

Article

Three Chaotic Strategies for Enhancing the Self-Adaptive Harris Hawk Optimization Algorithm for Global Optimization

Sultan Almotairi ^{1,2,*} , Elsayed Badr ^{3,4,*} , Mustafa Abdul Salam ^{5,6}  and Alshimaa Dawood ³ ¹ Department of Computer Science, College of Computer and Information Sciences, Majmaah University, Al-Majmaah 11952, Saudi Arabia² Department of Computer Science, Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah 42351, Saudi Arabia³ Scientific Computing Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha 13518, Egypt; elshimaa.abdelfattah@fci.bu.edu.eg⁴ Data Science Department, Faculty of Computers and Information Systems, Egyptian Chinese University, Cairo 11786, Egypt⁵ Artificial Intelligence Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha 13518, Egypt; mustafa.abdo@fci.bu.edu.eg⁶ Faculty of Computer Studies, Arab Open University, Cairo 11211, Egypt

* Correspondence: almotairi@mu.edu.sa (S.A.); alsayed.badr@fci.bu.edu.eg (E.B.)

Abstract: Harris Hawk Optimization (HHO) is a well-known nature-inspired metaheuristic model inspired by the distinctive foraging strategy and cooperative behavior of Harris Hawks. As with numerous other algorithms, HHO is susceptible to getting stuck in local optima and has a sluggish convergence rate. Several techniques have been proposed in the literature to improve the performance of metaheuristic algorithms (MAs) and to tackle their limitations. Chaos optimization strategies have been proposed for many years to enhance MAs. There are four distinct categories of Chaos strategies, including chaotic mapped initialization, randomness, iterations, and controlled parameters. This paper introduces SHHOIRC, a novel hybrid algorithm designed to enhance the efficiency of HHO. Self-adaptive Harris Hawk Optimization using three chaotic optimization methods (SHHOIRC) is the proposed algorithm. On 16 well-known benchmark functions, the proposed hybrid algorithm, authentic HHO, and five HHO variants are evaluated. The computational results and statistical analysis demonstrate that SHHOIRC exhibits notable similarities to other previously published algorithms. The proposed algorithm outperformed the other algorithms by 81.25%, compared to 18.75% for the prior algorithms, by obtaining the best average solutions for 13 benchmark functions. Furthermore, the proposed algorithm is tested on a real-life problem, which is the maximum coverage problem of Wireless Sensor Networks (WSNs), and compared with pure HHO, and two well-known algorithms, Grey Wolf Optimization (GWO) and Whale Optimization Algorithm (WOA). For the maximum coverage experiments, the proposed algorithm demonstrated superior performance, surpassing other algorithms by obtaining the best coverage rates of 95.4375% and 97.125% for experiments 1 and 2, respectively.

Keywords: Harris Hawk Optimization; metaheuristic; chaos optimization; chaotic maps; self-adaptive; maximum coverage; wireless sensor network

MSC: 68T05; 68Q32



Citation: Almotairi, S.; Badr, E.; Abdul Salam, M.; Dawood, A. Three Chaotic Strategies for Enhancing the Self-Adaptive Harris Hawk Optimization Algorithm for Global Optimization. *Mathematics* **2023**, *11*, 4181. <https://doi.org/10.3390/math11194181>

Academic Editors: Adrian Deaconu, Petru Adrian Cotfas and Daniel Tudor Cotfas

Received: 23 August 2023

Revised: 16 September 2023

Accepted: 22 September 2023

Published: 6 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, numerous different real-world optimization challenges, such as engineering and scientific experiments, have seen the widespread adoption of nature-inspired algorithms. The optimization problem involves locating the optimal solution (maximum or minimum) among all feasible alternatives. Due to the increasing complexity of optimization problems, traditional mathematical methods such as gradient descent have become

insufficient because they are time-consuming and confined to local optima. To tackle such issues, metaheuristic algorithms have gained popularity due to their efficiency, simplicity, easy implementation process, and low computational costs. Metaheuristic algorithms exhibit competitive performance and outperform other solvers in solving problems since they mimic biological and physical behavior in nature. Some examples are Particle Swarm Optimization (PSO) [1], Genetic algorithm (GA) [2], Grey Wolf Optimizer (GWO) [3], Whale Optimization Algorithm (WOA) [4], Flower Pollination Algorithm (FPA) [5], Salp Swarm Algorithm (SSA) [6], Moth–Flame Optimization (MFO) [7], Artificial Bee Colony Optimization (ABC) [8], Bat Algorithm (BA) [9], Wild Horse Optimizer (WHO) [10], Lightning Search Algorithm (LSA) [11], and Ant Colony Optimization (ACO) [12].

The No Free Lunch theorem (NFL) [13] states that there is no optimal algorithm capable of solving all optimization problems. Consequently, a significant number of metaheuristic algorithms have been devised over the past two decades, and it has been demonstrated that metaheuristic techniques can be used to efficiently and effectively find solutions to complex problems. In other words, if an algorithm performs well in one optimization task, it may not perform well in other optimization tasks. Therefore, the fact that a metaheuristic algorithm works well for solving one type of problem does not necessarily guarantee that it will work just as well for tackling another type of problem. HHO is one of the most effective metaheuristic algorithms; it is inspired by the cooperative behavior and chasing manner of Harris Hawks, as well as their predation and breeding behaviors in nature [14]. HHO outperforms many well-known Swarm Intelligence (SI) approaches, such as PSO, GWO, CS, MFO, DE, BA, and WOA, based on the outcomes of tests conducted on benchmark functions and multiple engineering optimization problems. Moreover, the results demonstrate that HHO obtains a good balance between exploration and exploitation, thereby enhancing its scalability and capacity to produce high-quality solutions [15]. As with many other algorithms, HHO has disadvantages, including the possibility of becoming locked in local optima and the absence of a study frame with dramatic convergence. Owing to the stochastic nature of the optimization process, achieving a balance between exploration and exploitation is one of the most difficult aspects of devising a metaheuristic algorithm [4]. The NFL has led to the development of HHO through a variety of techniques, including binary HHO release, evolutionary update structures, chaotic processes, and hybrid HHO [16].

In the literature, various studies have proposed enhancements to the HHO algorithm by incorporating different mathematical operators. For instance, Binary HHO (BHHO) and Quadratic Binary HHO (QBHHO) were introduced as binary versions of HHO [17]. Opposite HHO (OHOO) [18] was applied for feature selection in breast cancer classification. Chaotic HHO (CHHO) and variant chaotic HHO were proposed by Menesy et al. [19] and Basha et al. [20], respectively, while Hussien and Amin [21] enhanced a new HHO version based on chaotic local search, opposition-based learning, and self-adaption. Multi-Objective Harris Hawk Optimization algorithm (MOHHO) [22] employed Pareto dominance and crowding distance mechanisms to handle multiple objectives and improve diversity in the search space. Shi et al. [23] proposed a framework SGLