTM enhanced by using the Somersault Foraging Technique of the Manta Ray Foraging Optimization algorithm (MRFO), a boundary neighborhood updating method to maintain a demographically diversified population. It initially optimizes LSTM network hyper parameters and finds the optimal ones to tackle the manual tuning process and the insufficient stability problem. This model was used to recognize OFDM signals after being trained with the aforementioned hyper parameters. Zhou and Xu[19] find that the optimal size of each component is determined based on the actual local hourly weather data and the load demand over the course of a year using the Chameleon swarm algorithm (CSA) for the framework of the renewable micro-grid system. Dinh[20] used CSA to build an algorithm that enhanced the image and synthesized the high-frequency layer. Table 1 gives an overview of the most recently proposed modifications that have been suggested for the CSA algorithm.

New advancements in optimization have led to the development of hybrid techniques aimed at improving performance and robustness. Abed-alguni introduced[21] island-based Cuckoo Search (iCSPM) algorithm which improves population diversity and exploration by integrating an island model and replacing Levy flight with polynomial mutation, outperforming other methods in accuracy and reliability across standard benchmarks. The iCSPM2 algorithm introduced by Abed-alguni & Paul[22] further enhances iCSPM by incorporating Elite Opposition-based Learning and multiple mutation strategies, such as HDP, Jaya, and pitch adjustment, achieving better accuracy, convergence, and surpassing four well-known swarm optimization algorithms in benchmark tests. The Exploratory Cuckoo Search (ECS) proposed by Abed-alguni et al.[23] improves the original Cuckoo Search by integrating refraction learning, Gaussian perturbation, and multiple mutation methods, outshining traditional CS variations in 14 benchmark functions and exhibiting competitive performance compared to six renowned swarm optimization algorithms. Similarly, the Improved Salp Swarm Algorithm (ISSA)[24] introduced

References	Year	Methodology proposed	Key findings
Sridharan[7]	2021	Chameleon Swarm Optimization with machine learning using Sarcasm Detection and Classification (CSOML-SASC)	Used to improves the overall classification performance for Sentimental Analysis and Sarcasm Detection No change in CSA algorithm
Umamageswari et al.[8]	2021	Fuzzy C-Mean Based Chameleon Swarm Algorithm (FCM-CSA)	Used for segmentation of plant leaf disease To overcome the short coming of Fuzzy C Mean No change in the CSA algorithm
RizkAllah et al.[9]	2021	Semi-Empirical and Memory-based Chameleon Swarm Algorithm (MCSA)	Used for extraction of Solid Oxide Fuel Cell Models parameters. Keeping record of best solution in prior stage using internal memory No changes in the CSA algorithm
Mostafa et al.[12]	2022	Improved Chameleon Swarm Algorithm using Artificial Ecosystem-based Optimisation (AEO) consumption operator	Used as feature selection algorithm for Breast Cancer Diagnosis. Non-linear transfer operator, Lévy flight control parameter and Consumption operator of AEO algorithm are used in the CSA algorithm
Anitha et al.[10]	2022	A modified Grey Wolf-based Chameleon Swarm Algorithm	Used for selection of cluster head (CH) nodes from WSN Combination of improved Grey Wolf Optimizer and Chameleon Swarm Algorithm
RizkAllah et al.[11]	2022	Hybridization of Chameleon Swarm Algorithm(CSA) and Mayfly Optimization (MO) known as CSMO	Used to resolve the Combined Heat and Power Economic Dispatch (CHPED) problem Mayfly Optimization (MO) is used with CSA algorithm.
Wang et al.[13]	2023	A Multi-Objective Chameleon Swarm Optimization Algorithm using advanced feature selection method	Developed wind speed forecasting system Uses advanced feature selection method No changes were proposed in the CSA algorithm
Hu et al.[14]	2023	Enhanced Chameleon Swarm Algorithm (MCSA)	Used for two truss topology optimization problems Incorporation of Crossover-based Comprehensive Learning (CCL) strategy in CSA Algorithm Sinusoidal parameter adjustment and Fractional-order calculus, are used in this algorithm.
Damin et al.[15]	2023	Refraction Mirror Learning (RML) using S-type weight-based Chameleon Swarm Algorithm (RMSCSA)	Refraction Mirror Learning Strategy is introduced along with S-type weight
Braik et al.[16]	2023	Enhanced Chameleon Swarm Algorithm (ECSA)	Applied to address non-convex Economic Load Dispatch (ELD) problems Roulette Wheel Selection method and Lévy flight operator are incorporated in the CSA algorithm
Hu et al.[17]	2023	A hybridization of CSA and Crisscross Optimization (CCECSA Algorithm)	Designed to tackle Disk Wang–Ball (DWB) curve degree reduction problem Elite Guidance Strategy, Crisscross Optimization Algorithm, and Competitive Substitution Mechanism are added in CSA algorithm.
Sun et al.[18]	2023	An Improved Chameleon Swarm Algorithm (CLCSA-LSTM) using Manta Ray Foraging Optimization algorithm (MRFO)Somersault Foraging Technique	Applied to enhance the Long Short-Term Memory (LSTM) network Lens-Imaging Learning and Coupling variation are introduced in CSA algorithm
Zhou and Xu[19]	2023	Used existing Chameleon Swarm Algorithm (CSA)	Used to solve hybrid micro-grid design problem Increase the use of inexpensive, locally available renewable resources.
Dinh[20]	2023	Used existing Chameleon Swarm Algorithm (CSA)	Used for Medical image enhancement as well as Image fusion model No change in CSA algorithm

Table 1. Modifications suggested for the CSA algorithm.

by Abed-alguni et al. boosts the SSA's optimization capabilities with Gaussian Perturbation, highly disruptive polynomial mutation, Laplace crossover, and Mixed Opposition-based Learning, outperforming other SSA variants and 18 top optimization algorithms in solving single-objective continuous optimization problems.

From the above table, it can be deduced that a few studies have modified the standard CSA algorithm by introducing various techniques to solve some engineering problems, but it still has scope to improve further. Therefore, to improve the original algorithm's flaws, we proposed an improved Chameleon Swarm Algorithm using Morlet wavelet mutation and Lévy flight, named mCSAMWL.

The primary contributions of this paper are as follows:

- A modified CSA (Chameleon Swarm Algorithm) is proposed, developed and applied that combines the features of the Morlet Wavelet mutation and Lévy flight methods to keep the balance between searching capabilities by preventing the local optimal solution and slow convergence problems.
- The performance of the proposed mCSAMWL algorithm is established by evaluating it using 68 unimodal and multimodal benchmark test functions, CEC 2017 test suite functions and three real-world engineering design problems.
- A Clustering technique is implemented using the above metaheuristic algorithm, proving its efficiency.

The remainder of this paper is organized as follows: Sect. 2 reviews related work; Sect. 3 describes the materials and methods; Sect. 4 presents an empirical evaluation using 97 benchmark functions; Sect. 5 discusses a real-world engineering design problem; Sect. 6 presents results of the proposed mCSAMWL algorithm for balanced clustering in WSNs; and Sect. 7 concludes the paper.

Materials and methods

In 2021, Braik[25] introduced a meta-heuristic algorithm, the Chameleon Swarm Algorithm (CSA). This algorithm is based on the way a chameleon hunts and searches for food. Chameleons are a distinct species of animal due to their ability to adapt to their environment. Chameleons eat insects and can survive in alpine, lowland, desert, and semi-arid environments. Chameleons search for food through a series of processes, including locating their target, tracking it with their eyes, and finally attacking it. This section explains how to model this algorithm mathematically.

Initialization and function assessment

CSA begins the optimization process with the initial population as a population-based algorithm. In a d -dimensional search space, a population of n chameleons, where each chameleon represents a potential solution to a problem, can be represented by a z -matrix of size $n \times d$. As demonstrated below, a vector can be used to describe where chameleon h is in the search domain at each iteration itr.

$$z_{itr}^{h} = \left[z_{itr,1}^{h}, z_{itr,2}^{h}, z_{itr,3}^{h}, \ldots .z_{itr,d}^{h}\right] \tag{1}$$

where, $h = 1,2,3\ldots .n.itr$ denotes the count of iterations. d denotes the problem dimension and $z_{itr,d}^{h}$ denotes the h^{th}chameleon's location. Using a uniform random initialization method, the search space's initial population is generated while considering the problem's dimensions and the number of chameleons, which is shown in Eq. (2).

$$z^{h} = lb + rnd\left(ub_{m} - lb_{m}\right) \tag{2}$$

where, z^{h} denotes h^{th}chameleon initial vector, rnd is a random value between 0 and 1 and ub and lb denotes search area's upper and lower boundaries in m^{th} dimension. At each stage, the effectiveness of the solution is evaluated with the help of the objective function.

Search of the prey

The position update approach put forth below can be used to represent the chameleon's movement while foraging mathematically as in Eq. (3).

$$z_{itr+1}^{h,m} = z_{itr}^{h,m} + B_{1}\left(B_{itr}^{h,m} - G_{itr}^{m}\right)rnd_{1} + B_{2}\left(G_{itr}^{m} - z_{itr}^{h,m}\right)rnd_{2}, rnd^{itr} \geq B_{t} \tag{3}$$

where, $z_{itr+1}^{h,m}$ is the h^{th}chameleon's new position in the m^{th}dimension in the $(itr+1)^{th}$ iteration, itr and $(itr+1)$ represent the itr^{th}iteration count and $itr+1^{th}$ iteration count respectively. h and d are the d^{th} dimensionof h^{th} chameleon. $z_{i}^{h,m}$ represents the current position. The best and global best are $B_{itr}^{h,m}$ and $G_{itr}^{h,m}$. B_{1} and B_{2} govern the ability of exploration. rnd_{1}, rnd_{2} and rnd_{3} denotes random uniform numbers between 0 and 1. rnd^{itr} is an index-based random number and B_{t} is the chameleon's prey-recognition probability.

Chameleon's eyes rotation

Chameleons have a characteristic in their eyes that allows them to rotate at 360°, allowing them to see in all directions, and monitor their prey's presence. As a result, the position of each chameleon is adjusted so that it corresponds with this function as shown below:

$$z_{itr+1}^{h} = zrc_{itr}^{h} + \overline{z}_{itr}^{h} \tag{4}$$

where, z_{itr+1}^{h} the position after rotation, and z_{itr}^{h} is the current location prior to rotation. zrc_{i}^{h} denotes the chameleon's search space rotational coordinates, as illustrated in Eq. (5).

$$zrc_{itr}^{h} = rm \times \ zc_{itr}^{h} \tag{5}$$

where, rm represents the rotational matrix which shows the rotation of chameleons, zrc_{itr}^{h} denotes centering coordinates at iteration itr.

Hunting prey

Chameleons complete their hunts by ambushing their prey when the target predator gets too close to them. The chameleon that can approach its prey the most successfully is considered the best of the group, and optimal. This chameleon attacks prey using its tongue. As a result of being capable of extending its tongue twice as much, its position has been revised slightly. This permits chameleons to take advantage of the search space and catch their prey, which is mathematically described as follows:

$$V_{itr+1}^{h,m} = \omega \, V_{itr}^{h,m} + c_1 (G_{itr}^{m} - z_{itr}^{h,m}) rnd_1 + c_2 \left(B_{itr}^{h,m} - z_{itr}^{h,m} \right) rnd_2 \tag{6}$$

where, $V_{itr+1}^{h,m}$ is the h^{th}chameleon's new speed in the m^{th}dimension of iteration, $V_{itr}^{h,m}$ is h^{th}chameleon's current speed in the d^{th}dimension. The i^{th}chameleon's current location is denoted by $pop_{i}^{h,m}$ and the effects of $z_{itr}^{h,m}$ and G_{itr}^{m} on its tongue are regulated by two positive constant integers, c_1 and c_2. Here, $rnd_1$and rnd_2 are two arbitrary numbers chosen from the range $0-1$, and the inertia weight, denoted by ω , decreases linearly with every successive generation, as shown in the below formula.

$$\omega \ = \left(1 - {itr}\Big/{\max\_itr} \right)^{(\rho \sqrt{itr/\max\_itr}} \tag{7}$$

where, itr denotes the present iteration, $\max\_\mathrm{itr}$ denotes the total number of iterations and positive variable ρ controls the exploitation capacity. The CSA algorithm demonstrates how the chameleon's initial positions in the search space are created at random as an integral component of the optimization process. Equation (3) is used to update the chameleons' positions in each iteration cycle. If a chameleon escapes the search region, the simulated procedures specified for CSA will be used to bring it back to the boundary. Next, a fitness function is used to determine which chameleon is the most fit after each iteration. The best position of a chameleon in its pursuit of prey is known as the fittest solution. Following the initialization step, Algorithm 1 iterates through the remaining steps until the maximum criteria is reached. According to a swarm behavior model created by the CSA, chameleons constantly hunt for and take advantage of both fixed and moving prey in their environment before moving in to capture it. The optimization potential of the CSA should be displayed via its mathematical models.

Chameleon swarm optimization

The swarm-based metaheuristic Chameleon swarm algorithm (CSA)[25] was proposed by Braik in 2021.CSA mimic chameleon hunting and food finding. This method is based on how the chameleon hunts and looks for food. Chameleon food hunting involves several processes, including prey tracking, chasing the prey with their sight, and quickly attacking the prey with their long, sticky tongue. The fact that it is easy to operate and has a limited number of adjustment parameters are two of its positives; nonetheless, it is not very effective in resolving high-mode or multi-mode issues. The conventional CSA algorithm also suffers from insufficient population diversity, a sluggish convergence rate, and a low degree of precision. For this reason, a new modified Chameleon Swarm Algorithm incorporating the Morlet Wavelett mutation and Lévy Flight factor (mCSAMWL) is proposed. Finally, a modified Chameleon Swarm Algorithm is employed to tackle global optimization problems. The effectiveness of the algorithm has been measured against 68 benchmark test functions and three real world engineering design problems.

Concept of Morlet wavelet mutation

A physicist, Morlet, came up with the term "Morlet Wavelet" when he examined a seismic signal that had been transformed by a cosine function[26].Wavelet mutation is used to improve algorithm stability. In addition, wavelet mutation operations exhibit a fine-tuning ability. The CSA is vulnerable to becoming ensnared in local optima, preventing the algorithm from exploring the complete search space. In this work, Morlet wavelet mutation is used to enhance the exploration stage, the accuracy of the search, and the stability of solutions. The straightforward mutation approach does not easily solve the stagnation phenomenon. The key to this advancement is figuring out how to enhance the conventional mutation approach so that it can overcome the local optimum. Wavelet mutation uses the wavelet function's translation and expansion capabilities to look for other solutions in a feasible space that are close to the ones already known to be correct for a set of persons. To further fine-tune the mutation range with each iteration change, the wavelet function's stretching parameters can be modified to decrease the function's amplitude. As a result, wavelet mutation is used in place of the traditional mutation algorithm. A mutation probability, $mp\epsilon \, [0{,}1]$, is determined for each particle in the swarm at each iteration. If mp is positive ($mp > 0$) and getting close to 1, the mutated particle elements will tend toward the maximum value of $z_{itr+1}^{h,m}$. If mp is negative ($mp < 0$) and getting close to -1, the mutated particle element will tend toward the lowest value of $z_{itr+1}^{h,m}$.When |mp| is big, the search space for fine-tuning is big, and vice versa, when |mp| is small, the search space for fine-tuning is small. The formula for mutation is:

$$z_{itr+1}^{h} = \begin{cases} z_{itr}^{h,m} + mp \times \left(ub - z_{itr}^{h,m}\right), mp > 0 \\ z_{itr}^{h,m} + mp \times \left(z_{itr}^{h,m} - lb\right), mp < 0 \end{cases} \tag{8}$$

where, $z_{itr+1}^{h,m}\ (h = 1,2,\ldots,N)$ denotes the h^{th} individual location in itr^{th} iteration, lb denotes the lower bound, ub denotes the upper bounds of the present search space. Similarly, mp represents mutating wavelet coefficient as given in Eq. (9).

$$mp = \left(1/\sqrt{(aa)}\right) \times mw \tag{9}$$

where, aa is the stretching parameter, which increases with the change of iterations. Its expression is given in Eq. (10).

$$aa = ss \times \ (1/ss)^{(1-(itr/max_{itr}))} \tag{10}$$

where, ss indicates a given constant value. itr denotes the present iteration and max_{itr} denotes the total number of iterations.

Morlet Wavelet function mw is expressed as in Eq. (11).

$$mw = exp\left(-\left(\left(numb^2\right)/2\right)\right) \times \ cos\left(5 \times \ numb\right) \tag{11}$$

where, $numb$denotes a random number between – 2.5 aa and 2.5 aa.

This strategy ensures that an individual with superior fitness will enter the next iteration, thereby enhancing the algorithm's convergence speed and optimization capability.

Lévy flight distribution

The Levy Flight is an example of a stochastic search algorithm introduced by Paul Pierre Levy in the year 1930[26,27] which uses a random walk to revise its results. Step walks are characterized as random walks with a certain probability distribution. The step sizes of Lévy flights are too ornery, by altering the step size, they can be used for both exploration and exploitation. proposed strategy generates the step sizes using Lévy distribution to exploit the search area. While exploring new solutions, controlling the Lévy flight random walks is necessary, to avoid large moves that causing the solutions to jump outside of the search space. For this reason, a step size factor that is determined by the size of the relevant problem should be used step size controller with a default value of 0.005 has been put in place to minimize the impact of Lévy flight on the beginning positions and enable searching around the produced positions. To generate numbers that are random with a Lévy Flight distribution, Eq. (12) has been examined as follows:

$$L = \gamma \ \times \ \frac{\mu}{|v|^{\frac{1}{\beta}}} \tag{12}$$

where, μ and v have a Gaussian distribution, γ is step reducer factor having fixed value of 0.005.

$$\mu \sim \left(0, \ \sigma_{\mu}^{2}\right), \ v \sim \left(0, \ \sigma_{v}^{2}\right) \tag{13}$$

$$\sigma_{\mu} = \left[\frac{\Gamma\ (1+\beta\ ) \times \ \sin(\pi\,\beta\ /2)}{\Gamma\ [(1+\beta\ )/2]\,\beta\ \times\ 2^{\beta\ -1/2}}\right]^{1/\beta} \tag{14}$$

where, $\beta\ = 1.5, \sigma_{v} = 1$, and a classic gamma function is denoted by the symbol Γ .

In Eq. (15), the Lévy flight procedure is presented which is used to update the chameleon positions:

$$z_{itr+1}^{h} = z_{itr+1}^{h} + L \times \ z_{itr+1}^{h} \tag{15}$$

where, L represents the Lévy flight distribution.

$B_t \leftarrow 0.1$ (probability to update position)
$rnd_1, rnd_2, rnd_3, rnd^{itr}$ defines random numbers in range of 0 and 1
ub and lb are search area's upper bound and lower bound
$d \leftarrow$problem dimension's
z^h_{itr} is chameleon h current position at itr iteration
zrc^h_{itr} is h^{th} chameleon rotating centred coordinates at itr iteration, as shown in equation (5)
Initialize n chameleons random position in search space using equation (2)
Initialize chameleons' dropping tongue velocity using equation (6)
$B_1$and B_2 two positive number that govern ability of exploration
while $(itr < \max\_itr)$ do
Define ω using equation (7)
    $for\ h = 1$ to n do
        $for\ m = 1$ to d do
            if $rnd^{itr} \geq P_{up}$then
                $z^{h,m}_{itr+1} = z^{h,m}_{itr} + B_1\left(B^{h,m}_{itr} - G^m_{itr}\right)rnd_1 + B_2\left(G^m_{itr} - z^{h,m}_{itr}\right)rnd_2 rnd_{itr} \geq B_t$
            else
                Compute mp using equation (11)
                $z^{h,m}_{itr+1} = \begin{cases} z^{h,m}_{itr} + mp \times \left(ub - z^{h,m}_{itr}\right), mp > 0 \\ z^{h,m}_{itr} + mp \times \left(z^{h,m}_{itr} - lb\right), mp < 0 \end{cases}$
            $endif$
        $endfor$
    $endfor$
Compute L for Chameleon velocity update using equation (12)
    $for\ h = 1$ to n do
        $z^h_{itr+1} = z^h_{itr+1} + L \times z^h_{itr+1}$
    $endfor$
Rearrange positions of chameleons' with the help of ub and lb
Update the positions of the chameleons
$itr = itr + 1$
$endwhile$

Algorithm 1. Pseudo-code of proposed mCSAMWL algorithm.

Computational complexity analysis of the proposed algorithm

The operational efficiency of the proposed mCSAMWL algorithm, with respect to time and space complexity, is discussed in this section.

Time complexity

The time complexity of the algorithm is influenced by the population size (n), the variable dimension (d), and the number of iterations (itr). For the original Chameleon Swarm Algorithm (CSA), the primary factors contributing to time complexity are the initialization and update processes of chameleon positions (including prey searching, tracking, and capturing). This can be expressed as:

$$\begin{aligned} O\left(CSA\right) &= O\left(O(initialization) + itrO(update)\right) \\ &= O(n \times d + itr \times n \times d) \end{aligned}$$

The proposed mCSAMWL extends the CSA by incorporating parameter adjustment, Morlet Wavelet Mutation and Levy Flight with step reducer strategy in each iteration. However, only the Levy Flight Distribution strategy affects the algorithm's time complexity. Therefore, the time complexity of proposed mCSAMWL algorithm is

$$\begin{aligned} O\left(mCSAMWL\right) &= O\left(O(initialization) + itr \times \left(O\left(update\right) + O\left(Levy\right)\right)\right) \\ &= O\left((n * d) + irt \times \left((n \times d) + (n \times d)\right)\right) \\ &= O(n \times d + itr \times n \times d) \end{aligned}$$

Thus, the proposed MCSA achieves different performance compared to CSA without increasing time complexity.

Empirical evaluation

MATLAB R2018b is used to examine the efficiency and capabilities of the proposed modified Chameleon Swarm Algorithm using Morlet Wavelet and Lévy Flight (mCSAMWL) algorithm. The "Intel(R), Core i7-4790 CPU@3.60GHz with 8 GB RAM" was used in all experiments used to determine the results. This research evaluates the effectiveness of the suggested algorithm against 68 standard benchmark functions. There are four distinct types of benchmark functions: unimodal with fixed dimensions, multimodal with fixed dimensions, unimodal with variable dimensions, and multimodal with variable-dimensions. An algorithm's exploitative and exploratory abilities are commonly evaluated using unimodal or multimodal functions. In this research, the proposed algorithm was evaluated using multiple benchmark functions. The functions utilized in this research are outlined in Annex A. The reference is drawn from[28] for these benchmark functions. In addition, the proposed CSAMWL's algorithm performance has been compared to ten commonly used algorithms, the Chameleon Swarm Algorithm (CSA)[25], the Elephant Herding Optimization (EHO)[29], the Gravitational Search Algorithm (GSA)[30], the Ant Colony Optimization (ACO)[31], the Earthworm Optimization Algorithm (EWA)[32], the Particle Swarm Optimization (PSO)[33], the Sine Cosine Algorithm (SCA)[34], the Krill Herd Algorithm (KHA)[35], the Artificial Bee Colony (ABC)[36], and the Monarch Butterfly Optimization (MBO)[37]. Table 2 displays the parameters of the contrasting algorithms as they were initially specified in the aforementioned published research articles. The 'NFEs' column in Table 2 indicates how many times a given function was evaluated. For each benchmark function, 30 separate runs for each algorithm are executed to generate the results.

Unimodal functions performance evaluation and statistical analysis

The exploitative potential of an algorithm can be measured with the help of unimodal functions. As a result, two tests were conducted using unimodal benchmark functions as part of this research. Table 3 depicts and compares all of the results from the first experiment for unimodal fixed-dimension functions. In the second experiment, the unimodal variable-dimension functions outcomes of 10 algorithms are compared. Table 4 summarizes the findings.

Tables 3 and 4 show that the mCSAMWL algorithm yielded the optimum results for the test functions F1-F3, F5, F6 as well as F8, F9, and F11 globally. It delivered excellent results for the benchmark functions F11, F18, and F21, and F15. It displays the effective exploitation capabilities of the proposed mCSAMWL method. To demonstrate the statistical distinction between the proposed mCSAMWL and other commonly used algorithms, the Friedman mean rank test is used. The statistical findings from the test are presented graphically in Fig. 1. The proposed algorithms and additional cutting-edge ones are represented on the X-axis. The Friedman mean ranks are displayed on the Y-axis.

The graph above illustrates that the best mean rank is the one with the smallest number. As can be seen in Fig. 1, the suggested mCSAMWL algorithm outperforms other popular metaheuristic algorithms when it comes to solving unimodal functions. The proposed algorithm mCSAMWL scores in first place, followed by GSA and CSA in second and third place, respectively. The results demonstrate that the mCSAMWL algorithm outperforms conventional metaheuristic algorithms in exploitative behaviour.

Multimodal function performance evaluation and statistical result analysis

Exploratory behaviour in algorithms is measured with the help of multimodal functions. Within the scope of this research, we conducted two experiments on multimodal benchmark functions. Table 5 shows the first experiment's results, which compare the performance of 10 commonly used metaheuristic algorithms against 27 multimodal fixed-dimension functions. The performance of 10 commonly used metaheuristic algorithms to the results of 17 multimodal variable-dimension functions is shown in Table 6.

After evaluating the statistics, we concluded that the proposed mCSAMWL algorithm can find global optimal solutions for the benchmark functions, F26-F35, F38-F39, F42, F43, F48-F50and comparable result for function F40 which are shown in Tables 5 and 6. The proposed mCSAMWL algorithm also outperforms the F46, F54, F61, F63 and gives competitive results for F50, F58, F67 function when compared to alternative algorithms. It demonstrates how the suggested mCSAMWL algorithm efficiently explores it's given search space. The Friedman's mean rank test demonstrates the statistical distinction between mCSAMWL and other commonly used algorithms. The graphical representation of Friedman's mean rank outcomes from the test can be seen in Fig. 2.

In Fig. 2, the performance of the proposed mCSAMWL algorithm is superior to existing, widely used metaheuristic algorithms for multimodal benchmark functions. In light of the findings, the mCSAMWL achieved a top ranking, followed by the CSA and the GSA in the stipulated order. It shows that the suggested mCSAMWL algorithm has statistically superior exploratory behaviour compared to the other commonly used metaheuristic algorithms. It can be seen in Fig. 2, the proposed mCSAMWL algorithm outshines when applied to multimodal functions compared to the other commonly used metaheuristic algorithms. The findings showed that mCSAMWL was the winner, with CSA and GSA coming in second and third place, respectively. Finally, it can be concluded that the proposed mCSAMWL algorithm performs better in exploratory behavior than other

Algorithms	Parameters	NFE's
	Population size = 50, Iteration = 1000	
CSA[25]	$p_1 = 0.25, p_2 = 1.50, p_3 = 1,\ c_1 = 1.75, c_2 = 1.75$	50,000
ACO[31]	N=50, Q=20, τ_0=1e−06, q_0=1, ρ_g=0.9, ρ_l=0.5, s=1, β=5	50,000
ABC[36]	$N = 50,\ Limit = 0.5 \times N \times D$	50,000
EHO[29]	$N = 25,\ \alpha = 0.5,\ \beta = 0.1$	50,000
EWA[32]	$N = 50,\ \alpha = 0.98,\ \beta = 1,\ \gamma = 0.9$	50,000
GSA[30]	N=50, α=20, G_0=100, k=[$N \rightarrow 1$]	50,000
KHA[35]	$N = 50,\ N^{max} = 0.01,\ V_f = 0.02,\ D^{max} = 0.005$	50,000
MBO[37]	$N = 50,\ S_{max} = 1.0,\ BAR = 5/12,\ p = 5/12$	50,000
PSO[33]	N=50, c_1=2, c_2=2, w=[$0.2 \rightarrow 0.9$]	50,000
SCA[34]	$N = 50,\ a = 2$	50,000

Table 2. Algorithm's parameter settings.

F. No.	F1			F2			F3		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**0**	**0**	**0**	**0**	**0**	**0**	**1.38E−87**	**1.38E−87**	6.81E-103
CSA	**0**	**0**	**0**	**0**	**0**	**0**	**1.38E-87**	**1.38E-87**	6.81E-103
ABC	5.98E−05	9.43E−05	0.000112	1.87E−05	3.36E−05	3.64E−05	**1.38E−87**	**1.38E-87**	6.80E-103
ACO	**0**	4.19E−27	2.29E−26	0	0	0	**1.38E−87**	**1.38E-87**	6.80E-103
EHO	0.0032	0.004055	0.003694	0.018358	0.023039	0.018387	0.024692	0.039861	0.041087
EWA	0.145742	0.265849	0.360356	1.146993	2.006107	2.651282	4.516292	5.77657	3.933894
GSA	1.07E−20	1.58E−20	1.49E−20	8.59E−21	1.32E−20	1.32E-−20	2.73E−06	3.34E-05	6.68E-05
KHA	1.52E−11	0.026154	0.143252	2.88E−11	3.67E−11	3.99E−11	**1.38E−87**	**1.38E-87**	6.80E-103
MBO	0.026759	0.118279	0.285487	0.0014	0.003748	0.006119	**1.38E−87**	**1.38E-87**	6.80E-103
PSO	**0**	0.025402	0.139134	**0**	**0**	**0**	**1.38E−87**	**1.38E-87**	6.81E-103
SCA	4.80E−05	9.60E−05	1.51E−04	2.39E−04	3.34E−04	2.85E−04	**1.38E−87**	**1.38E-87**	6.81E-103
F. No.	**F4**			**F5**			**F6**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	4.68E−36	1.78E−35	3.46E−35	**0.292579**	**0.292579**	8.48E−11	**19.10588**	**19.10588**	1.12E-06
CSA	4.65E−65	1.62E−64	2.86E−64	**0.292579**	**0.292579**	8.87E−17	**19.10588**	**19.10588**	1.33E-14
ABC	2.40E−05	3.67E−05	3.51E−05	0.292588	0.292595	1.92E−05	19.10589	19.10591	3.03E-05
ACO	8.40E−176	2.04E−78	1.12E−77	**0.292579**	**0.292579**	7.46E−07	**19.10588**	**19.10588**	1.54E-14
EHO	1.93E−09	2.25E−09	1.75E−09	0.292584	0.29259	1.42E−05	19.20103	19.2239	0.116325
EWA	4.95E−07	1.83E−06	3.40E−06	0.293813	0.29456	0.002437	31.3169	105.956	236.8043
GSA	3.63E−22	5.72E−22	5.30E−22	0.30328	0.30602	0.011648	**19.10588**	**19.10588**	8.16E-15
KHA	1.55E−12	2.15E−12	1.85E−12	**0.292579**	**0.292579**	3.83E−07	**19.10588**	19.10589	1.10E-05
MBO	8.63E−12	1.78E−11	2.92E−11	0.292596	0.292674	0.000159	19.10731	19.11448	0.020192
PSO	4.78E−221	1.31E−211	0	**0.292579**	**0.292579**	9.94E−17	**19.10588**	**19.10588**	9.17E-15
SCA	1.92E−132	8.64E−122	4.72E−121	**0.292579**	**0.292579**	2.70E−07	19.11437	19.11842	1.05E-02
F. No.	**F7**			**F8**			**F9**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	1.74E−08	2.14E−08	1.76E−08	**0**	**0**	**0**	**-0.00379**	**-0.00379**	3.97E-14
CSA	**0**	**0**	**0**	**0**	**0**	**0**	**-0.00379**	**-0.00379**	1.76E-18
ABC	0.000582	0.001055	0.001004	0.001171	0.00151	0.001425	**-0.00379**	**-0.00379**	2.90E-08
ACO	0.015208	0.015547	0.010022	0.011239	0.012175	0.008807	**-0.00379**	**-0.00379**	1.76E-18
EHO	0.001531	0.001736	0.001328	**0**	**0**	**0**	**-0.00379**	**-0.00379**	2.23E-06
EWA	0.183569	0.24004	0.242151	0.314836	0.377297	0.35672	0.125618	0.180528	0.177383
GSA	0.001838	0.002423	0.0026	**0**	**0**	**0**	**-0.00379**	**-0.00379**	1.42E-18
KHA	0.00433	0.007633	0.008756	1.85E−09	3.92E−09	5.93E−09	**-0.00379**	**-0.00379**	2.75E-10
MBO	3.34E−09	6.39E−09	8.14E−09	**0**	**0**	**0**	**-0.00379**	**-0.00379**	3.09E-10
PSO	7.87E−22	1.68E−19	5.10E−19	**0**	**0**	**0**	**-0.00379**	**-0.00379**	1.76E-18
SCA	9.88E−05	1.76E−04	1.75E−04	**0**	**0**	**0**	**-0.00379**	**-0.00379**	1.09E-10

Table 3. Performance analysis for unimodal fixed-dimension benchmark functions.

popular metaheuristic algorithms. Friedman's mean rank test is performed for the complete statistical evaluation of unimodal and multimodal benchmark functions. Figure 3 depicts the outcomes of a Friedman mean rank test. Friedman's mean rank test results reveal that the proposed mCSAMWL algorithm is the finest among other algorithms, followed by the CSA and the GSA algorithms. Finally, the proposed mCSAMWL algorithm has proven its perfection and significant potential for handling a wide range of optimization challenges across various situations.

Comparison of proposed mCSAMWL algorithm with other algorithms on CEC2017 benchmark functions

Table 7 presents a comparative analysis of ten optimization algorithms evaluated on the CEC 2017 benchmark functions (F69-F97). Performance is assessed using mean and standard deviation. The proposed algorithm outperformed the Chameleon Search Algorithm (CSA) specifically on benchmark functions F82, F84, F88, F95, and F97, while maintaining comparable performance levels across all other test functions. For other optimization algorithms, the proposed algorithm, mCSAMWL, achieved the lowest mean value for function F70 and also demonstrated superior performance by obtaining the lowest mean values among functions F75, F79-F82, and F84-F86. Furthermore, mCSAMWL exhibited relatively low standard deviations for functions F69-F75, indicating good stability. Notably, mCSAMWL significantly outperformed other algorithms on functions F79-F85, demonstrating exceptional stability in this range. While both mCSAMWL and ABC frequently

F. No.	F10			F11			F12		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	6.19E−06	1.32E−05	2.51E−05	**5.52E−21**	**2.40E−20**	4.34E−20	0.00533168	0.00646178	0.0037578
CSA	6.60E−02	0.11517009	0.149641953	1.79E−07	2.47E−07	2.01E−07	48.04349124	50.54523542	17.34103894
ABC	1241.975	1402.788	750.5982	0.000124	0.000247	0.000282	128.8663	134.1316	29.39484
ACO	13.0063	13.7159	5.10303	7.39E−04	1.24E−03	1.16E−03	2.63076	2.70499	4.43E-01
EHO	0.006492	0.00651	0.000444	8.73E−11	1.28E−10	1.16E−10	0.353929	0.354485	0.014399
EWA	2.517293	11.39267	23.56852	8.94E−07	2.18E−06	3.29E−06	7.466396	14.35868	14.94444
GSA	2.07E−17	2.12E−17	6.08E−18	2.13E−18	3.38E−17	9.47E−17	2.17E−08	2.21E−08	3.49E-09
KHA	0.020362	0.026725	0.02001	2.22E−08	6.97E−08	1.50E − 07	0.035143	0.052076	0.061898
MBO	4596.25	14744.9	22191.3	1.71E−13	5.32E−13	8.30E−13	423.6722	455.6837	379.9961
PSO	3.21E−04	1.44E−02	6.22E−02	1.13E−15	1.75E−09	7.52E-09	2.37E-05	7.49E-05	9.73E-05
SCA	2.63E-04	1.13E-01	0.612081	4.96E-14	1.35E-09	6.45E-09	6.59E-06	4.71E-05	7.98E-05

F. No.	F13			F14			F15		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	0.18304515	0.22068021	0.1882462	0.00010503	0.00010722	2.36E-05	**-155**	**-155**	**0**
CSA	11.8880527	12.1822205	2.30788966	0.04923997	0.04781611	0.02958289	-145	-144.1	12.8716475
ABC	61.09136	61.24847	3.794662	1102.435	1285.905	580.0211	-132	-131.9	5.510804
ACO	79.6819	80.3156	4.35088	12.7369	13.5783	5.8264	**-155**	**-155**	**0**
EHO	0.02531	0.025286	0.001971	2.904072	2.876425	0.282671	-42	-42.1667	4.000718
EWA	0.985505	1.344147	1.023447	10.87574	29.13764	36.82736	-29.5	-31.1667	9.089909
GSA	3.22E-09	0.00784	0.0316	2.60E-17	2.46E-17	6.72E-18	-118	-118.067	2.58554
KHA	0.672508	0.707143	0.251597	0.018918	0.025874	0.024961	-69.5	-69.7	8.847949
MBO	29.15408	31.31096	25.76838	5374.206	18974.32	25001.76	**-155**	-141.633	23.68978
PSO	11.62564	15.29934	10.26808	4.251723	4.17188	0.300366	-106.5	-107.033	5.635377
SCA	1.20E + 01	1.27E + 01	8.59E + 00	4.380958	4.339362	4.13E-01	-108	-108.567	3.88E + 00

F. No.	F16			F17			F18		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	0.01837619	0.03352073	0.04473552	6.55E-35	1.12E-30	3.20E-30	**0.1931209**	**2.63902853**	7.45892528
CSA	285.490459	288.25074	47.4462996	9.06E-06	0.00072376	0.00360032	82.996705	140.372557	155.406148
ABC	194.1793	14811.07	71275.07	553.2635	4650.138	14193.3	193181.6	261336.7	222116.3
ACO	1.79E + 20	2.23E + 22	6.63E + 22	2.45E + 06	3.99E + 06	3.98E + 06	4.14E + 04	4.71E + 04	2.50E + 04
EHO	0.36319	0.357741	0.01336	3.92E-26	4.52E-26	3.18E-26	28.87013	28.86558	0.024155
EWA	16.71175	25.38461	23.04249	5.95E-09	0.013327	0.047334	256.4014	1538.986	3501.003
GSA	54.294	57.2278	38.3186	2.54E-88	4.47E-88	6.25E-88	26.0875	28.2475	11.7991
KHA	593.1884	2.47E + 12	1.28E + 13	1.01E-11	5.78E-11	1.49E-10	33.72902	61.43647	54.47974
MBO	3.94E + 28	1.81E + 38	6.85E + 38	44412.16	4.74E + 08	1.86E + 09	1.59E + 07	3.60E + 07	7.10E + 07
PSO	8.84E-06	1.63E-05	2.13E-05	0.019961	11.86305	37.93097	35.98856	239.1065	554.296
SCA	4.39E-06	5.41E-05	1.11E-04	0.004608	144.4601	548.8833	30.98788	148.2003	4.83E + 02

F. No.	F19			F20			F21		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	3.23E-09	3.99E-09	3.20E-09	0.039781	0.12569	0.21624	**1.29E-05**	**2.18E-05**	2.66E-05
CSA	0.000290	0.000507	0.000947	3.921985	6.11043	6.445442	9.479590	12.36284	10.195504
ABC	1.835381	1.840576	0.913245	1803.107	2367.89	1856.635	109.4227	120.606	37.40893
ACO	5.26E-03	6.02E-03	2.72E-03	1.85E + 02	2.17E + 02	1.19E + 02	1.79E + 02	2.00E + 02	1.17E + 02
EHO	6.01E-05	5.99E-05	3.21E-06	0.909557	0.906891	0.01413	0.000292	0.000289	4.27E-05
EWA	275.002	280.5366	109.2263	2.663677	13.7822	34.05692	133.3671	128.9955	29.75407
GSA	3.71E-17	3.75E-17	1.28E-17	0.666667	0.667923	0.006882	0.011159	0.017749	0.016491
KHA	5.400354	9.023223	10.98069	0.827851	0.971692	0.43077	0.586635	0.67029	0.408763
MBO	14.16177	347.1522	616.0515	8.73E + 05	7.72E + 05	6.85E + 05	1991.609	3067.511	3226.672
PSO	1.02E-07	3.38E-06	1.38E-05	0.698241	1.295457	1.484248	2.46E-03	1.56E-01	4.76E-01
SCA	1.29E-07	4.98E-06	1.29E-05	0.710611	0.90133	0.499484	0.006219	0.025125	0.041677

F. No.	F22			F23			F24		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	-0.9999999	-0.7666655	0.43018239	-0.9999999	-0.76666545	0.43018239	6.20E-07	8.84E-07	9.16E-07
CSA	4.34E-232	4.34E-232	0	4.34E-232	4.34E-232	0	2.13E + 00	3.41E + 00	2.98E + 00
ABC	5.67E-51	2.74E-46	1.29E-45	5.67E-51	2.74E-46	1.29E-45	2.38E + 02	2.32E + 02	8.58E + 01

Continued

F. No.	F22			F23			F24		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
ACO	6.30E-216	3.00E-206	1.70E-209	6.30E-216	3.00E-206	1.70E-209	8.90E-01	1.00E+00	4.82E-01
EHO	1.92E-74	7.76E-70	3.53E-69	1.92E-74	7.76E-70	3.53E-69	1.03E-03	9.96E-04	1.06E-04
EWA	7.00E-42	1.69E-37	7.70E-37	7.00E-42	1.69E-37	7.70E-37	3.05E-01	2.87E+00	6.13E+00
GSA	7.63E-43	3.27E-37	1.04E-36	7.63E-43	3.27E-37	1.04E-36	1.53E-16	1.81E-16	1.01E-16
KHA	9.10E-228	2.71E-45	1.48E-44	9.10E-228	2.71E-45	1.48E-44	4752.622	4874.505	3272.272
MBO	4.30E-232	1.50E-210	1.80E-222	4.30E-232	1.50E-210	1.80E-222	1.39E-02	3.28E-02	4.96E-02
PSO	7.73E-199	1.77E-191	0	7.73E-199	1.77E-191	0	4.84E-06	3.39E-05	8.33E-05
SCA	6.49E-199	1.87E-193	0	6.49E-199	1.87E-193	0	1.81E-05	3.56E-04	1.06E-03

Table 4. Performance analysis for unimodal variable-dimension benchmark functions.

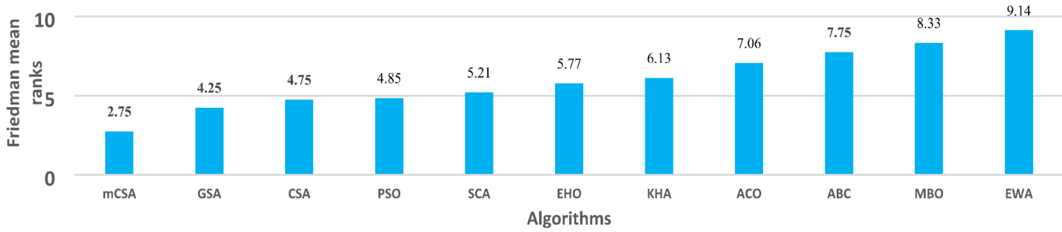

Fig. 1. Friedman Mean Rank test comparison for unimodal variable-dimension benchmark functions.

achieved the lowest mean values for functions F88-F92, mCSAMWL maintained competitive performance across the remaining functions F86-F97, consistently yielding results close to the optimal values, even when not achieving the absolute lowest mean.

In contrast, EWA and EHO generally exhibited higher variability and less optimal performance compared to the other evaluated algorithms. Overall, mCSAMWL demonstrated strong performance, with particular excellence observed in functions F79-F85. This suggests balanced optimization capabilities, effectively combining exploration and exploitation, as evidenced by its consistent performance across diverse function types. The low standard deviations associated with mCSAMWL further underscore its reliable and stable performance, positioning it as a robust choice for a variety of optimization problems.

Ablation study of the proposed mCSAMWL algorithm

To overcome the defects in the original algorithm, this paper proposes a modified version of Chameleon Search Algorithm. First, the exploration phase of CSA is modified using Morlet Wavelet Mutation to achieve better convergence performance. Then, we introduce the Lévy Flight distribution with step reducer feature in the exploitation part to help search agents escape from the local optima. To evaluate the effectiveness of each component, two mCSAMWL-derived variants are designed individually for comparison study in this subsection, which are listed below:

- CSAMW (modification of CSA with Morlet Wavelet Mutation only).
- CSALF (modification of CSA with Lévy Flight distribution only).
- Proposed mCSAMWL (modified CSA using Morlet Wavelet mutation and Lévy Flight Distribution).

Under the same experimental setting original CSA, CSAMW, CSALF, and mCSAMWL are tested on 23 different types of benchmark functions concurrently. The obtained median, mean fitness (mean) and standard deviation (Std) results are listed in Table 8.

A preliminary analysis on the simple functions F1-F3 reveals comparable performance across all methods, with the proposed method and CSA achieving near-optimal results. CSAMW and CSALF exhibit marginally weaker performance on F1. While CSA and CSALF demonstrate advantages on specific functions (F4 and F7 for CSA; F4 for CSALF), the proposed method maintains competitive and comparable or equivalent performance across the majority of other functions.

The proposed algorithm demonstrates a distinct advantage on functions F10-F15, significantly outperforming CSA and CSAMW across most of this range (F10-F14) and achieving superior results on F15. Furthermore, it exhibits improved performance compared to CSALF on F11-F14. Regarding functions F16-F24, the proposed algorithm continues to perform strongly, exhibiting substantial improvements over CSA and generally achieving superior results compared to CSAMW. The comparison with CSALF is more complex, with CSALF demonstrating better performance on F22 and F24; however, the proposed algorithm demonstrates greater overall consistency across this function set. Overall, the proposed algorithm performs well, particularly on more complex functions, demonstrating significant improvements over CSA and frequently outperforming CSAMW.

F. No.	F25			F26			F27		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	2.11E-34	2.11E-32	5.94E-32	**-195.62903**	**-195.62903**	3.41E-09	**-2.0218068**	**-2.0218068**	0.159154704
CSA	3.36E-63	2.24E-62	5.60E-62	**-195.629**	**-195.629**	5.78E-14	-2.02181	-2.02181	1.36E-15
ABC	1.29E-13	3.03E-13	3.53E-13	**-195.629**	**-195.629**	7.83E-10	**-2.02181**	**-2.02181**	9.40E-16
ACO	**0**	**0**	**0**	**-195.629**	**-195.629**	5.78E-14	**-2.02181**	**-2.02181**	1.36E-15
EHO	1.14E-06	1.19E-06	4.57E-07	-195.617	-195.616	0.007699	**-2.02181**	**-2.02181**	7.83E-06
EWA	7.33E-05	8.33E-04	2.29E-03	-195.598	-195.539	0.173797	-1.87048	-1.82072	0.14998
GSA	8.50E-20	1.08E-19	1.07E-19	**-195.629**	**-195.629**	5.78E-14	-2.02114	-2.02087	0.00099
KHA	2.15E-11	4.36E-11	7.74E-11	**-195.629**	**-195.629**	4.67E-09	**-2.02181**	**-2.02181**	6.04E-11
MBO	2.82E-12	8.20E-12	1.37E-11	**-195.629**	**-195.629**	0.000321	-2.02166	-2.02134	0.000756
PSO	1.90E-157	1.44E-148	7.86E-148	**-195.629**	**-195.629**	6.14E-05	**-2.02181**	**-2.02181**	1.59E-09
SCA	2.88E-158	6.95E-152	3.79E-151	**-195.629**	**-195.629**	8.44E-05	**-2.02181**	**-2.02181**	1.05E-09
F. No.	F28			F29			F30		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-106.765**	**-106.765**	2.71E-06	**-1.03163**	**-1.03163**	7.25E-10	**0.3978874**	**0.3978874**	2.81E-08
CSA	**-106.765**	**-106.765**	3.82E-14	**-1.03163**	**-1.03163**	6.65E-16	**0.397887**	**0.397887**	0
ABC	**-106.765**	**-106.765**	8.81E-05	**-1.03163**	**-1.03163**	1.29E-10	**0.397887**	**0.397887**	3.09E-11
ACO	**-106.765**	**-106.765**	4.05E-14	**-1.03163**	**-1.03163**	6.78E-16	**0.397887**	**0.397887**	**0**
EHO	-106.557	-106.524	0.208396	-1.02369	-1.01918	0.012386	0.403988	0.405804	0.007404
EWA	-99.6773	-95.3087	11.65851	-0.97467	-0.88761	0.204595	0.519468	0.631437	0.299147
GSA	**-106.765**	-106.687	0.333367	**-1.03163**	**-1.03163**	5.61E-16	**0.397887**	**0.397887**	**0**
KHA	**-106.765**	**-106.765**	4.31E-09	**-1.03163**	**-1.03163**	1.59E-10	**0.397887**	**0.397887**	9.52E-10
MBO	-106.762	-106.754	0.023937	-1.03159	-1.0314	0.000505	0.397905	0.397922	5.48E-05
PSO	-106.747	-106.738	0.02994	-1.03162	-1.03161	1.26E-05	0.398202	0.398307	0.000391
SCA	-106.745	-106.737	0.028856	-1.03162	-1.03162	1.38E-05	0.398185	0.398302	3.90E-04
F. No.	F31			F32			F33		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**3**	**3**	2.81E-09	**-3.862782**	**-3.862782**	7.72E-08	**-3.322**	-3.26650591	0.060329422
CSA	**3**	**3**	6.12E-16	**-3.86278**	**-3.86278**	2.68E-15	**-3.322**	-3.28214	0.057338
ABC	3.00005	3.00021	0.000453	**-3.86278**	**-3.86278**	3.78E-10	-3.32195	-3.32194	3.70E-05
ACO	**3**	**3**	1.25E-15	**-3.86278**	**-3.86278**	2.71E-15	**-3.32199**	-3.27443	0.05924
EHO	3.135808	3.192015	0.173313	-3.83639	-3.83114	0.019951	-2.83397	-2.81893	0.143102
EWA	20.88954	20.48409	13.54759	-3.75671	-3.74107	0.100396	-2.49959	-2.39237	0.519278
GSA	**3**	**3**	2.36E-15	**-3.86278**	**-3.86278**	2.44E-15	**-3.322**	**-3.322**	1.39E-15
KHA	**3**	**3**	6.79E-09	**-3.86278**	**-3.86278**	9.17E-10	**-3.322**	-3.27422	0.059526
MBO	3.004415	5.721823	14.78531	-3.86218	-3.86172	0.001306	-3.20289	-3.25594	0.060095
PSO	**3**	**3**	9.84E-06	-3.85482	-3.85683	0.003227	-3.01043	-3.0077	0.160297
SCA	**3**	**3**	6.99E-06	-3.85474	-3.85641	0.003118	-3.01438	-3.03234	9.49E-02
F. No.	F34			F35			F36		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-2.0626119**	**-2.062611**	3.58E-10	**1**	**1**	2.26E-15	90.66391828	91.73547588	4.461266533
CSA	**-2.06261**	**-2.06261**	9.03E-16	**1**	**1**	2.55E-15	**180.3276**	**180.3276**	**0**
ABC	**-2.06261**	**-2.06261**	5.09E-12	1.000146	1.000167	0.00014	180.2127	181.39	1.575748
ACO	**-2.06261**	**-2.06261**	9.03E-16	**1**	**1**	**0**	180.031	181.432	1.66114
EHO	-2.06257	-2.06255	6.94E-05	1.001608	1.001698	0.000707	**180.3276**	**180.3276**	**0**
EWA	**-2.06207**	**-2.05512**	0.021859	1.067353	1.177343	0.298134	180.7353	180.8577	0.48185
GSA	**-2.06261**	**-2.06261**	9.03E-16	**1**	**1**	1.05E-05	**180.3276**	**180.3277**	0.000182
KHA	**-2.06261**	**-2.06261**	2.64E-11	1.000001	1.000007	1.15E-05	180.9794	182.3246	1.11834
MBO	**-2.06261**	**-2.06261**	2.31E-13	**1**	**1**	3.69E-07	**180.3276**	**180.3276**	**0**
PSO	**-2.06261**	**-2.06261**	4.68E-06	**1**	**1**	**0**	**180.3276**	**180.3276**	**0**
SCA	**-2.06261**	**-2.06261**	6.70E-06	**1**	**1**	**0**	**180.3276**	**180.3276**	**0**
F. No.	F37			F38			F39		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	-4.31E+07	-9.8E+07	1.41E+084	**-42.944387**	**-42.944387**	6.66E-08	**4.85E-05**	**4.85E-05**	1.23E-14
CSA	**-24.1568**	**-24.1568**	1.03E-14	**-42.9444**	**-42.9444**	3.61E-14	**4.85E-05**	**4.85E-05**	1.41E-20
Continued									

F. No.	F37			F38			F39		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
ABC	**-24.1568**	**-24.1568**	2.66E-10	**-42.9444**	**-42.9444**	2.50E-12	**4.85E-05**	**4.85E-05**	6.14E-17
ACO	**-24.1568**	-24.1549	0.00968	**-42.9444**	**-42.9444**	3.61E-14	**4.85E-05**	**4.85E-05**	1.38E-20
EHO	-24.0661	-24.0457	0.081798	-42.8615	-42.8367	0.098118	**4.85E-05**	**4.85E-05**	1.52E-09
EWA	-11.3453	-13.1553	6.227318	-42.4701	-40.2579	3.145068	**4.85E-05**	**4.86E-05**	2.36E-07
GSA	**-24.1568**	-24.148	0.043026	-42.92	-42.8443	0.155228	**4.84E-05**	**4.84E-05**	1.37E-20
KHA	**-24.1568**	**-24.1568**	1.55E-09	-42.7208	-42.7208	0.227429	**4.85E-05**	**4.85E-05**	2.49E-16
MBO	**-24.1568**	**-24.1568**	3.00E-11	-42.9438	-42.9418	0.004484	**4.85E-05**	**4.85E-05**	5.45E-18
PSO	-24.1464	-24.142	0.013467	-42.9438	-42.9434	0.000925	**4.85E-05**	**4.85E-05**	1.56E-10
SCA	-24.1359	-24.1308	0.020502	-42.9436	-42.9432	0.001581	**4.85E-05**	**4.85E-05**	1.91E-10

F. No.	F40			F41			F42		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-0.0004414**	**-0.000428**	0.000110794	0.225177603	0.236948535	0.072175916	**-1**	**-1**	2.95E-08
CSA	-0.08478	-0.08478	2.82E-17	0.00118	0.00118	0	**-1**	**-1**	**0**
ABC	-0.00135	-0.00147	0.000477	0.085066	0.086875	0.018614	-0.2743	-0.38552	0.351841
ACO	-0.00023	-0.00308	0.015429	0.40729	0.34216	0.19321	**-1**	**-1**	**0**
EHO	-0.00033	-0.00033	8.30E-05	0.299955	0.319105	0.064923	-0.97574	-0.96931	0.023361
EWA	-0.00018	-0.00019	6.03E-05	0.518583	0.538881	0.165609	-0.0821	-0.28969	0.360464
GSA	-0.00721	-0.00734	0.000887	0.012586	0.012677	0.001616	**-1**	**-1**	**0**
KHA	-0.00032	-0.00033	6.76E-05	0.31831	0.330702	0.066606	**-1**	-0.93333	0.253707
MBO	-0.00147	-0.00153	0.000427	0.071294	0.073001	0.016403	**-1**	**-1**	2.79E-12
PSO	-0.05622	-0.47053	0.50395	0.379077	0.275878	0.219059	-0.99968	-0.99928	0.001
SCA	-0.00023	-0.26699	0.44958	0.415254	0.305494	0.218981	-0.99969	-0.99958	0.000406

F. No.	F43			F44			F45		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**0.06447**	**0.06447**	2.89E-10	6.33E-09	1.02E-08	1.11E-08	1.42E-07	2.33E-07	2.68E-07
CSA	**0.06447**	**0.06447**	5.65E-17	5.19E-62	2.09E-61	3.16E-61	0	1.58E-31	3.21E-31
ABC	**0.06447**	**0.06447**	6.35E-17	0.005392	0.0068	0.005398	6.53E-09	3.16E-08	7.23E-08
ACO	**0.06447**	**0.06447**	4.82E-17	1.22E-05	1.75E-03	4.97E-03	**0**	**0**	**0**
EHO	0.064678	0.064827	0.000361	2.360286	2.952881	1.908784	0.015626	0.027624	0.023508
EWA	0.067695	0.070012	0.007023	1.328246	2.232653	2.055741	0.424127	0.972663	1.938851
GSA	**0.06447**	**0.06447**	5.04E-17	0.001635	0.010816	0.031779	7.12E-20	1.17E-19	1.15E-19
KHA	**0.06447**	**0.06447**	1.13E-12	5.27E-09	0.000189	0.001033	1.73E-11	5.40E-11	7.20E-11
MBO	**0.06447**	**0.06447**	1.18E-12	0.074933	0.251114	0.424296	0.000758	0.007305	0.013659
PSO	0.064473	0.064476	4.85E-06	0.000602	0.001295	0.001885	0.0042	0.006728	0.007226
SCA	0.064475	0.064478	7.28E-06	0.000434	0.000853	0.001007	0.003451	0.004811	0.003892

F. No.	F46			F47			F48		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-19.2085**	**-19.2085**	31070.34351	-0.99999	-0.99999	5.82E-06	**-10.8723**	**-10.868999**	0.007508046
CSA	**-19.2085**	**-19.2085**	4.57E-15	**-0.96353**	**-0.96353**	0	**-10.8723**	**-10.869**	0.007508
ABC	**-19.2085**	**-19.2085**	3.72E-10	**-0.96353**	**-0.96353**	3.92E-13	**-10.8723**	**-10.8723**	7.06E-11
ACO	**-19.2085**	-19.2084	2.74E-04	**-0.96353**	**-0.96353**	**0**	**-10.8723**	**-10.8723**	2.21E-09
EHO	-19.1904	-19.176	0.035102	-0.96348	-0.96346	7.70E-05	-10.8692	-10.8695	0.00184
EWA	-17.0841	-15.965	3.548643	-0.95901	-0.95425	0.011804	-10.8147	-10.7669	0.113016
GSA	**-19.2085**	-19.1985	0.014989	-0.96339	-0.96329	0.000228	-10.8649	-10.8624	0.010089
KHA	**-19.2085**	**-19.2085**	5.87E-10	**-0.96353**	**-0.96353**	1.08E-11	**-10.8723**	-10.8644	0.00987
MBO	-19.2084	-19.2074	0.001899	**-0.96353**	**-0.96353**	2.16E-14	**-10.8723**	-10.8714	0.001805
PSO	-19.2035	-19.2004	0.008389	-0.96352	-0.96352	1.56E-05	**-10.8723**	**-10.8723**	2.94E-07
SCA	-19.2031	-19.2009	0.007262	-0.96352	-0.96352	2.07E-05	**10.8723**	**-10.8723**	3.92E-07

F. No.	F49			F50			F51		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-186.731**	**-186.731**	9.47E-05	**-10.5364**	-10.2783333	1.411185639	1.97E-35	1.41E-34	3.19E-34
CSA	**-186.731**	**-186.731**	3.81E-14	**-10.5364**	-9.02633	2.579842	2.29E-64	8.22E-64	1.82E-63
ABC	**-186.731**	**-186.731**	6.01E-05	-10.5305	-10.5247	0.010206	1.78E-12	1.06E-11	3.48E-11
ACO	-186.652	-186.561	0.189325	**-10.5364**	-9.56257	2.56129	3.4e-315	2.4e-311	1.4e-310
EHO	-186.201	-186.085	0.478853	-4.95503	-5.3281-	1.17823	4.77E-08	4.94E-08	2.27E-08
Continued									

F. No.	F49			F50			F51		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
EWA	-163.845	-152.86	32.1348	-2.26778	-2.37761	1.178693	1.55E-06	3.40E-05	9.46E-05
GSA	-185.427	-185.171	1.287655	**-10.5364**	**-10.5364**	1.86E-15	2.92E-21	4.85E-21	5.40E-21
KHA	**-186.731**	**-186.731**	7.49E-07	**-10.5364**	-7.78394	3.709944	1.78E-12	2.54E-12	2.93E-12
MBO	-186.729	-186.718	0.020366	-2.87114	-4.47229	2.898341	3.01E-13	1.81E-12	3.85E-12
PSO	-186.647	-186.592	0.137766	-5.01782	-5.17974	1.641081	1.63E-155	4.87E-145	2.66E-144
SCA	-186.638	-186.513	0.274842	-4.95368	-4.99818	0.417868	1.94E-154	2.35E-146	9.10E-146

Table 5. Performance analysis on multimodal fixed-dimension benchmark test functions.

The Proposed algorithm demonstrates superior overall performance, especially in complex functions, while maintaining good stability. CSAMW shows excellent performance in specific cases but lacks consistency. CSALF excels in simpler functions and maintains good precision but may struggle with more complex optimization problems. Therefore, it can be concluded that the modifications implemented have a demonstrable and positive impact on the algorithm's capabilities.

Real-world engineering design problems

In this section, the proposed mCSAMWL algorithm is tested on three real world engineering design problems: the design of welded beams, tension/compression springs, the pressure vessel problem, and their performances are evaluated. In the real world, meta-heuristic algorithms are frequently used to solve engineering design problems. These engineering design problems from the actual world may involve up to 15,000 function evaluations. The parameter values are identical to those in Table 6. Thirty independent runs were conducted to determine the best, average, standard deviation, and worst outcomes. The MATLAB platform was used to evaluate the proposed mCSAMWL algorithm results, while the other algorithm's findings were obtained from the main research publications.

Welded beam design

It serves as a crucial benchmark for evaluating various optimization techniques. This problem aims to bring down the costs of setting up, welder jobs, and material expenses associated with constructing the welded beam. Shear stress, bending stress, buckling load, end deflection, and side constraints are among the property constraints. The design variables include the length of the welded part (l), the thickness of the welding (h), width (b) and height (t). The problem's mathematical representation can be expressed in the following Eqs. (16–25).

Consider

$$\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b] \tag{16}$$

Minimize

$$f\left(\overrightarrow{x}\right) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 \left(14.0 + x_2\right), \tag{17}$$

Subject to

$$z_1\left(\overrightarrow{x}\right) = \tau\ \left(\overrightarrow{x}\right) - \tau_{\ max} \leq\ 0, \tag{18}$$

$$z_2\left(\overrightarrow{x}\right) = \sigma\ \left(\overrightarrow{x}\right) - \sigma_{\ max} \leq\ 0, \tag{19}$$

$$z_3\left(\overrightarrow{x}\right) = \delta\ \left(\overrightarrow{x}\right) - \delta_{\ max} \leq\ 0, \tag{20}$$

$$z_4\left(\overrightarrow{x}\right) = x_1 - x_2 \leq\ 0, \tag{21}$$

$$z_5\left(\overrightarrow{x}\right) = P - P_c\left(\overrightarrow{x}\right) \leq\ 0, \tag{22}$$

$$z_6\left(\overrightarrow{x}\right) = 0.125 - x_1 \leq\ 0, \tag{23}$$

$$z_7\left(\overrightarrow{x}\right) = 1.10471 x_1^2 + 0.04811 x_3 x_4 \left(14.0 + x_2\right) - 5.0 \leq\ 0 \tag{24}$$

where,

$$\tau\ \left(\overrightarrow{x}\right) = \sqrt{\tau^{\,\prime\,2} + 2\tau^{\,\prime}\tau^{\,\prime\,\prime}\frac{x_2}{2R} + \tau^{\,\prime\,\prime\,2}}, \tau^{\,\prime}\ = \frac{P}{\sqrt{2}x_1x_2}, \tau^{\,\prime\,\prime}\ = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right),$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \sigma\ \left(\overrightarrow{x}\right) = \frac{6PL}{x_4 x_3^2},$$

F. No.	F52			F53			F54		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	8.796635	37.33597	46.83735	16.6103	15.79659	10.26487	**0.9**	**0.908184**	0.031144
CSA	238.3937	236.7943	5.995471	33.82992	34.46036	11.69824	1.000004	1.000005	5.43E-06
ABC	141.4293	142.5645	12.27796	101.5629	102.9385	9.394156	2.130489	2.083757	0.199241
ACO	268.1623	269.0643	7.641825	261.5707	258.3739	11.41031	7.44305	7.42176	0.28933
EHO	282.9957	283.2481	10.05296	0.004174	0.004197	0.000297	0.900089	0.900088	7.01E-06
EWA	308.6437	307.3129	20.38896	9.749386	17.30806	20.31796	1.238955	1.533064	0.682945
KHA	192.3084	189.6491	30.76055	14.45001	14.44976	5.087514	1.002005	1.670184	1.639661
GSA	334.164	329.096	18.884	13.929	14.725	3.8437	1	1	4.12E-17
MBO	120.3378	138.4848	95.70285	153.1993	182.8509	147.503	1.11029	1.984482	1.425243
PSO	284.885	283.941	10.731	2.403	22.051	32.595	5.352	5.187	1.932939
SCA	284.328	283.5439	9.383524	4.347903	23.4144	38.67129	5.80416	5.250232	1.544437
F. No.	F55			F56			F57		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	2.975284	82.41973	220.0163	0.007112	0.359627	0.527572	0.000191	0.020201	0.059137
CSA	7261.82	15884.43	26610.79	3.889502	4.069479	2.154352	1.494609	7.157269	16.25589
ABC	2.02E+08	2.57E+08	2.13E+08	8.093523	7.72828	1.512424	1.43E+07	5.90E+07	9.06E+07
ACO	5.45E+07	6.22E+07	4.93E+07	19.10605	19.49763	2.515075	3.313857	4.967021	5.238239
EHO	3185.183	3115.875	348.5288	0.003822	0.003748	0.00021	8.76E-08	9.38E-08	6.00E-08
EWA	8966.783	729977.8	2,427,104	0.149382	0.461467	0.578919	0.000483	0.002448	0.004004
KHA	62.0103	84.86197	79.89481	0.002959	0.007069	0.01026	39.05739	671.1095	1634.143
GSA	1.30E+06	3.19E+06	5.46E+06	2.33E-09	2.41E-09	4.29E-10	9.63E-07	0.000117	0.00061
MBO	1.11E+10	4.49E+10	5.77E+10	5.903644	16.86573	21.05661	9.62E+09	2.31E+11	4.71E+11
PSO	5648.34	24108.2	74191.49	0.00263	0.36883	1.91746	1.61E-05	0.068383	0.235971
SCA	5467.49	155527.1	663171.5	0.012907	0.2607	1.027096	2.92E-06	0.00044	0.001596
F. No.	F58			F59			F60		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**0.00053**	0.426366	0.796014	167.3204	187.7189	101.2445	0.649873	0.789873	0.430997
CSA	19.96677	19.96677	2.50E-13	559.5002	551.3081	126.7834	2.199873	2.229873	0.453454
ABC	11.37484	11.44861	1.135508	34930.49	37092.29	17430.46	10.55882	10.48822	1.535124
ACO	20.1862	17.82319	6.144791	537.5839	544.264	115.8048	2.437575	2.430375	0.162851
EHO	0.020634	0.020484	0.000879	91.22564	89.54167	7.977985	0.003621	0.003589	0.00068
EWA	2.143827	2.272101	1.256036	240.376	642.5243	941.1934	0.703339	0.708207	0.403842
KHA	0.015684	0.593088	0.773717	26.23202	28.17637	11.21959	0.399873	0.37654	0.067891
GSA	3.60E-09	3.63E-09	5.96E-10	6938.763	7228.061	3661.526	1.10053	1.23176	0.37839
MBO	18.79768	14.48087	6.868299	320131.8	523060.6	580796.7	11.55212	12.37401	9.481711
PSO	20.03872	11.22343	9.946171	98.94225	108.0212	33.63604	0.19989	0.257741	0.127403
SCA	18.40507	12.60029	8.840296	98.17692	106.4712	22.18569	0.199893	0.234155	0.090015
F. No.	F61			F62			F63		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**-1174.94**	**-1174.94**	0.005321	9.81E-07	0.008174	0.011995	**-0.99962**	**-0.99963**	0.000156
CSA	-1033.61	-1029.02	38.97252	0.063872	0.072718	0.036005	3.70E-16	2.19E-15	6.89E-15
ABC	-1017.71	-1021.8	22.05166	1.485707	1.470638	0.194775	1.52E-12	1.58E-12	5.04E-13
ACO	-671.582	-668.226	49.94495	0.825488	0.802076	0.081675	1.65E-10	1.71E-10	6.44E-11
EHO	-684.502	-688.633	23.95882	0.000323	0.000314	4.65E-05	-0.39712	-0.42138	0.108902
EWA	-697.45	-689.519	40.91739	0.241972	0.373633	0.371149	5.51E-11	-0.04825	0.126594
KHA	-1017.27	-1006.15	31.37359	0.010242	0.012663	0.011143	1.20E-14	4.00E-13	1.88E-12
GSA	-1111.37	-1109.95	26.92511	**0**	**0**	**0**	5.52E-30	5.98E-30	2.03E-30
MBO	-1076.01	-944.27	260.2547	4.011049	7.176058	6.325458	9.28E-12	1.46E-08	3.57E-08
PSO	-619.056	-613.96	43.5838	0.000387	0.103111	0.178225	1.52E-10	1.80E-10	1.31E-10
SCA	-623.941	-626.155	52.84877	0.003898	0.092368	0.157481	1.16E-10	1.56E-10	1.11E-10
F. No.	F64			F65			F66		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**3.52E-12**	**3.52E-12**	9.53E-16	8.70E-05	0.021863	0.055795	3.96E-06	0.007619	0.028974
CSA	1.76E-11	1.76E-11	3.69E-18	45.57699	41.77412	25.13016	6.2805	5.935862	2.325529
ABC	2.71E-11	2.64E-11	4.61E-12	53026.77	102341.9	125225.6	9.173057	5612.849	29411.94

Continued

F. No.	F64			F65			F66		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
ACO	3.12E-10	3.15E-10	1.67E-10	1.34E+05	1.91E+05	2.14E+05	2.29E+04	5.32E+04	7.41E+04
EHO	3.87E-07	4.23E-07	2.03E-07	2.473693	2.46669	0.132225	0.460823	0.45868	0.044851
EWA	9.86E-08	4.95E-07	1.09E-06	4.878637	5.158461	1.260947	1.82917	2.078248	0.963983
KHA	1.25E-10	2.32E-07	7.78E-07	0.000167	0.002012	0.004889	0.27899	0.467207	0.567746
GSA	3.65E-12	3.64E-12	5.52E-14	2.16E-18	2.26E-18	6.19E-19	1.47E-19	0.035409	0.103724
MBO	1.71E-11	1.50E-11	4.94E-12	1,410,627	1.72E+08	3.03E+08	65.10721	66,255,921	1.56E+08
PSO	2.93E-10	3.30E-10	2.02E-10	2.438557	5.188774	11.15144	0.681527	1.258365	1.435293
SCA	4.15E-10	4.08E-10	1.79E-10	2.687764	3.719848	3.361161	0.669408	0.865804	0.520371
F. No.	**F67**			**F68**					
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**			
Proposed	**-24.2097**	**-23.9244**	1.192935	0.555545	0.55615	0.258694			
CSA	-18.4328	-18.5526	1.970272	0.576142	0.646125	0.268376			
ABC	-19.5097	-19.6271	1.195944	2.051203	2.155658	0.618281			
ACO	-9.60851	-9.58506	0.368916	0.212721	0.219866	0.053785			
EHO	-11.6481	-11.5874	0.593605	**2.91E-05**	**3.09E-05**	**2.31E-05**			
EWA	-11.0755	-11.3069	0.87826	0.037217	0.044023	0.031715			
KHA	-22.3215	-22.3038	1.537473	0.037357	0.040049	0.016935			
GSA	-27.4414	-27.4173	0.739022	0.018441	0.020708	0.009074			
MBO	-16.5023	-15.9714	3.072747	82.25565	64.20257	42.81425			
PSO	-7.79848	-7.8029	0.748354	0.0117	0.015809	0.016752			
SCA	-7.5375	-7.60054	0.845505	0.01594	0.026684	0.03207			

Table 6. Performance analysis on multimodal variable-dimension benchmark test functions.

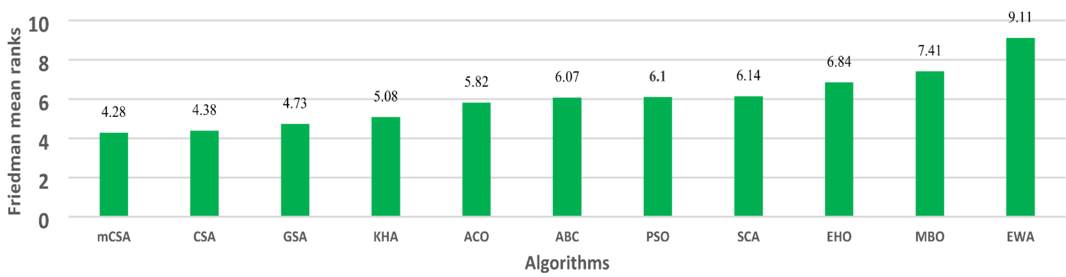

Fig. 2. Friedman mean rank test comparison for multi modal variable-dimension benchmark functions.

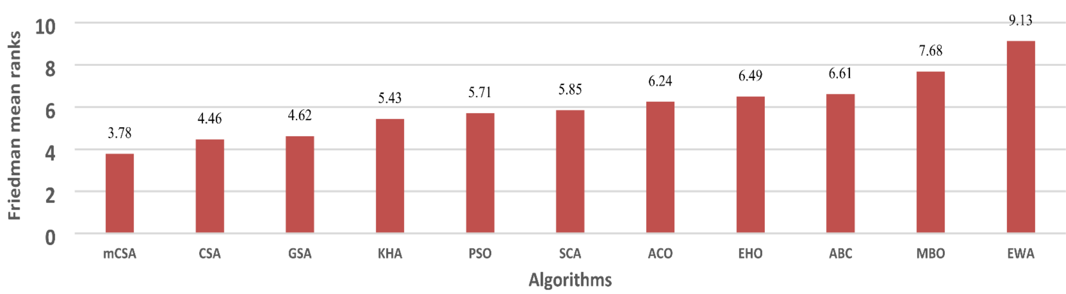

Fig. 3. Overall Friedman mean rank test comparison for 68-benchmark functions.

F. No.	Metrics	Proposed	CSA	ABC	ACO	EHO	EWA	GSA	KHA	MBO	PSO	SCA
F69	Mean	2.71E + 04	2.14E + 03	1.51e + 07	3473.399	3.91e + 09	9.11e + 09	275.9256	1231.884	2.69e + 09	6.84e + 08	7.63e + 08
	Std. dev	1.15E + 04	2.39E + 03	1.10e + 07	3912.219	9.35e + 08	3.23e + 09	233.972	1535.154	1.44e + 09	2.18e + 08	2.99e + 08
F70	Mean	3.00E + 02	3.00E + 02	11870.86	429.3429	8551.457	13011.73	10930.07	2447.185	26292.06	1907.703	1667.599
	Std. dev.	8.66E-01	1.62E-04	2601.034	100.7011	2534.666	3215.011	2476.143	2029.243	15858.02	1613.804	1068.301
F71	Mean	4.05E + 02	4.04E + 02	410.8322	403.9510	652.5524	1166.998	406.132	406.0538	570.7414	449.6804	442.3137
	Std. dev.	1.21E + 01	2.70E + 00	2.67069	0.168421	72.50278	321.4921	1.31166	2.621986	161.6766	28.83612	22.11951
F72	Mean	5.24E + 02	5.09E + 02	522.2496	529.6379	568.2867	593.2093	557.1713	528.6551	551.0986	548.0846	546.8131
	Std. dev.	9.82E + 00	3.51E + 00	3.519481	4.499739	8.934784	17.42522	9.202492	9.015167	15.73492	7.398523	5.74464
F73	Mean	6.06E + 02	6.02E + 02	602.8508	600	635.7782	652.2342	624.9813	609.1646	628.8333	619.3922	618.1006
	Std. dev.	5.56E + 00	1.26E + 00	0.860507	4.22e-14	4.67684	12.12105	10.25628	5.834042	12.43779	3.409804	3.228987
F74	Mean	7.37E + 02	7.21E + 02	741.9019	742.8243	827.7705	828.1968	715.2126	721.3522	784.1267	773.5919	775.7746
	Std. dev.	1.48E + 01	5.61E + 00	6.306468	4.737585	12.13333	25.76585	2.990161	6.346181	31.40896	10.22259	11.7603
F75	Mean	8.15E + 02	8.09E + 02	821.5658	830.0375	855.5772	864.5029	822.1544	818.7726	858.7612	839.6358	838.8361
	Std. dev.	6.65E + 00	4.60E + 00	5.143925	4.060205	5.797003	10.65639	4.887784	8.482399	20.30692	7.819929	7.597678
F76	Mean	9.46E + 02	9.03E + 02	928.2383	900	1357.765	1903.74	900	908.6887	1284.198	1006.305	996.8667
	Std. dev.	5.30E + 01	4.33E + 00	15.33102	0	128.6872	322.8076	0	28.82673	358.8684	64.61087	33.98794
F77	Mean	1.67E + 03	1.61E + 03	1558.567	2129.827	2497.313	3021.653	2765.49	2073.715	2217.919	2256.504	2317.386
	Std. dev.	2.73E + 02	2.26E + 02	135.5875	297.6214	191.6777	292.5522	240.1817	309.9358	280.2803	280.8657	260.37
F78	Mean	1.13E + 03	1.12E + 03	1134.388	1107.343	1349.82	6825.715	1138.12	1141.113	2563.498	1184.773	1192.63
	Std. dev.	1.47E + 01	2.88E + 01	10.65167	1.297857	80.78795	6327.225	14.08996	21.46149	2340.484	33.72219	37.18676
F79	Mean	3.12E + 04	1.12E + 04	9.50e + 05	22347.765	1.03e + 08	3.88e + 08	745403.5	1,820,698	77,676,773	14,944,227	16,555,949
	Std. dev.	5.10E + 04	1.09E + 04	6.41e + 05	13611.621	35,872,548	3.97e + 08	478,064	1,538,504	82,611,343	10,984,716	17,144,986
F80	Mean	1.99E + 03	1.75E + 03	15297.02	9620.5418	626073.7	4.87e + 07	12266.49	11241.84	2.66e + 06	31942.19	33333.14
	Std. dev.	3.45E + 02	2.70E + 02	6548.883	8275.0999	544045.6	6.26e + 07	3145.719	5797.629	3.35e + 06	19903.58	18764.29
F81	Mean	1.45E + 03	1.44E + 03	1947.539	2730.3463	1984.571	110349.4	6050.159	2808.717	141579.3	1754.587	1826.561
	Std. dev.	1.72E + 01	1.23E + 01	378.4723	3022.3391	550.6392	190099.6	1902.638	2590.406	293227.6	706.1036	663.9633
F82	Mean	1.58E + 03	1.59E + 03	1999.952	4073.5241	6461.742	1.83e + 06	19462.96	9883.937	273324.6	2396.541	2094.954
	Std. dev.	5.75E + 01	6.01E + 01	321.4302	4844.1921	2898.962	6.80e + 06	5487.597	9024.832	1,171,099	664.8275	358.072
F83	Mean	1.71E + 03	1.62E + 03	1665.888	1612.1326	1960.062	2197.858	2169.556	1947.692	1945.786	1736.888	1739.073
	Std. dev.	9.47E + 01	3.56E + 01	46.79145	0.7997813	88.95026	146.9085	103.7306	116.05	132.5913	84.03994	69.70736
F84	Mean	1.74E + 03	1.74E + 03	1738.019	1747.7528	1801.469	1903.499	1893.805	1763.032	1791.424	1775.685	1779.924
	Std. dev.	1.48E + 01	1.94E + 01	9.408983	2.6588455	20.7347	87.11427	109.6289	22.26015	65.79866	13.38982	18.55763
F85	Mean	2.80E + 03	2.01E + 03	15416.37	25677.209	1,531,451	1.74e + 08	9800.403	13680.13	3.03e + 06	127255.2	110502.3
	Std. dev.	1.71E + 03	1.76E + 02	9039.951	14976.346	1,270,096	2.52e + 08	4763.055	9106.063	6.53e + 06	113605.9	83703.33
F86	Mean	1.94E + 03	1.93E + 03	2514.381	7690.0857	9278.28	8.68e + 06	46813.25	8120.749	55548.52	6002.25	4925.699
	Std. dev.	2.94E + 01	2.86E + 01	959.7638	6775.2704	6721.444	1.39e + 07	19162.87	6219.891	231086.8	5463.183	4423.837
F87	Mean	2.05E + 03	2.03E + 03	2037.829	2004.4208	2141.217	2282.004	2275.684	2134.03	2112.056	2093.246	2096.934
	Std. dev.	3.20E + 01	9.34E + 00	10.01486	21.904078	29.40974	104.6475	100.1734	67.27155	49.97817	19.88185	25.01844
F88	Mean	2.29E + 03	2.30E + 03	2217.431	2331.0533	2261.978	2368.285	2358.121	2256.811	2346.541	2251.825	2243.573
	Std. dev.	5.72E + 01	3.28E + 01	12.55623	3.3602774	20.39812	47.79793	27.33747	61.28068	43.06961	62.53914	56.49253
F89	Mean	2.30E + 03	2.30E + 03	2285.16	2304.6889	2589.488	2945.56	2300.011	2302.17	2735.675	2355.994	2371.848
	Std. dev.	1.47E + 01	2.76E + 01	17.93655	0.9469408	91.07898	224.153	0.06297	1.620206	307.9171	25.29143	35.03303
F90	Mean	2.63E + 03	2.61E + 03	2589.826	2626.7905	2705.663	2791.775	2756.771	2631.141	2648.309	2655.9	2654.836
	Std. dev.	1.80E + 01	4.05E + 00	82.60015	4.3128631	15.3191	53.75314	53.20707	63.52294	17.83721	8.909316	11.34653
F91	Mean	2.73E + 03	2.73E + 03	2565.885	2760.0581	2706.624	2897.337	2653.707	2655.101	2793.87	2775.073	2755.772
	Std. dev.	8.18E + 01	4.43E + 01	19.14422	3.8818984	55.72094	92.80119	168.3652	129.5381	43.37924	43.80023	73.90712
F92	Mean	2.92E + 03	2.91E + 03	2918.783	2942.0745	3115.528	3460.244	2942.62	2927.836	3019.739	2962.343	2957.563
	Std. dev.	2.36E + 01	2.25E + 01	14.63454	15.009559	48.11038	207.4497	5.160336	22.26353	74.62458	12.74057	14.70874
F93	Mean	3.01E + 03	2.92E + 03	2821.021	3551.1879	3440.261	4140.495	3698.63	3073.743	3583.037	3081.613	3069.668
	Std. dev.	1.39E + 02	5.36E + 01	83.7193	517.84485	108.0022	403.2259	710.2253	278.4648	462.6585	52.76962	28.39842
F94	Mean	3.10E + 03	3.09E + 03	3102.174	3090.2951	3174.783	3308.397	3260.365	3139.024	3122.047	3103.287	3102.828
	Std. dev.	1.38E + 01	2.85E + 00	2.774271	1.3980473	19.00431	69.83134	38.84866	44.11037	26.0317	1.492919	1.780704
F95	Mean	3.30E + 03	3.37E + 03	3191.306	3411.8218	3466.476	3806.799	3456.981	3272.782	3435.844	3275.676	3253.551
	Std. dev.	2.17E-01	1.43E + 02	46.93048	1.85e-12	78.28678	142.6995	35.92788	141.697	103.3833	68.47646	35.62172
F96	Mean	3.19E + 03	3.15E + 03	3195.017	3174.7205	3330.083	3522.554	3427.118	3249.654	3327.52	3238.357	3229.173
	Std. dev.	3.84E + 01	1.51E + 01	28.17671	13.523257	48.34743	159.3801	138.0466	58.46674	104.0239	40.04224	39.43467

Continued

F. No.	Metrics	Proposed	CSA	ABC	ACO	EHO	EWA	GSA	KHA	MBO	PSO	SCA
F97	Mean	1.49E+05	1.57E+05	46151.02	393629.20	6,033,642	3.90e+07	963721.7	1.23e+06	4.56e+06	832719.8	768822.3
	Std. dev.	3.63E+05	2.75E+05	50420.62	533225.08	3,387,805	3.27e+07	273350.8	1.47e+06	4.09e+06	636666.3	655289.3

Table 7. Comparison of proposed algorithm and other algorithms on CEC-2017 benchmark functions.

F. No.	F1			F2			F3		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	**0**	**0**	**0**	**0**	**0**	**0**	**1.38E-87**	**1.38E-87**	6.81E-103
CSA	**0**	**0**	**0**	**0**	**0**	**0**	**1.38E-87**	**1.38E-87**	6.81E-103
CSAMW	0	5.75E-33	1.42E-32	0	0	0	**1.38E-87**	**1.38E-87**	6.80E-103
CSALF	1.28E-09	2.30E-09	2.52E-09	5.51E-09	7.65E-09	5.96E-09	**1.38E-87**	**1.38E-87**	6.80E-103
F. No.	**F4**			**F5**			**F6**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	4.68E-36	1.78E-35	3.46E-35	**0.292579**	**0.292579**	8.48E-11	**19.10588**	**19.10588**	1.12E-06
CSA	4.65E-65	1.62E-64	2.86E-64	**0.292579**	**0.292579**	8.87E-17	**19.10588**	**19.10588**	1.33E-14
CSAMW	9.50E-36	3.52E-35	7.42E-35	0.292588	2.93E-01	6.84E-17	19.10589	19.10591	5.32E-15
CSALF	1.52E-65	9.28E-65	2.00E-64	**0.292579**	**0.292579**	1.12E-11	**19.10588**	**19.10588**	2.85E-07
F. No.	**F7**			**F8**			**F9**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	1.74E-08	2.14E-08	1.76E-08	**0**	**0**	**0**	**-0.00379**	**-0.00379**	3.97E-14
CSA	**0**	**0**	**0**	**0**	**0**	**0**	**-0.00379**	**-0.00379**	1.76E-18
CSAMW	0	0	0	0	0	0	**-0.00379**	**-0.00379**	1.75E-18
CSALF	5.65E-09	1.27E-08	1.34E-08	0	0	0	**-0.00379**	**-0.00379**	6.53E-15
F. No.	**F10**			**F11**			**F12**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	6.19E-06	1.32E-05	2.51E-05	**5.52E-21**	**2.40E-20**	4.34E-20	5.33E-03	6.46E-03	3.76E-03
CSA	6.60E-02	1.15E-01	1.50E-01	1.79E-07	2.47E-07	2.01E-07	4.80E+01	5.05E+01	1.73E+01
CSAMW	6.93E-06	3.08E-03	9.60E-03	8.96E-14	1.36E-10	3.76E-10	5.72E-02	2.70E-01	4.21E-01
CSALF	8.44E-14	2.36E-13	5.15E-13	6.93E-44	2.56E-42	5.77E-42	1.78E-06	1.76E-06	9.61E-07
F. No.	**F13**			**F14**			**F15**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	1.83E-01	2.21E-01	1.88E-01	1.05E-04	1.07E-04	2.36E-05	**-155**	**-155**	**0**
CSA	1.19E+01	1.22E+01	2.31E+00	4.92E-02	4.78E-02	2.96E-02	-145	-144.1	1.29E+01
CSAMW	3.78E-02	8.60E-02	9.95E-02	1.26E-05	1.23E-03	4.06E-03	-155	-155	0
CSALF	1.68E+00	1.72E+00	9.37E-01	6.55E-05	6.70E-05	8.92E-06	**-155**	**-154.8**	**9.13E-01**
F. No.	**F16**			**F17**			**F18**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	1.84E-02	3.35E-02	4.47E-02	6.55E-35	1.12E-30	3.20E-30	**1.93E-01**	**2.64E+00**	7.46E+00
CSA	2.85E+02	2.88E+02	4.74E+01	9.06E-06	7.24E-04	3.60E-03	8.30E+01	1.40E+02	1.55E+02
CSAMW	1.28E-01	1.68E+01	5.23E+01	4.94E-43	1.58E-23	8.61E-23	2.16E-05	1.54E-03	5.45E-03
CSALF	1.71E-05	4.41E-05	7.51E-05	3.10E-45	3.54E-39	1.58E-38	2.51E+01	3.08E+01	1.75E+01
F. No.	**F19**			**F20**			**F21**		
Metrics	Median	Mean	Std. dev.	Median	Mean	Std. dev.	Median	Mean	Std. dev.
Proposed	3.23E-09	3.99E-09	3.20E-09	3.98E-02	1.26E-01	2.16E-01	**1.29E-05**	**2.18E-05**	2.66E-05
CSA	2.90E-04	5.03E-04	9.48E-04	3.92E+00	6.11E+00	6.45E+00	9.48E+00	1.24E+01	1.02E+01
CSAMW	1.68E-08	7.54E-07	2.03E-06	2.49E-01	2.74E-01	1.37E-01	2.07E-07	1.05E-04	2.95E-04
CSALF	2.04E-15	2.69E-15	2.87E-15	6.67E-01	6.67E-01	7.60E-04	2.79E-04	2.80E-04	1.21E-04
F. No.	**F22**			**F23**			**F24**		
Metrics	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**	**Median**	**Mean**	**Std. dev.**
Proposed	-1.00E+00	-7.67E-01	4.30E-01	1.73E+00	4.40E+00	6.86E+00	6.20E-07	8.84E-07	9.16E-07
CSA	4.34E-232	4.34E-232	0.00E+00	8.55E-02	2.29E+00	6.59E+00	2.13E+00	3.41E+00	2.98E+00
CSAMW	-1.00E+00	-7.00E-01	4.66E-01	4.75E-02	2.72E-01	7.03E-01	5.95E-06	3.26E-04	1.12E-03
CSALF	4.34E-232	4.35E-232	0	9.58E-01	6.86E+00	2.20E+01	7.29E-13	1.61E-12	2.70E-12

Table 8. Analysis of derived variants of proposed mCSAMWL algorithm for 24 benchmark functions.

Algorithm		Proposed	ABC	ACO	CSA	EWA	GSA	MBO	KHA	PSO	SCA
Optimal values for variables	h	0.1994	0.2093828	0.205735	0.20573	0.3756133	0.1471	0.1981115	0.205899	0.19741	0.2047
	l	3.2639	3.5463679	3.2530202	3.47049	2.2838984	5.49074	3.2881733	3.3844076	3.31506	3.53629
	t	7.0541	9.0481796	9.0366232	9.03662	6.5632029	10	9.4315589	8.8912834	10	9.00429
	b	0.20382	0.2211365	0.205729	0.20573	0.4560403	0.21773	0.2039614	0.2126729	0.2014	0.21003
Optimum cost		1.69702	1.860814	1.695245	1.72485	2.7008027	2.17286	1.7254857	1.7425535	1.8204	1.75917

Table 9. Comparative analysis for the welded beam design problem.

	Proposed	ABC	ACO	CSA	EHO	EWA	GSA	MBO	KHA	PSO	SCA
Best	1.69702	1.860814	1.695245	1.72485	1.7434	2.7008027	2.17286	1.7425535	1.7400148	1.8204	1.75917
Mean	1.7409	2.089541	1.781379	1.72485	2.0042	4.0255967	2.54424	2.455537	2.33025	2.23031	1.81766
Worst	1.7307	2.494788	2.009871	1.72485	2.48355	5.6322353	3.00366	3.284443	3.023209	3.04823	1.87341
Std.	0.40149	0.15017	0.097678	2.2352E−05	0.16505	0.8006264	0.25586	0.403898	0.367012	0.32453	0.02754

Table 10. Comparative statistical analysis for the welded beam design problem.

$$\delta\left(\overrightarrow{x}\right)=\frac{4PL^3}{Ex_3^3x_4}, P_c\left(\overrightarrow{x}\right)=\frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1-\frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), P=6000lb,$$

$$L=14\in, E=30\times\ 10^6psi, G=12\times\ 10^6psi$$

Range of design variables

$$0.1\leq\ x_1\leq\ 2.00{,}0.1\leq\ x_2\leq\ 10.0{,}0.1\leq\ x_3\leq\ 10.0, 0.1\leq\ x_4\leq\ 2.00 \tag{25}$$

The mCSAMWL algorithm optimizes the welding beam problem's parameters, and the outcome is 1.69702. It is evident from Tables 9 and 10 that the mCSAMWL algorithm has generated the best solution, which is superior to other algorithms except the ACO algorithm. In conclusion, the proposed algorithm is reliable for getting good results for the welded beam design problem.

Tension/compression spring design

Tension/compression spring design is another popular mechanical problem. The weight of the spring needs to be reduced as much as possible to accomplish the goal of this problem's. It can be achieved by managing the coil mean diameter D, the wire diameter d, and the active coil count N. When designing a compression spring, the predetermined restrictions on shear stress, minimum deflection and surge frequency must be adhered to, all while maintaining as little weight as is feasible. The following is the way to define an individual of the goal variables:

Consider

$$\overrightarrow{x}=[x_1,x_2,x_3]=[d,D,N] \tag{26}$$

Minimize

$$f\left(\overrightarrow{x}\right)=(2+x_3)\,x_2x_1^2, \tag{27}$$

Subject to

$$z_1\left(\overrightarrow{x}\right)=1-\frac{x_2^3x_3}{71785x_1^4}\leq\ 0, \tag{28}$$

$$z_2\left(\overrightarrow{x}\right)=\frac{4x_2^2-x_1x_2}{12566\left(x_2x_1^3-x_1^4\right)}+\frac{1}{5108x_1^2}-1\leq\ 0, \tag{29}$$

$$z_3\left(\overrightarrow{x}\right)=1-\frac{140.45x_1}{x_2^2x_3}\leq\ 0, \tag{30}$$

$$z_4\left(\overrightarrow{x}\right)=\frac{x_1+x_2}{1.5}-1\leq\ 0, \tag{31}$$

Design variables range

Algorithms	Proposed	ABC	ACO	CSA	EWA	GSA	KHA	MBO	PSO	SCA
$d\left(x_1\right)$	0.051899	0.05	0.0506762	0.05178	0.0576993	0.05	0.0500053	0.3104684	0.051728	0.05078
$D\left(x_2\right)$	0.321778	0.3155046	0.3328353	0.35885	0.5148301	0.31731	0.3174998	14.993803	0.357644	0.33478
$N\left(x_3\right)$	11.0644	14.454456	12.840343	11.165	10.408336	14.2287	14.024663	0.0131901	11.244543	12.7227
$f\left(\overrightarrow{x}\right)$	0.012668	0.0129786	0.0126847	0.0127	0.0212676	0.01287	0.0126744	0.3104684	0.012674	0.01271

Table 11. Comparative analysis for tension/compression spring design problem.

Algorithms	Proposed	ABC	ACO	CSA	EWA	GSA	KHA	MBO	PSO	SCA
Best	0.0126681	0.0129786	0.0126847	0.01267	0.0212676	0.01287	0.0126744	0.0131901	0.012674	0.01271
Mean	0.0126717	0.0141119	0.0135736	0.01267	9.66E+04	0.01344	0.0129385	0.0185521	0.01273	0.01284
Worst	0.0126708	0.0193867	0.0160979	0.01267	8.19E+05	0.01421	0.0141048	0.0260374	0.012924	0.013
Std. dev	1.2193E-05	1.40E-03	7.59E-04	1.23E−03	1.87E+05	0.00029	4.17E-04	3.57E-03	5.19E-05	7.8E-05

Table 12. Comparative statistical analysis for tension/compression spring design problem.

Algorithms	Proposed	CSA	ABC	ACO	EWA	GSA	KHA	MBO	PSO	SCA
$T_s\left(x_1\right)$	0.778186	12.4507	0.8881	0.7841	2.007	1.0858	0.7917	0.7944	0.77896	0.81758
$T_h\left(x_2\right)$	0.38466	6.15439	0.4313	0.3874	5.3999	0.94961	0.3961	0.4233	0.38468	0.41793
$R\left(x_3\right)$	40.3103	40.3196	45.930349	40.808465	59.0617	49.3452	41.033723	41.345423	40.3209	41.7494
$L\left(x_4\right)$	199.9898	200	139.73094	193.30955	166.76476	169.487	190.68425	194.39631	200	183.573
$f\left(\overrightarrow{x}\right)$	5885.228	5885.3	6280.4471	5886.9648	52642.982	11550.3	5930.2521	6179.5732	5891.39	6137.37

Table 13. Comparative analysis for pressure vessel design problem.

$$0.05 \leq\ x_1 \leq\ 2.00, 0.25 \leq\ x_2 \leq\ 1.3, 2 \leq\ x_3 \leq\ 15 \tag{32}$$

The tension/compression spring design results from mCSAMWL and other algorithms are shown in Table 11. The results illustrate that the mCSAMWL algorithm works better than the other state-of-the-art algorithms. Table 12 shows that the suggested mCSAMWL algorithm for the tension/compression spring design problem got similar results after a very small number of function evaluations.

Pressure vessel design

By optimizing four variables, the pressure vessel design problem's objective is to reduce the cost of fabrication by satisfying four constraints. The design variables consist of the thickness of the head (T_h), the length of the section without a head (L), thickness of the shell (T_s), and the inner radius (R). The problem can be modeled mathematically as in following Eqs. (33–39).

Consider

$$\overrightarrow{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \tag{33}$$

Minimize

$$f\left(\overrightarrow{x}\right) = 0.6224x_1x_3x_4 + 1.7881x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{34}$$

Subject to

$$z_1\left(\overrightarrow{x}\right) = -x_1 + 0.0193x_3 \leq\ 0 \tag{35}$$

$$z_2\left(\overrightarrow{x}\right) = -x_2 + 0.0193x_3 \leq\ 0 \tag{36}$$

$$z_3\left(\overrightarrow{x}\right) = -\pi\ x_3^2x_4 - \frac{4}{3}\pi\ x_3^3 + 1,296,000 \leq\ 0 \tag{37}$$

$$z_4\left(\overrightarrow{x}\right) = x_4 - 240 \leq\ 0 \tag{38}$$

Design variables range

$$0 \leq\ x_1, x_2 \leq\ 99, 10 \leq\ x_2, x_4 \leq\ 200 \tag{39}$$

Algorithms	Proposed	CSA	ABC	ACO	EWA	GSA	KHA	MBO	PSO	SCA
Best	5885.228	5885.33	6280.4471	5886.9648	52642.982	11550.3	5930.2521	52642.982	5891.39	6137.37
Mean	5887.867	5886.7	7595.104	6235.0563	377087.01	23342.3	6370.5053	377087.01	6531.5	6326.76
Worst	6645.244	5887.01	8984.3704	7264.8084	824178.13	33226.3	7451.2312	824178.13	7394.59	6512.35
Std. dev.	2.4009	2.2501	664.42143	345.23833	209370.59	5790.63	277.87154	209370.59	534.12	126.609

Table 14. Comparative statistical analysis for pressure vessel design problem.

Table 13 presents a summary of the best results that the mCSAMWL algorithm and the other commonly used metaheuristic algorithms delivered when solving the problem of designing pressure vessels. Table 13 illustrates that the proposed mCSAMWL algorithm delivered superior outcomes compared to the rest of the algorithms. Table 14 illustrates the results of a statistical analysis carried out on the algorithms used to address the pressure vessel design problem. In summary, the mCSAMWL algorithm delivered the most accurate solutions to the pressure vessel design problem while exhibiting the smallest degree of standard deviation.

Application using mCSAMWL algorithm for balance clustering in WSN

Let $T=\{t_1,t_2,\ldots t_z\}$ be the set of z sensors in the region of interest. by $t_z=(x_i,y_i)\in R^2$ indicates the location of the sensor T_z. In order to save energy, sensor nodes are organised into clusters[38], with one node in each cluster acting as the cluster head. It is the job of the cluster head to gather data from the other nodes in the cluster and send it on to the base station $bstCL=\{cl_1,cl_2,\ldots .cl_y\}\subset S$. denotes the subset of sensors chosen as cluster heads[38]. Sensor t_i i is a member of cluster $cl\left(t_i\right)$,which is given by Eq. (40).

$$cl\left(t_i\right)=arg_{cl_j}min \tag{40}$$

where indicates the Euclidean distance between sensors $t_i$and cl_j. For a cluster, the sensor subset Q_j, Cluster head h_j is defined as Eq. (41).

$$Q_j=\{t_i|\,t_i\in T,cl\left(t_i\right)=cl_j\} \tag{41}$$

This research optimizes the three most recurrent functions. First, the objective function is the average intra cluster distance. If the sensor nodes and cluster head are closer, less energy is needed to transmit data between them. It is represented in Eq. (42).

$$P_{DsCH}=\sum\nolimits_{i=1}^{N}D_{s_i}^{ch^k} \tag{42}$$

where, $D_{s_i}^{ch^k}$ denotes the Euclidean distance between sensor nodes and cluster heads

Balance cluster formation is the second objective function to consider, as it depends on the node degree, i.e. number of nodes that are associated with the CHs. It can be achieved by considering the average distance between the CHs. The distance between cluster heads should be maximum to attain disbursement of the clusters throughout the network. It is represented below in Eq. (43).

$$P_{DCH}=\frac{\sum\nolimits_{i=1}^{K}\sum\nolimits_{j=1}^{K}D_{ch_j}^{ch_i}}{K} \tag{43}$$

where, $D_{ch_j}^{ch_i}$ denotes the distance between cluster heads.

The average cluster head-to-base station distance is the third objective function. The shorter the distance between CHs and BS, the more likely it is that a node closer to BS will be chosen as a CH because it will take less energy to send all the data to BS as given in Eq. (44).

$$P_{CHBS}=\frac{\sum\nolimits_{i=1}^{M}dis\left(CH_i,BS\right)}{K} \tag{44}$$

where, the distance from the cluster head to the base station is denoted by $dis\left(CH_i,BS\right)$,K represents number of cluster heads. So, P_{CHBS} is expressed as in Eq. (45).

$$f=\phi_1P_{DsCH}+\phi_2P_{DCH}+\phi_3P_{CHBS} \tag{45}$$

where, $\phi_1,\phi_{12}\wedge\ \phi_3$ are the weighted coefficients such that, $\phi_1+\phi_2+\phi_3=1$

Results and discussion

In this research, we present an efficient method for selecting CHs using the modified Chameleon Swarm Optimization Algorithm (mCSAMWL) and a fitness function that considers average intra-cluster distance, average inter-cluster distance, and average cluster head to base station distance. Based on the area the WSN network covers, the algorithm comprises three distinct groups: WSN ~ 1 for 100 × 100, WSN ~ 2 for 200 × 200 m,

Parameter	Value
Nodes in the network area	$100\,m \times 100\,m,\ 200\,m \times 200\,m,\ 300\,m \times 300\,m$
Number of rounds	20,000
Number of cluster heads	10% of the network's total nodes
Number of nodes	G1# 100, G2 # 200, G#3 300
Base station position	Centre
Initial energy (E_0)	1 Joules
Receiving power (E_{Rx})	50 × 0.000000001nJ/bit/ m^2
Transmission power (E_{Tx})	50 × 0.000000001nJ/bit/ m^2
Data aggregation energy (E_{DA})	5 × 0.000000001nJ/bit/ m^2

Table 15. Parameter setting.

	WSN ~ 1 100 × 100			WSN ~ 2 200 × 200			WSN ~ 3 300 × 300		
Name of technique	G#1	G#2	G#3	G#1	G#2	G#3	G#1	G#2	G#3
ASO	0.0404	0.0381	0.0370	0.1505	0.1513	0.1446	0.3171	0.2835	0.3174
PSO-GWO	0.0415	0.0386	0.0372	0.1560	0.1278	0.1451	0.4532	0.3855	0.4099
BES	0.0390	0.0366	0.0357	0.1343	0.1258	0.1431	0.3800	0.3203	0.3291
AVOA	0.0388	0.0356	0.0340	0.1282	0.1093	0.0978	0.3678	0.3164	0.3029
CSA	0.0394	0.0365	0.0353	0.1480	0.0715	0.0672	0.1969	0.1249	0.1937
Proposed	0.0387	0.0351	0.0336	0.0738	0.0594	0.0539	0.1340	0.1015	0.0925

Table 16. Comparison based on average energy consumption for different network scenarios.

and WSN ~ 3 for 300 × 300 m. Each of these groups corresponds to a different size by varying the number of sensor nodes. G#1 consists of 100 nodes, G#2 consists of 200 nodes, and G#3 consists of 300 nodes, respectively. The parameter setting for the WSN network is shown in Table 15. In this section, the performance of six commonly used techniques, the Atom Search Optimization (ASO)[39], the Hybrid Particle Swarm Optimization and Grey Wolf Optimization (PSO-GWO)[40], the African Vulture Optimization Algorithm (AVOA)[41], the Bald Eagle Search Algorithm (BES)[42,43], and the Chameleon Swarm Algorithm (CSA)[25], are assessed in light of simulation parameters, average energy consumption, residual energy of the network, total energy consumption, dead nodes, and cluster head frequency. The simulation was run 20,000 times to determine which nodes were alive and which were deceased, while 1,000 rounds of execution were performed to assess the metrics for the performance of the algorithms mentioned above.

Simulation parameters
The effectiveness of the proposed clustering technique has been determined using the simulated parameters listed below:

(a) Average Energy Consumption: It determines the mean gap between every sensor node's starting and ending energy levels. To put it another way, it's the amount of energy that each node in a WSN network uses every round to send and receive packets of data.
(b) Total Energy Consumption: The energy dissipation of a network over a single round is the total amount of power consumed by the network's nodes during that round.
(c) Total Residual Energy: The total amount of residual energy is equal to the sum of the energies currently present in each sensor node.
(d) Dead Node: It is defined as the number of nodes that died over time during the simulation.
(e) CH Frequency: The frequency at which the sensor nodes performed the duties of CH during a certain time frame. High frequency suggests a sensor node is regularly selected as a CH, while low frequency means it is not.

Average energy consumption evaluation
Table 16 illustrates the average energy consumption performance of metaheuristic algorithms for a range of network sizes as well as node densities. Clustering techniques using the mCSAMWL algorithm consume less energy on average than other algorithms. Table 14 clearly shows the clustering technique using the mCSAMWL algorithm has the lowest average energy consumption in all network scenarios. The performance of different algorithms based on average energy consumption is depicted in Figs. 4 and 5. This technique outperforms its rivals' algorithms in each of the nine scenarios. The PSO-GWO algorithm performs the worst, followed by the ASO algorithm. The standard CSA algorithm has also demonstrated subpar performance compared to the mCSAMWL-based clustering method.

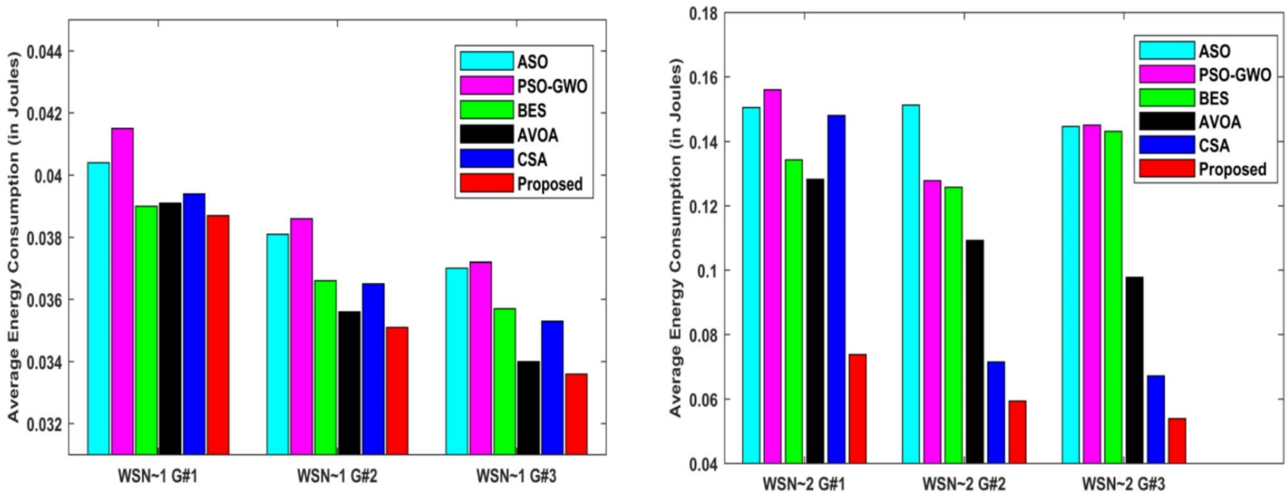

Fig. 4. Comparative analysis of average energy consumption for WSN scenario-1 &WSN scenario-2.

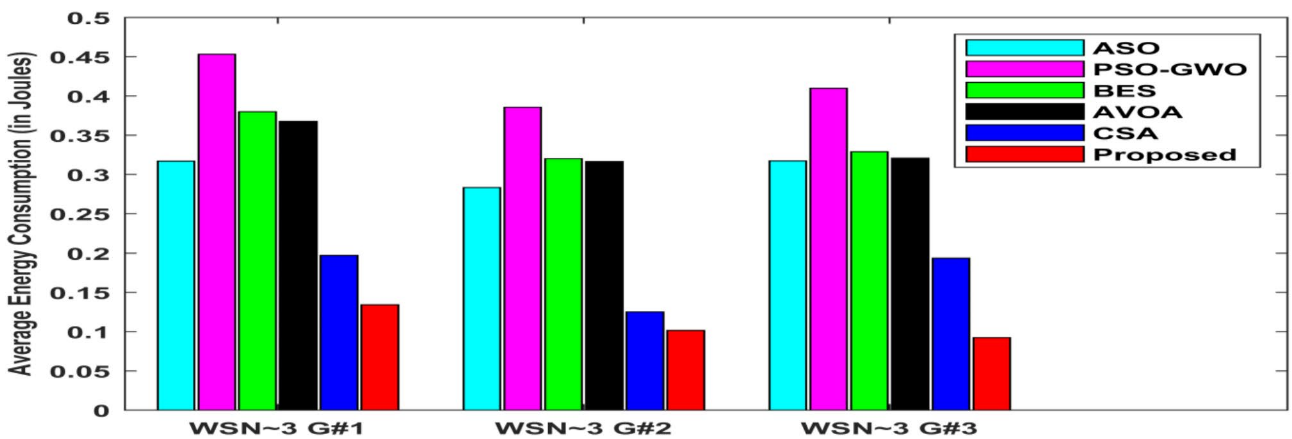

Fig. 5. Comparison based on average energy consumption for WSN scenario-3.

For WSN ~ 1, the average energy consumption of clustering techniques based on the mCSAMWL algorithm is 0.0387, 0.0351, and 0.0336 joules which is 1.77%, 3.83% and 4.81% lesser than CSA technique. Also, the respective values for WSN ~ 2 are 0.0738, 0.0594, and 0.0539 joules which are 50.1%, 59.8% and 19.7% better than CSA technique. Moreover, with values of 0.1340, 0.1015, and 0.0925 joules proposed technique consumes lesser average energy than CSA technique by 31.9%, 18.7%, and 52.2% respectively. These results reveal minimal energy consumption variance across all the mCSAMWL algorithm-based clustering scenarios. It is not the case for ASO[39], PSO-GWO AVOA[41], BES[42,43], CSA[25] techniques.

Total energy consumption evaluation

Figures 6, 7, 8, 9 and 10 depicts the total energy consumption of the clustering technique using the mCSAMWL algorithm in comparison to the remaining techniques ASO[39], PSO-GWO AVOA[41], BES[42,43], CSA[25] for 1000 iterations. Based on Table 17, PSO-GWO and ASO are the worst performers in terms of total energy consumption, followed by the BES technique. In all the scenarios considered, the mCSAMWL algorithm for the clustering technique performs optimally, followed by the CSA and AVOA techniques. Compared to the mCSAMWL algorithm-based clustering technique, the total amount of energy consumed by other techniques is significantly higher. The faster the sensor node depletes its energy, the sooner the WSN network will collapse. Whereas, in the case of the mCSAMWL algorithm-based clustering technique, the WSN network will last longer due to its low energy consumption compared to the other techniques.

In the WSN ~ 1 scenario, the total energy consumption of the clustering technique based on the mCSAMWL algorithm is 7.7373, 6.9473, and 6.6526 joules which is lesser than 1.62% in G#1, 6.25% in G#3 scenario of CSA technique but in case of G#2 scenario, it is 5.64% higher as compared to CSA technique. For the WSN ~ 2 scenario, the respective values are 14.7756, 12.0804, and 11.6942 joules which are 50.7%, 15.7% and 16.2% better than CSA technique. Furthermore, with values of 26.8223, 20.4252, and 17.6456 joules, mCSAMWL gives more efficient results in WSN ~ 3 than CSA technique by 44.3%, 31.9%, and 1%. Compared with the other

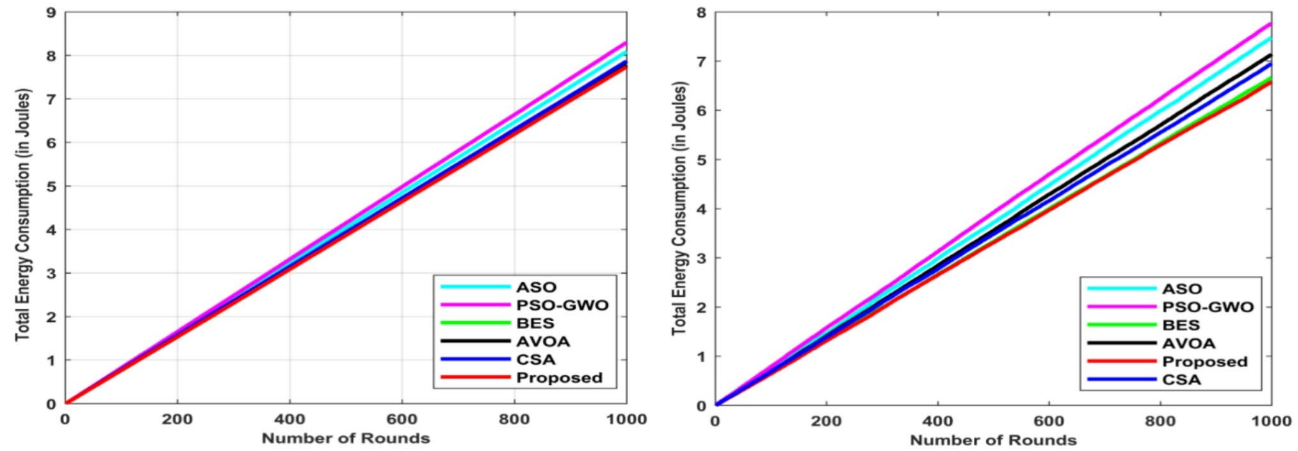

Fig. 6. Comparison based on Total Energy Consumption for WSN ~ 1 G#1 and WSN ~ 1 G#2.

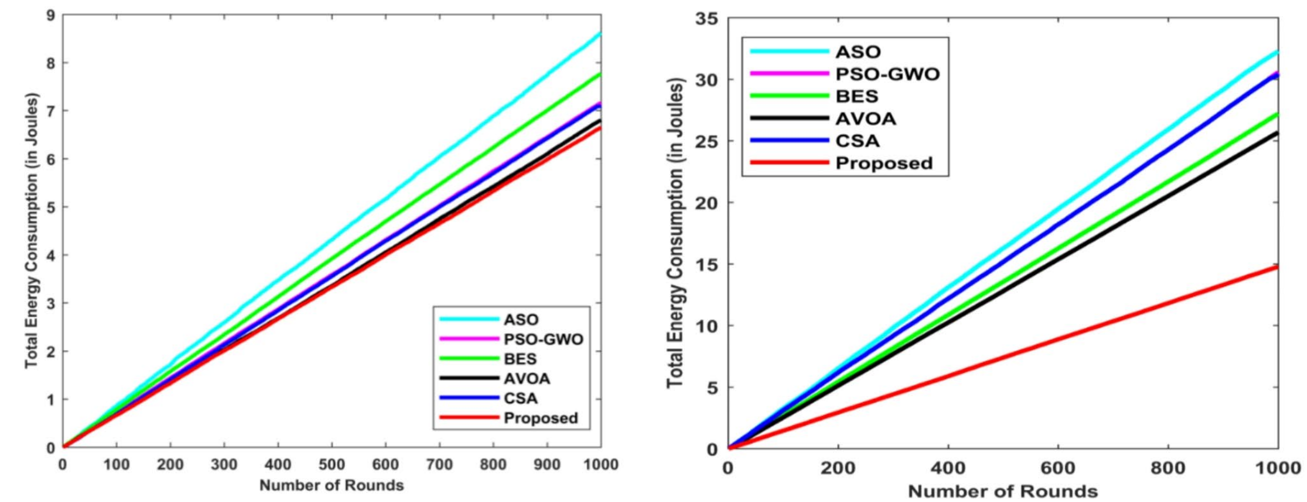

Fig. 7. Comparison based on total energy consumption for WSN ~ 1 G#3 and WSN ~ 2 G#1.

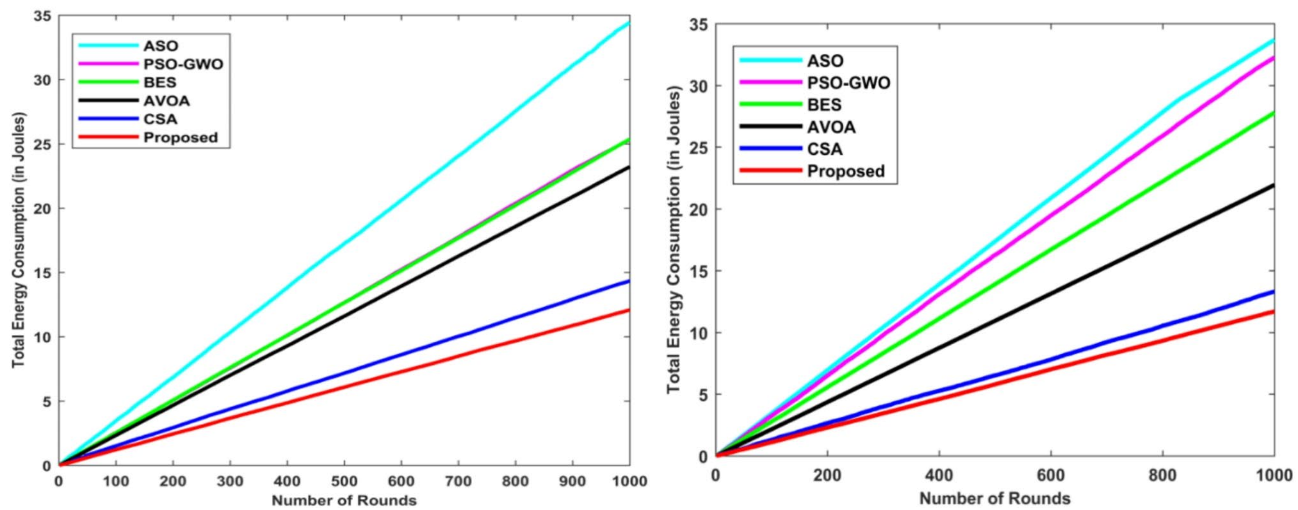

Fig. 8. Comparison based on total energy consumption for WSN 2 G#2 and WSN ~ 2 G#3.

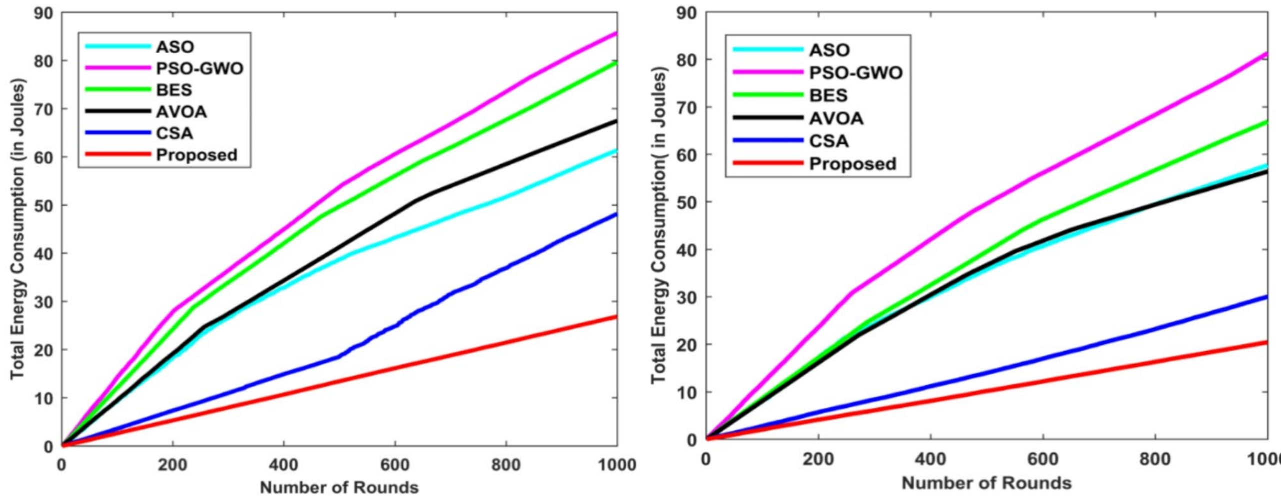

Fig. 9. Comparison based on total energy consumption for WSN ~ 3 G#1 and WSN ~ 3 G#2.

techniques, the clustering technique based on the mCSAMWL algorithm demonstrated the least variation in energy consumption across all scenarios.

Total residual energy

Table 18 depicts residual network energy after 1000 iterations for various scenarios. The mCSAMWL algorithm-based clustering technique outperforms the others in terms of residual energy, as shown in Table 18 The more energy that remains in the network, the longer it will last. Table 18 shows that the mCSAMWL algorithm-based clustering technique has the most residual energy left for all WSN scenarios.

As a result, the lifespan of the WSN network is prolonged. The PSO-GWO technique is the worst performer, followed by the ASO technique. The AVOA technique excelled in BES, ASO, and PSO-GWO, but it was unable to compete with the CSA and mCSAMWL algorithm-based clustering techniques.

Dead nodes

Table 19 presents the details of dead nodes in six algorithms based on the number of rounds. It represents the number of nodes that have some amount of energy that correlates with the number of rounds that have been completed. At the end of 4000 rounds, the mCSAMWL algorithm-based clustering method had no dead nodes, while other techniques did. ASO has the highest number of dead nodes, followed by PSO-GWO, BES, and CSA. It has been noted that the node lifespan has been extended in the clustering technique employing the mCSAMWL algorithm when compared to the ASO[39], PSO-GWO AVOA[41], BES[42,43], CSA[25] techniques.

Similarly, after 18,000 rounds, the CSA technique has 56 dead nodes, followed by AVOA, PSO-GWO, CSA, BES, and ASO. With only 30 dead nodes after 18,000 iterations, the mCSAMWL algorithm-based clustering technique clearly stands well ahead of other techniques.

Cluster head frequency

To ensure that each sensor node draws on a comparable amount of energy, the cluster head's responsibility must be equally divided among sensor nodes. The frequency with which a node in a given network size and density becomes the cluster head throughout the duration of the first 1000 simulation iterations is depicted in Figs. 11, 12, 13, 14, 15, 16, 17 and 18, and 19. It is clear that the behavior of many techniques changes as network size or density changes. The mCSAMWL-based clustering technique has demonstrated remarkable consistency in selecting sensor nodes to serve as cluster heads. It has been accomplished by distributing the responsibility of cluster head throughout the WSN network and maintaining small oscillations around the average cluster head frequency for all network densities. PSO-GWO, ASO, and BES techniques are the worst performers in terms of cluster head frequency parameters.

The proposed modified metaheuristic mCSAMWL algorithm applies Morlet wavelet mutation and Lévy Flight distribution as a different approach to solving optimization challenges. These modifications have made the standard CSA algorithm more effective and assisted in achieving a better equilibrium between the exploitation and exploration phases. The proposed mCSAMWL algorithm's performance has been assessed using 97 benchmark functions and three real-world engineering design problems. Based on the encouraging outcomes, the proposed mCSAMWL method has been implemented for clustering in WSN. The clustering technique using the proposed mCSAMWL algorithm excels over the original CSA and other clustering techniques in terms of average energy consumption, residual energy of the network, total energy consumption, dead nodes, and cluster head frequency. This technique performs extremely well in all network scenarios with variable node densities. The incorporation of Morlet wavelet and Lévy Flight into the existing standard CSA algorithm has improved the capabilities of the original CSA Algorithm.

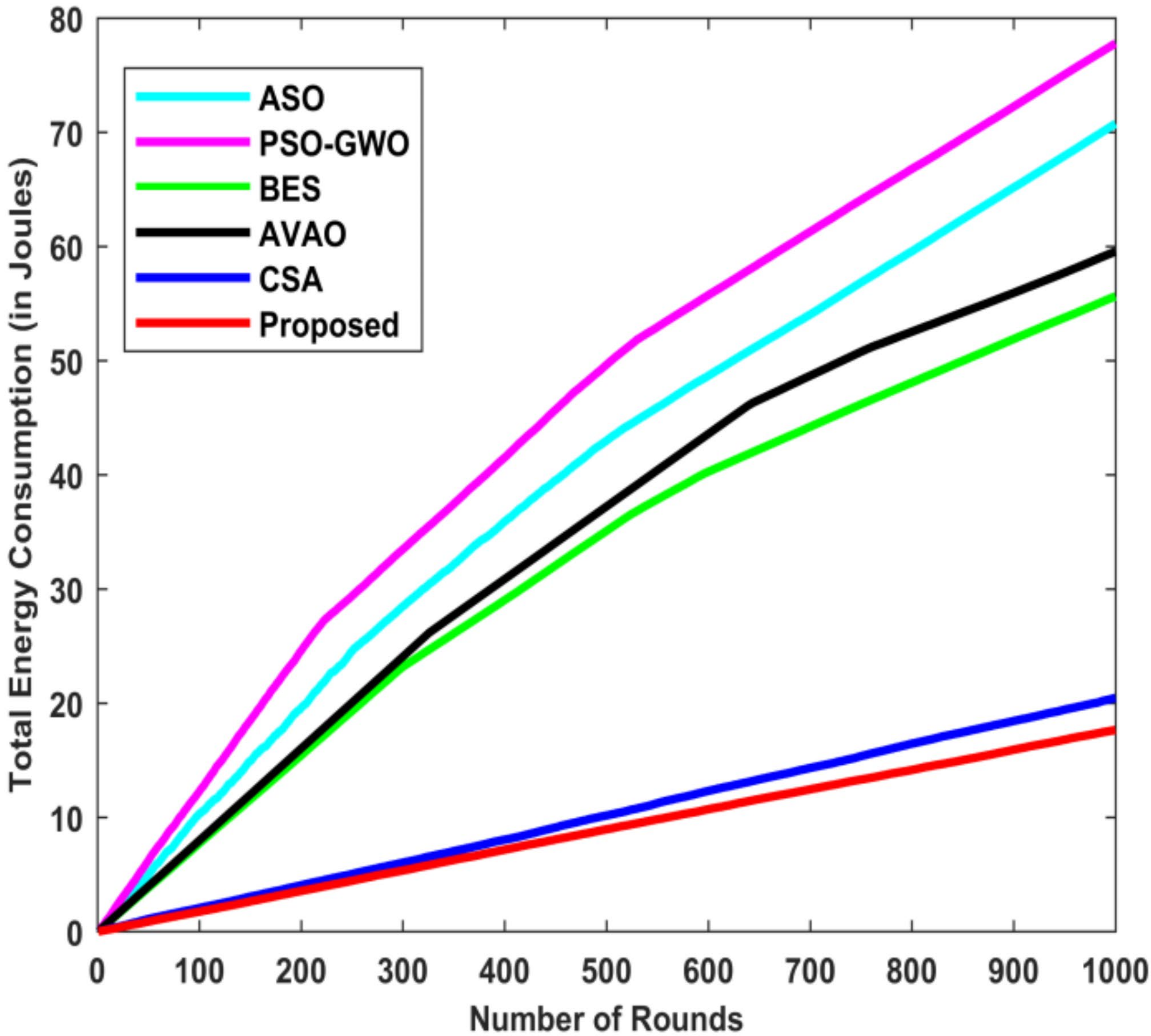

Fig. 10. Comparison based on total energy consumption for WSN ~ 3 G#3.

Name of Technique	WSN ~ 1 100 × 100			WSN ~ 2 200 × 200			WSN ~ 3 300 × 300		
	G#1	G#2	G#3	G#1	G#2	G#3	G#1	G#2	G#3
ASO	8.0839	7.4727	8.6137	30.4609	34.4287	33.6945	61.3471	57.6814	70.7377
PSO-GWO	8.2962	7.7676	7.1644	31.7030	25.3171	32.2717	85.6833	78.0320	77.7704
BES	7.7904	6.6607	7.0981	27.1928	25.3527	22.8524	73.0381	66.8577	55.6158
AVOA	7.7757	7.1381	6.8028	25.6898	23.2069	21.9429	67.4622	56.3751	59.5466
CSA	7.8654	6.5761	7.0950	30.0110	14.3342	13.9598	48.1628	30.0099	17.8225
Proposed	7.7372	6.9473	6.6526	14.7756	12.0804	11.6942	26.8223	20.4252	17.6456

Table 17. Comparison based on total energy consumption for different network scenarios.

Conclusions and recommendations

Metaheuristic algorithms have gained popularity as a fast and effective way to solve optimization problems. To overcome the limitations of the existing studies in this area, this work proposes, develops, and applies a modified, better performing Chameleon Swarm Algorithm incorporating Morlet wavelet and Lévy Flight distribution to enhance the efficacy of the standard CSA algorithm. The Morlet wavelet mutation is used to enhance the

Name of technique	WSN ~ 1 100 × 100			WSN ~ 2 200 × 200			WSN ~ 3 300 × 300		
	G#1	G#2	G#3	G#1	G#2	G#3	G#1	G#2	G#3
ASO	82.8148	169.0419	250.9480	63.9322	130.4940	201.6039	45.5858	104.7615	139.9848
PSO-GWO	84.1456	168.5606	254.6063	66.4208	141.2798	201.8322	24.1213	77.4302	103.0415
BES	83.8614	169.9365	251.4467	68.2203	142.0272	226.6893	30.3246	88.9584	133.9864
AVOA	83.1846	167.8084	251.9390	69.8836	146.2063	228.9982	36.7370	90.0454	141.4964
CSA	83.6342	167.6406	252.6286	67.2867	157.6655	238.8621	55.3708	142.4404	219.7570
Proposed	83.6283	167.7664	252.1614	77.1133	159.0475	241.9464	66.2082	144.8714	220.7759

Table 18. Comparison based on residual energy of network for different network scenarios.

Rounds	ASO	PSO-GWO	AVOA	BES	CSA	Proposed
2000	0	1	0	0	0	0
4000	5	4	1	4	4	0
6000	7	6	3	5	7	3
8000	10	8	5	8	9	7
10,000	13	11	9	9	11	9
12,000	16	14	16	13	19	13
14,000	20	17	24	17	29	15
16,000	29	23	36	28	39	22
18,000	45	49	50	48	56	30
20,000	63	79	80	66	69	52

Table 19. Comparative analysis in terms of dead nodes.

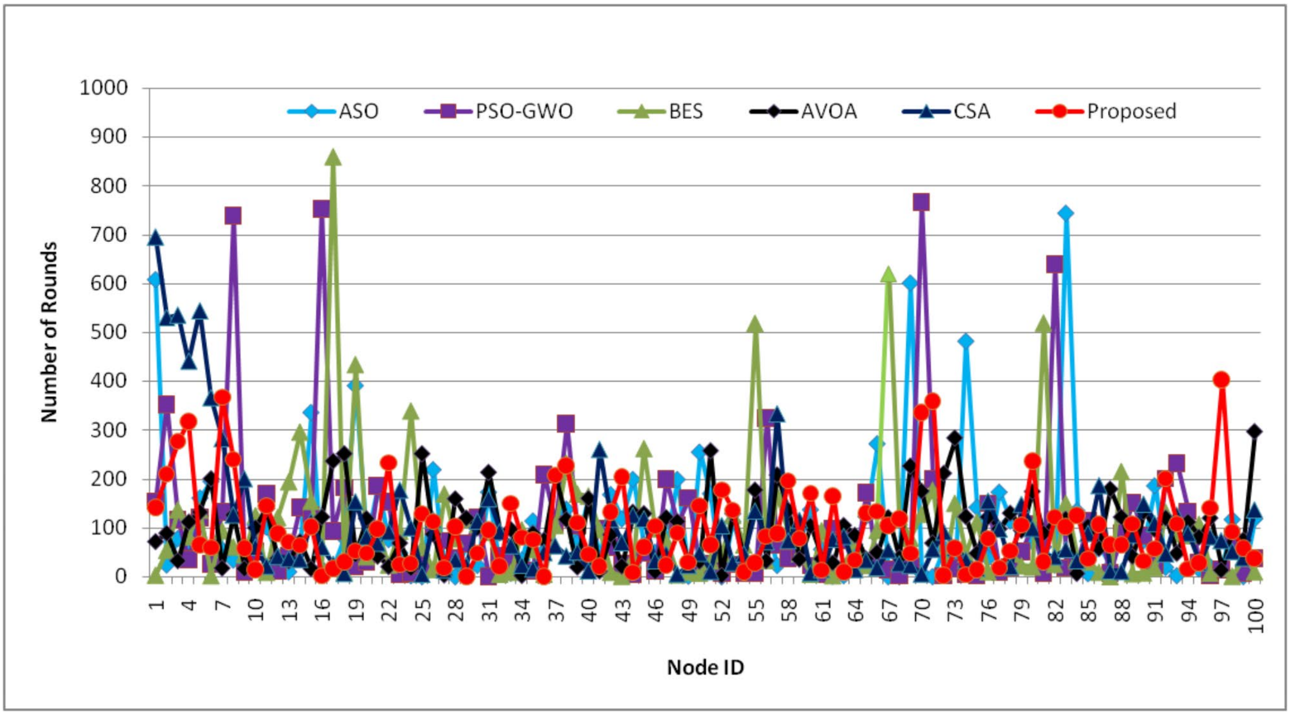

Fig. 11. Comparative analysis of cluster head frequency WSN ~ 1 G#1.

exploration phase of the CSA algorithm by exploring the entire search space and dividing it into two distinct regions. To improve the exploitation phase, the Lévy Flight distribution strategy with a step reducer factor is added to the normal CSA algorithm. So, the proposed algorithm applies changes to achieve an appropriate equilibrium between the exploration and exploitation phases. The proposed algorithm's efficacy is tested on 68

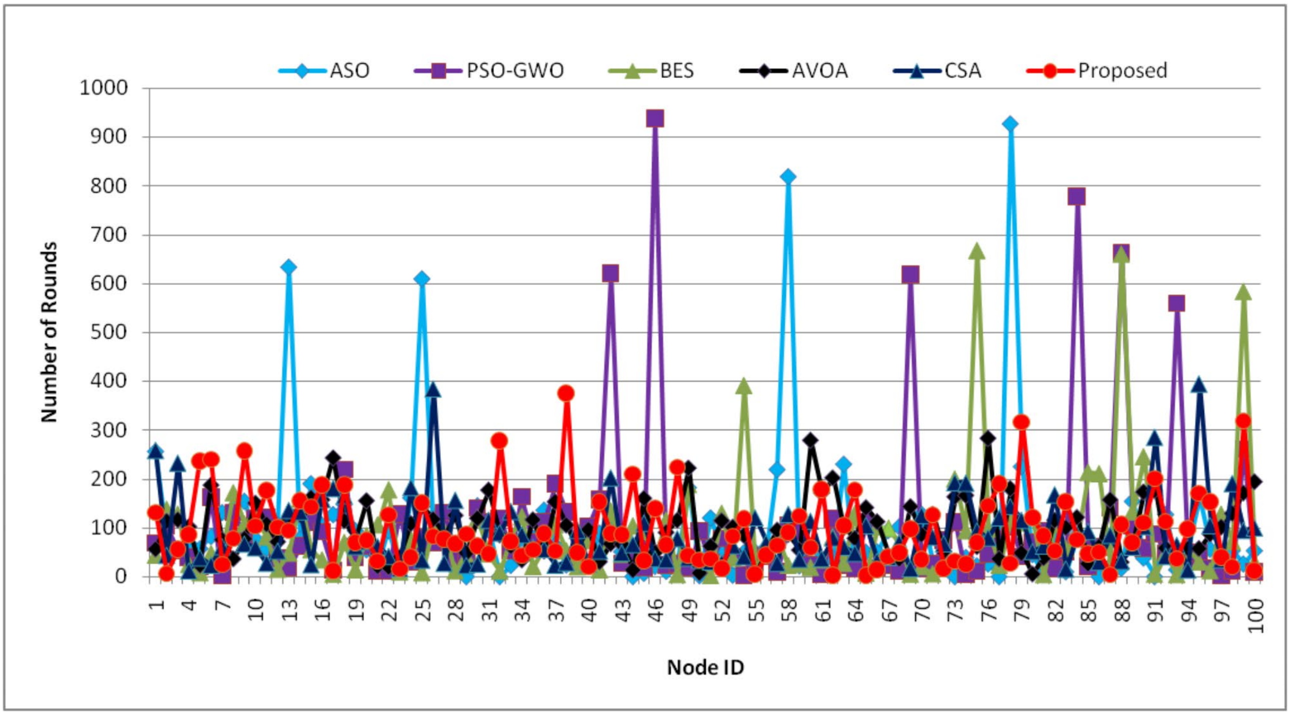

Fig. 12. Comparative analysis of cluster head frequency WSN ~ 1 G#2.

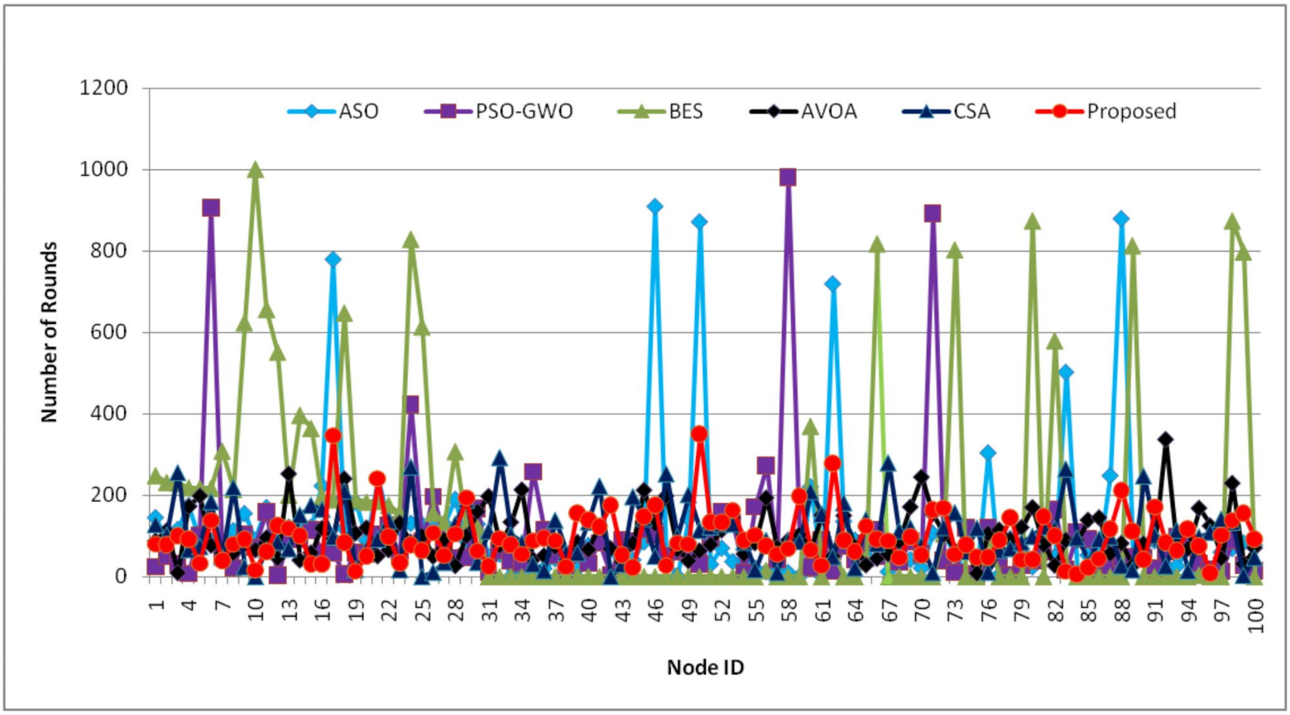

Fig. 13. Comparative analysis of cluster head frequency WSN ~ 1 G#3.

unimodal and multimodal benchmark functions and CEC 2017 test suite functions, and results are compared with 10 commonly used metaheuristic algorithms. The proposed mCSAMWL algorithm obtains the lowest Friedman mean rank, demonstrating its superiority over the other state-of-the-art algorithms.

Furthermore, the proposed algorithm has been used to effectively address three real-world engineering design problems. Finally, the proposed mCSAMWL algorithm has been applied for clustering in WSN to find

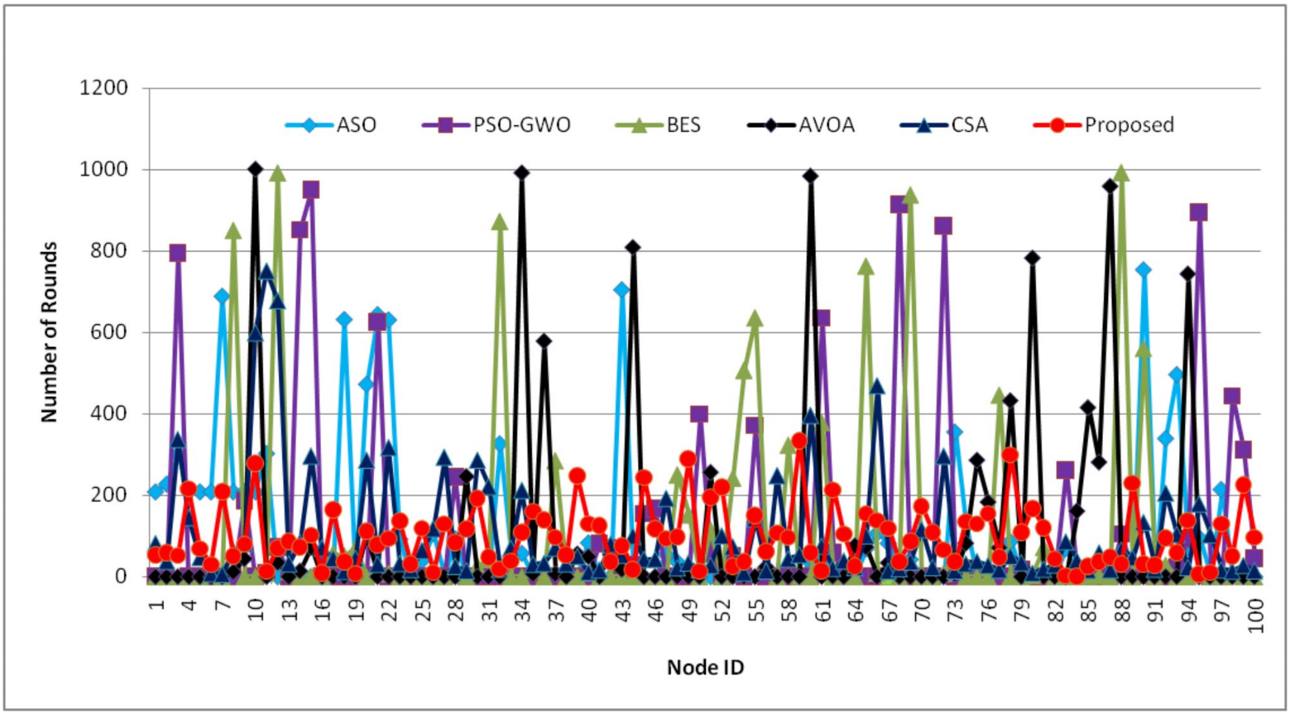

Fig. 14. Comparative analysis of cluster head frequency WSN ~ 2 G#1.

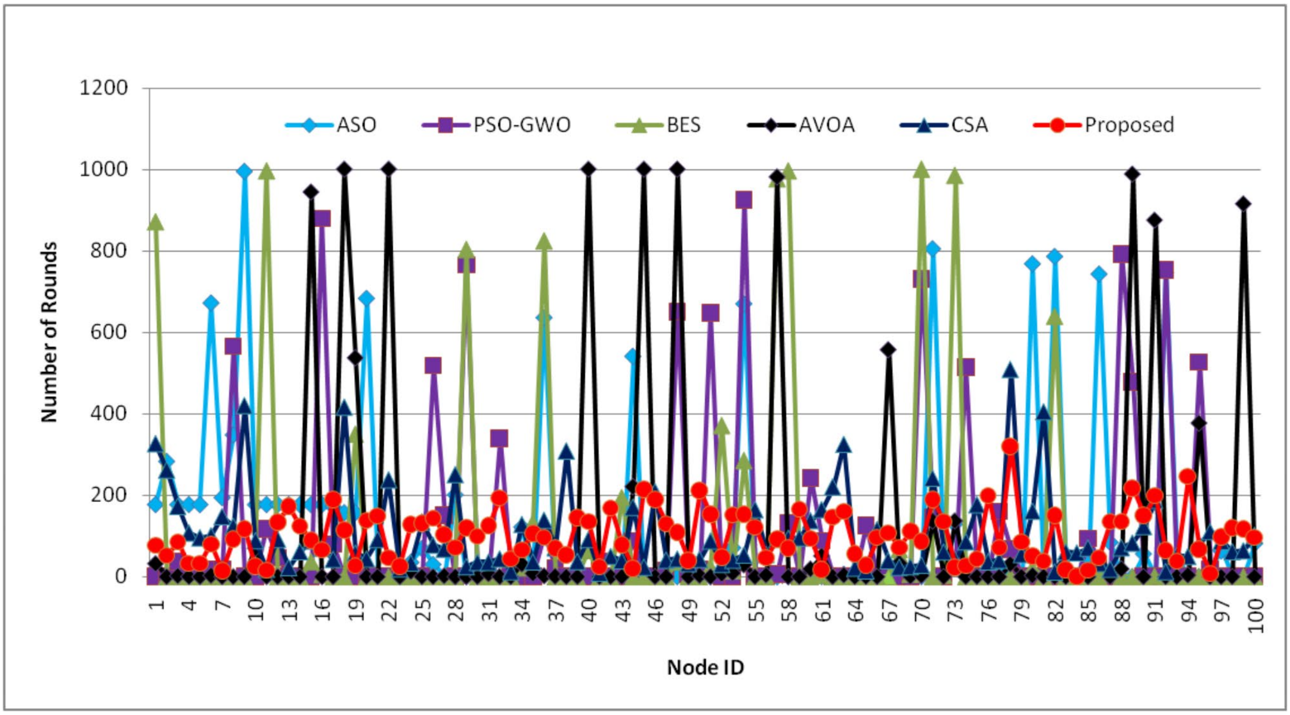

Fig. 15. Comparative analysis of cluster head frequency WSN ~ 2 G#2.

the optimal cluster head set and balance out the clustering process. The fitness function for this clustering technique uses average intra-cluster distance, average inter-cluster distance, and distance between cluster heads and the base station. This clustering technique has been thoroughly tested with three different WSN scenarios under varying node densities. The simulation performance of this technique has been computed against six commonly used metaheuristic techniques. From the experimental results, the clustering technique using the

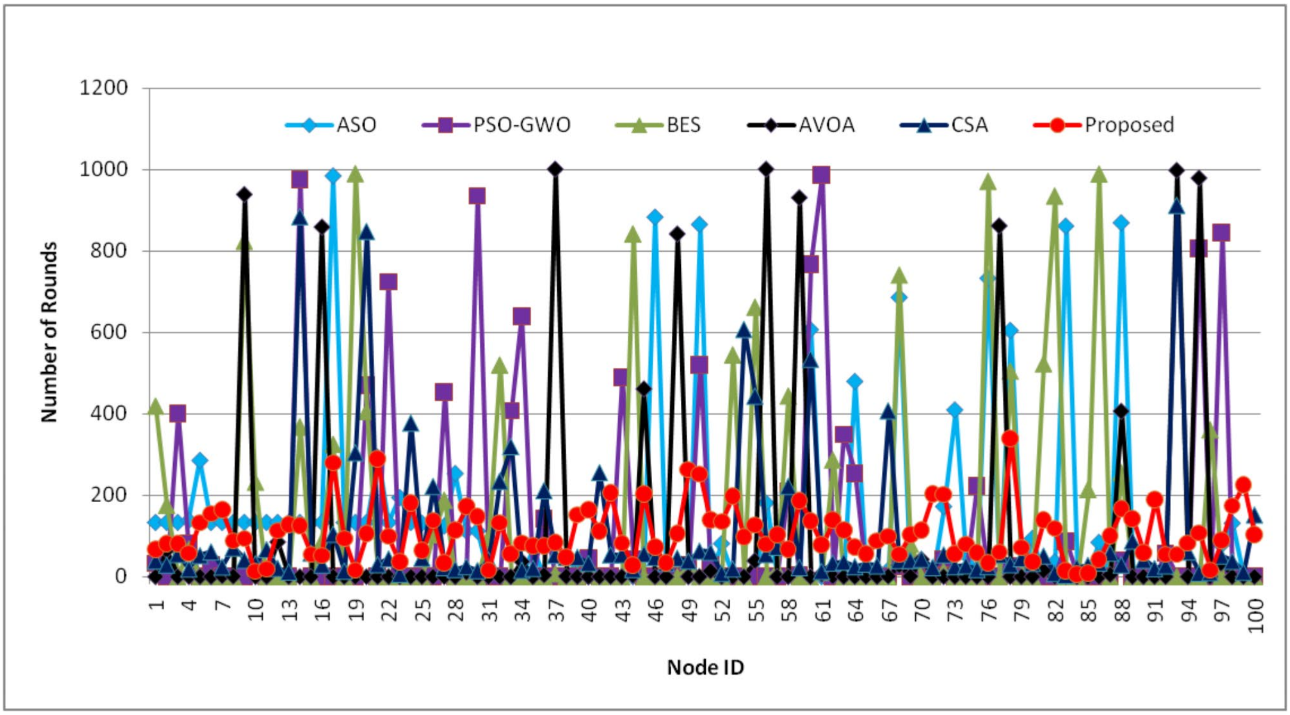

Fig. 16. Comparative analysis of cluster head frequency WSN ~ 2 G#3.

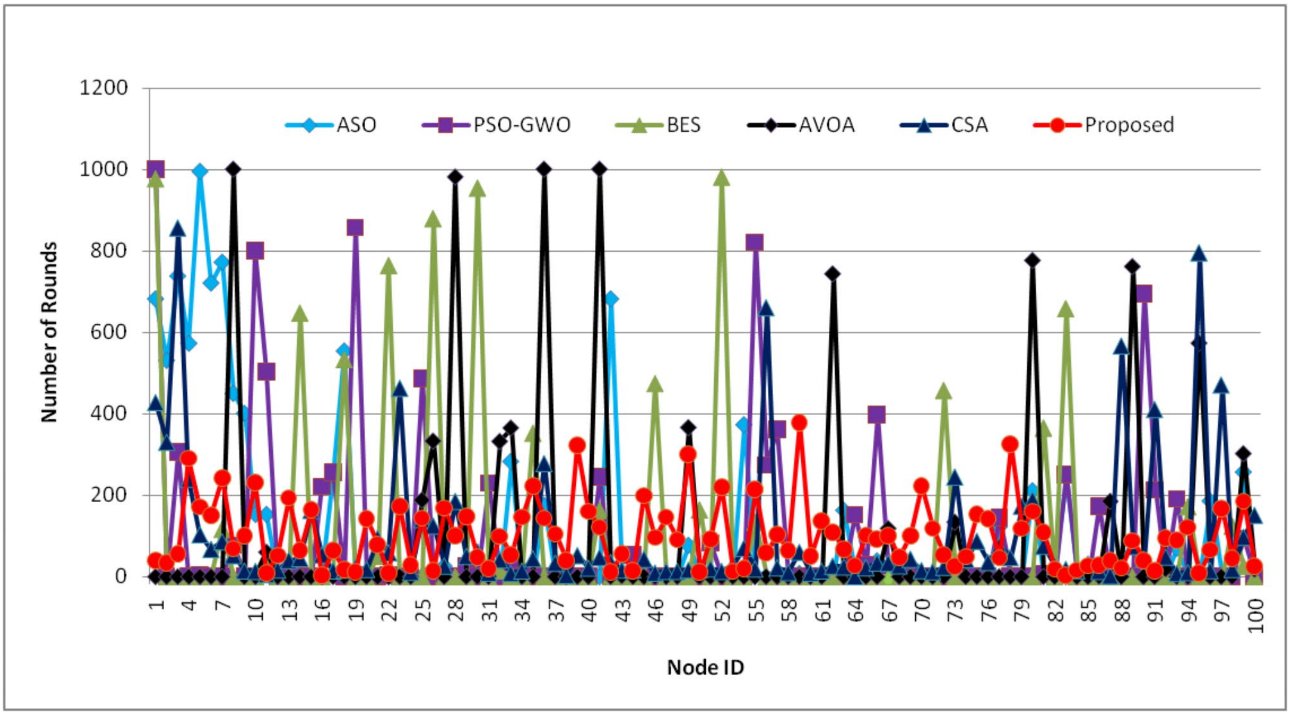

Fig. 17. Comparative analysis of cluster head frequency WSN ~ 3 G#1.

mCSAMWL algorithm outperforms the other technique in terms of average energy consumption, total energy consumption, residual energy, dead nodes and cluster head frequency. Significantly, the clustering technique using the mCSAMWL algorithm has resulted in increasing the lifetime of the WSN network by balancing out the cluster formation process and the average energy consumption of the sensor nodes. Further, the proposed improved algorithm can have applications to address various clustering, medical imaging, image segmentation,

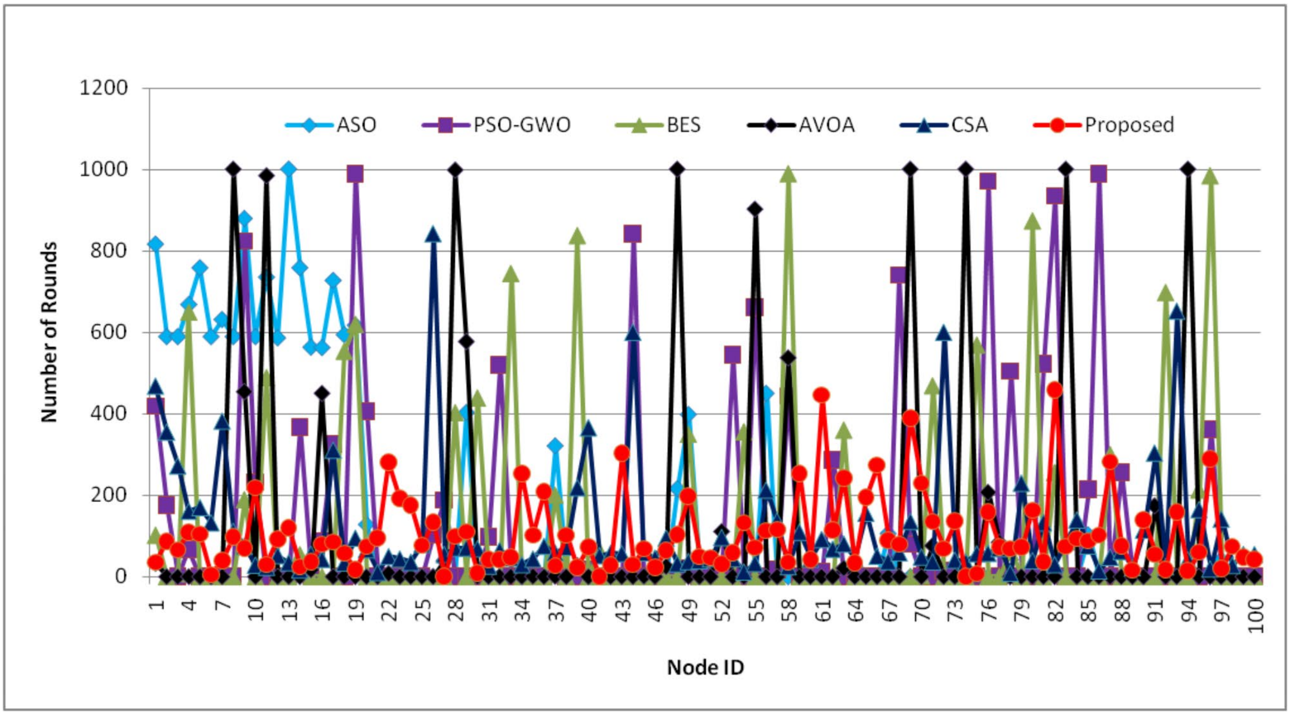

Fig. 18. Comparative analysis of cluster head frequency WSN ~ 3 G#2.

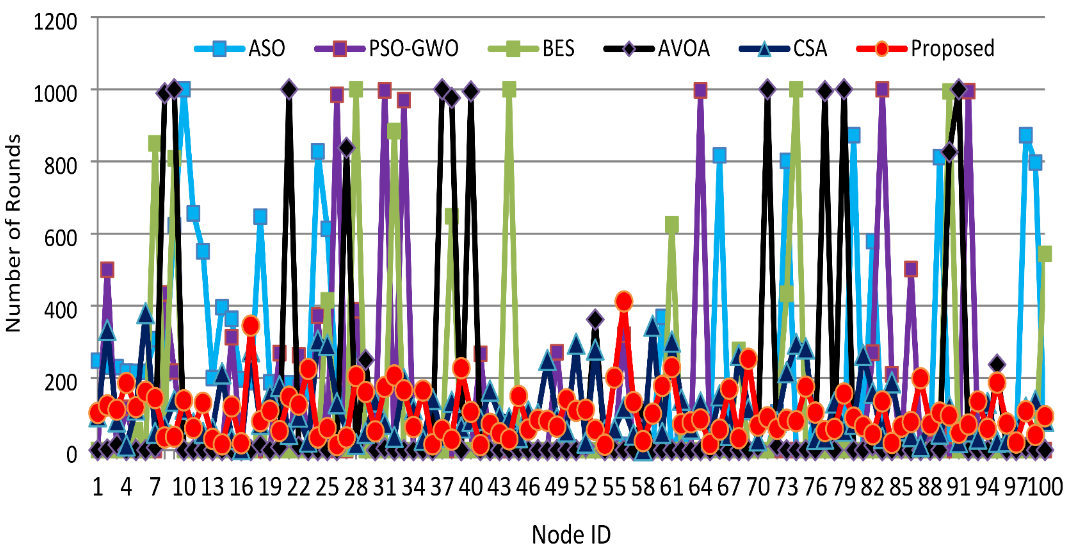

Fig. 19. Comparative analysis of cluster head frequency WSN ~ 3 G#3.

engineering design, data forecasting, classification, feature selection, and other real-world problems. As a future work, a variant of the mCSAMWL is being worked on to handle multi-objectives problems.

Data availability

The benchmark functions used in this research are publicly available and can be accessed from the CEC-BC-2017 dataset, referenced in[28,29]. https://www.kaggle.com/code/kooaslansefat/cec-2017-benchmark.

Appendix 1

In the following table, f. no. represents the function number, function name defines the name of the function, dim represents the number of dimensions (design variables) of the function, range defines the lower and upper bound of search space for the function, global value defines the global optimum value of the function.

F. no.	Function name	Dim	Range	Global value
Unimodal functions with fixed dimension				
F1	Beale	2	[-4.5, 4.5]	0
F2	Booth	2	[-10, 10]	0
F3	Brent	2	[-10, 10]	0
F4	Matyas	2	[-10, 10]	0
F5	Schaffer N. 4	2	[-100, 100]	0.292579
F6	Wayburn Seader 3	2	[-500, 500]	19.10588
F7	Leon	2	[-1.2, 1.2]	0
F8	Cube	2	[-10, 10]	0
F9	Zettl	2	[-5, 10]	-0.00379
Unimodal functions with variable dimensions				
F10	Sphere	30	[-100, 100]	0
F11	Power Sum	30	[-1, 1]	0
F12	Schwefel's 2.20	30	[-100, 100]	0
F13	Schwefel's 2.21	30	[-100, 100]	0
F14	Step	30	[-100, 100]	0
F15	Stepint	30	[-5.12, 5.12]	-155
F16	Schwefel's 2.22	30	[-100, 100]	0
F17	Schwefel's 2.23	30	[-10, 10]	0
F18	Rosenbrock	30	[-30, 30]	0
F19	Brown	30	[-1, 4]	0
F20	Dixon and Price	30	[-10, 10]	0
F21	Powell Singular	30	[-4, 5]	0
F22	Xin-She Yang	30	[-20, 20]	-1
F23	Perm 0, D, Beta	5	[-Dim, Dim]	0
F24	Sum Suqares	30	[-10, 10]	0
Multimodal functions with fixed- dimension				
F25	Egg Crate	2	[-5, 5]	0
F26	Ackley N.3	2	[-32, 32]	-195.629
F27	Adjiman	2	[-1, 2]	-2.02181
F28	Bird	2	[-2 π , 2 π]	-106.765
F29	Camel 6 Hump	2	[-5, 5]	-1.0316
F30	Branin RCOS	2	[-5, 5]	0.397887
F31	Goldstien Price	2	[-2, 2]	3
F32	Hartman 3	3	[0, 1]	-3.86278
F33	Hartman 6	6	[0, 1]	-3.32236
F34	Cross-in-tray	2	[-10, 10]	-2.06261
F35	Bartels Conn	2	[-500, 500]	1
F36	Bukin 6	2	[(-15, -5), (-5, -3)]	180.3276
F37	Carrom Table	2	[-10, 10]	-24.1568
F38	Chichinadze	2	[-30, 30]	-43.3159
F39	Cross function	2	[-10, 10]	0
F40	Cross leg table	2	[-10, 10]	-1
F41	Crowned Cross	2	[-10, 10]	0.0001
F42	Easom	2	[-100, 100]	-1
F43	Giunta	2	[-1, 1]	0.060447
F44	Helical Valley	3	[-10, 10]	0
F45	Himmelblau	2	[-5, 5]	0
F46	Holder	2	[-10, 10]	-19.2085
F47	Pen Holder	2	[-11, 11]	-0.96354
F48	Test Tube Holder	2	[-10, 10]	-10.8723
F49	Shubert	2	[-10, 10]	-186.731
F50	Shekel	4	[0, 10]	-10.5364
F51	Three-Hump Camel	2	[-5, 5]	0
Multimodal function with variable dimension				
F52	Schwefel's 2.26	30	[-500, 500]	-418.983
F53	Rastrigin	30	[-5.12, 5.12]	0

F. no.	Function name	Dim	Range	Global value
F54	Periodic	30	[-10, 10]	0.9
F55	Qing	30	[-500, 500]	0
F56	Alpine N. 1	30	[-10, 10]	0
F57	Xin-She Yang	30	[-5, 5]	0
F58	Ackley	30	[-32, 32]	0
F59	Trignometric 2	30	[-500, 500]	0
F60	Salomon	30	[-100, 100]	0
F61	Styblinski-Tang	30	[-5, 5]	-1174.98
F62	Griewank	30	[-100, 100]	0
F63	Xin-She Yang N. 4	30	[-10, 10]	-1
F64	Xin-She Yang N. 2	30	[-2 π , 2 π]	0
F65	Gen. Penalized	30	[-50, 50]	0
F66	Penalized	30	[-50, 50]	0
F67	Michalewics	30	[0, π]	-29.6309
F68	Quartic Noise	30	[-1.28, 1.28]	0
CEC-BC-2017 Functions				
F69	Shifted and Rotated Bent Cigar Function	10	[-100, 100]	100
F70	Shifted and Rotated Rosenbrock Function	10	[-100, 100]	300
F71	Shifted and Rotated Rastrigin Function	10	[-100, 100]	400
F72	Shifted and Rotated Expanded Scaffer's F6 Function	10	[-100, 100]	500
F73	Shifted and Rotated Lunacek Bi Rastrigin Function	10	[-100, 100]	600
F74	Shifted and Rotated Non-Continuous Rastrigin's Function	10	[-100, 100]	700
F75	Shifted and Rotated Levy Function	10	[-100, 100]	800
F76	Shifted and Rotated Schwefel's Function	10	[-100, 100]	900
F77	Hybrid Function 1 (N = 3)	10	[-100, 100]	1000
F78	Hybrid Function 2 (N = 3)	10	[-100, 100]	1100
F79	Hybrid Function 3 (N = 3)	10	[-100, 100]	1200
F80	Hybrid Function 4 (N = 4)	10	[-100, 100]	1300
F81	Hybrid Function 5 (N = 4)	10	[-100, 100]	1400
F82	Hybrid Function 6 (N = 4)	10	[-100, 100]	1500
F83	Hybrid Function 6 (N = 5)	10	[-100, 100]	1600
F84	Hybrid Function 6 (N = 5)	10	[-100, 100]	1700
F85	Hybrid Function 6 (N = 5)	10	[-100, 100]	1800
F86	Hybrid Function 6 (N = 6)	10	[-100, 100]	1900
F87	Composite Function 1 (N = 3)	10	[-100, 100]	2000
F88	Composite Function 2 (N = 3)	10	[-100, 100]	2100
F89	Composite Function 3 (N = 4)	10	[-100, 100]	2200
F90	Composite Function 4 (N = 4)	10	[-100, 100]	2300
F91	Composite Function 5 (N = 5)	10	[-100, 100]	2400
F92	Composite Function 6 (N = 5)	10	[-100, 100]	2500
F93	Composite Function 7 (N = 6)	10	[-100, 100]	2600
F94	Composite Function 8 (N = 6)	10	[-100, 100]	2700
F95	Composite Function 9 (N = 6)	10	[-100, 100]	2800
F96	Composite Function 10 (N = 3)	10	[-100, 100]	2900
F97	Composite Function 11 (N = 3)	10	[-100, 100]	3000

Received: 18 October 2024; Accepted: 1 April 2025
Published online: 22 April 2025

References

1. Iwendi, C. et al. A metaheuristic optimization approach for energy efficiency in the IoT networks. *Software: Pract. Experience.* **51**(12), 2558–2571 (2021).
2. Vennila, H. et al. Static and dynamic environmental economic dispatch using tournament selection based ant Lion optimization algorithm. *Front. Energy Res.* **10**, 972069. https://doi.org/10.3389/fenrg.2022.972069 (2022).
3. Wang, C. A distributed particle-swarm-optimization-based fuzzy clustering protocol for wireless sensor networks. *Sensors* **23**(15), 6699. https://doi.org/10.3390/s23156699 (2023).
4. Adegboye, O. R. et al. Refinement of dynamic hunting leadership algorithm for enhanced numerical optimization. *IEEE Access.* https://doi.org/10.1109/access.2024.3427812 (2024).

5. Adegboye, O. R. et al. DGS-SCSO: enhancing sand Cat swarm optimization with dynamic pinhole imaging and golden sine algorithm for improved numerical optimization performance. *Sci. Rep.* **14**(1), 1491. https://doi.org/10.1038/s41598-023-50910-x (2024).
6. Adegboye, O. R. & Deniz Ülker, E. Hybrid artificial electric field employing cuckoo search algorithm with refraction learning for engineering optimization problems. *Sci. Rep.* **13**(1), 4098. https://doi.org/10.1038/s41598-023-31081-1 (2023).
7. Sridharan, A. Chameleon swarm optimisation with machine learning based sentiment analysis on sarcasmdetection and classification model. *Int. Res. J. Eng. Technol.* **8**(10), 821–828 (2021).
8. Umamageswari, A., Bharathiraja, N. & Irene, D. S. A novel fuzzy C-means based chameleon swarm algorithm for segmentation and progressive neural architecture search for plant disease classification. *ICT Express*. https://doi.org/10.1016/j.icte.2021.08.019 (2021).
9. Rizk-Allah, R. M., El-Hameed, M. A. & El-Fergany, A. A. Model parameters extraction of solid oxide fuel cells based on semi-empirical and memory-based chameleon swarm algorithm. *Int. J. Energy Res.* **45**(15), 21435–21450. https://doi.org/10.1002/er.7192 (2021).
10. Anitha, S., Saravanan, S. & Chandrasekar, A. A modified Gray wolf-based chameleon swarm algorithm for minimizing energy consumption and enabling secure communication in wireless sensor network. *Concurrency Computation: Pract. Experience.* **34**(26), e7295. https://doi.org/10.1002/cpe.7295 (2022).
11. Rizk-Allah, R. M., Hassanien, A. E. & Snášel, V. A hybrid chameleon swarm algorithm with superiority of feasible solutions for optimal combined heat and power economic dispatch problem. *Energy* **254**, 124340. https://doi.org/10.1016/j.energy.2022.124340 (2022).
12. Mostafa, R. R., Ewees, A. A., Ghoniem, R. M., Abualigah, L. & Hashim, F. A. Boosting chameleon swarm algorithm with consumption AEO operator for global optimization and feature selection. *Knowl. Based Syst.* **246**, 108743. https://doi.org/10.1016/j.knosys.2022.108743 (2022).
13. Wang, J., Lv, M., Li, Z. & Zeng, B. Multivariate selection-combination short-term wind speed forecasting system based on convolution-recurrent network and multi-objective chameleon swarm algorithm. *Expert Syst. Appl.* **214**, 119129. https://doi.org/10.1016/j.eswa.2022.119129 (2023).
14. Hu, G., Yang, R., Qin, X. & Wei, G. MCSA: Multi-strategy boosted chameleon-inspired optimization algorithm for engineering applications. *Comput. Methods Appl. Mech. Eng.* **403**, 115676. https://doi.org/10.1016/j.cma.2022.115676 (2023).
15. Damin, Z., Yi, W. & Linna, Z. Chameleon swarm algorithm for segmental variation learning of population and S-type weight. *J. Syst. Simul.* **35**(1), 11. https://doi.org/10.16182/j.issn1004731x.joss.21-0968 (2023).
16. Braik, M. S., Awadallah, M. A., Al-Betar, M. A., Hammouri, A. I. & Zitar, R. A. A non-convex economic load dispatch problem using chameleon swarm algorithm with roulette wheel and levy flight methods. *Appl. Intell.* 1–40. https://doi.org/10.1007/s10489-022-04363-w (2023).
17. Hu, G., Yang, R. & Wei, G. Hybrid chameleon swarm algorithm with multi-strategy: A case study of degree reduction for disk Wang–Ball curves. *Math. Comput. Simul.* **206**, 709–769. https://doi.org/10.1016/j.matcom.2022.12.001 (2023).
18. Sun, Y. et al. Research on signal detection of OFDM systems based on the LSTM network optimized by the improved chameleon swarm algorithm. *Mathematics* **11**(9), 1989. https://doi.org/10.3390/math11091989 (2023).
19. Zhou, J. & Xu, Z. Optimal sizing design and integrated cost-benefit assessment of stand-alone microgrid system with different energy storage employing chameleon swarm algorithm: A rural case in Northeast China. *Renew. Energy*. **202**, 1110–1137 (2023).
20. Dinh, P-H. Medical image fusion based on enhanced three-layer image decomposition and chameleon swarm algorithm. *Biomed. Signal Process. Control.* **84**, 104740 (2023).
21. Abed-alguni, B. H. Island-based cuckoo search with highly disruptive polynomial mutation. *Int. J. Artif. Intell.* **17**(1), 57–82. https://doi.org/10.1007/s13369-020-05141-x (2019).
22. Abed-alguni, B. H. & Paul, D. Island-based cuckoo search with elite opposition-based learning and multiple mutation methods for solving discrete and continuous optimization problems. https://doi.org/10.1007/s00500-021-06665-6 (2021).
23. Abed-alguni, B. H., Alawad, N. A., Barhoush, M. & Hammad, R. Exploratory cuckoo search for solving single-objective optimization problems. *Soft. Comput.* **25**(15), 10167–10180. https://doi.org/10.1007/s00500-021-05939-3 (2021).
24. Abed-Alguni, B. H., Paul, D. & Hammad, R. Improved salp swarm algorithm for solving single-objective continuous optimization problems. *Appl. Intell.* **52**(15), 17217–17236. https://doi.org/10.1007/s10489-022-03269-x (2022).
25. Braik, M. S. Chameleon swarm algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Syst. Appl.* **174**, 114685. https://doi.org/10.1016/j.eswa.2021.114685 (2021).
26. Gao, Y., Zhang, H., Duan, Y. & Zhang, H. A novel hybrid PSO based on levy flight and wavelet mutation for global optimization. *Plos One*. **18**(1), e0279572. https://doi.org/10.1371/journal.pone.0279572 (2023).
27. Jagadeesh, S. & Muthulakshmi, I. Dynamic clustering and routing using multi-objective particle swarm optimization with levy distribution for wireless sensor networks. *Int. J. Commun Syst.* **34**(13). https://doi.org/10.1002/dac.4902 (2021).
28. Isiet, M. & Gadala, M. Sensitivity analysis of control parameters in particle swarm optimization. *J. Comput. Sci.* **41**, 101086 (2020).
29. Singh, H., Singh, B. & Kaur, M. An improved elephant herding optimization for global optimization problems. *Eng. Comput.* 1–33. https://doi.org/10.1007/s00366-021-01471-y (2021).
30. Mittal, H., Tripathi, A., Pandey, A. C. & Pal, R. Gravitational search algorithm: a comprehensive analysis of recent variants. *Multimedia Tools Appl.* **80**, 7581–7608. https://doi.org/10.1007/s11042-020-09831-4 (2021).
31. Fidanova, S. & Fidanova, S. Ant colony optimization. *Ant Colony Optim. Appl.* 3–8. https://doi.org/10.1007/978-3-030-67380-2_2 (2021).
32. Pasupuleti, V. & Balaswamy, C. Performance analysis of fractional earthworm optimization algorithm for optimal routing in wireless sensor networks. *EAI Endorsed Trans. Scalable Inform. Syst.* **8**, 32. https://doi.org/10.4108/eai.21-4-2021.169419 (2021).
33. Tian, Y. & Chang, Y. Application of the particle swarm optimization algorithm-back propagation neural network algorithm introducing new parameter terms in the application field of industrial design. *Results Eng.* https://doi.org/10.1016/j.rineng.2023.101728 (2023).
34. Gabis, A. B., Meraihi, Y., Mirjalili, S. & Ramdane-Cherif, A. A comprehensive survey of sine cosine algorithm: variants and applications. *Artif. Intell. Rev.* **54**(7), 5469–5540. https://doi.org/10.1007/s10462-021-10026-y (2021).
35. Sadrishojaei, M., Navimipour, N. J., Reshadi, M. & Hosseinzadeh, M. A new clustering-based routing method in the mobile internet of things using a Krill herd algorithm. *Cluster Comput.* 1–11 https://doi.org/10.1007/s10586-021-03394-1 (2022).
36. Kaya, E., Gorkemli, B., Akay, B. & Karaboga, D. A review on the studies employing artificial bee colony algorithm to solve combinatorial optimization problems. *Eng. Appl. Artif. Intell.* **115**, 105311. https://doi.org/10.1016/j.engappai.2022.105311 (2022).
37. Ghetas, M. Learning-based monarch butterfly optimization algorithm for solving numerical optimization problems. *Neural Comput. Appl.* 1–19. https://doi.org/10.1007/s00521-021-06654-8 (2022).
38. Wohwe Sambo, D., Yenke, B. O., Förster, A. & Dayang, P. Optimized clustering algorithms for large wireless sensor networks: A review. *Sensors* **19**(2), 322. https://doi.org/10.3390/s19020322 (2019).
39. Bi, J. & Zhang, Y. An improved atom search optimization for optimization tasks. *Multimedia Tools Appl.* **82**(5), 6375–6429. https://doi.org/10.1007/s11042-022-13171-w (2023).
40. Gul, F. et al. Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO–GWO optimization algorithm with evolutionary programming. *J. Ambient Intell. Humaniz. Comput.* **12**, 7873–7890. https://doi.org/10.1007/s12652-020-02514-w (2021).

41. Abdollahzadeh, B., Gharehchopogh, F. S. & Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Comput. Ind. Eng.* **158**, 107408. https://doi.org/10.1016/j.cie.2021.107408 (2021).
42. Kapileswar, N. & Phani Kumar, P. Energy efficient routing in IOT based UWSN using bald eagle search algorithm. *Trans. Emerg. Telecommunications Technol.* **33**(1), e4399. https://doi.org/10.1002/ett.4399 (2022).
43. Kusla, V. & Brar, G. S. MOBES: a modified bald eagle search based technique for optimal clustering in wireless sensor network. *Univ. Politehnica Buchar. Sci. Bull. Ser. C* **86**(1), 207–224 (2024).

Acknowledgements
The authors would like to acknowledge Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Automated Systems and Soft Computing Lab (ASSCL) at Prince Sultan University, Riyadh, Saudi Arabia, for their support to this work.

Author contributions
All authors reviewed the manuscript.V. Kusla: Conceptualization, Methodology, Formal analysis, Validation, Writing-original draft. GS. Brar: Formal analysis, Validation. Harpreet Kaur: Methodology, Formal analysis, Investigation, Data curation, Writing-review and editing. Ramandeep Sandhu: Conceptualization, Resources, Visualization, Supervision. Chander Prabha: Methodology, Validation. Md Rittique Alam: Writing-review and editing, Md. Mehedi Hassan: Supervision, Investigation, Writing-review and editing. Shahab Abdullah: Writing-review and editing. Samah Alshathri: Conceptualization, Resources, Funding, Supervision. Walid El-Shafai: Methodology, Validation, Visualization.

Funding
The authors would like to acknowledge Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Declarations

Competing interests
The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to G.S.B., M.M.H. or W.E.-S.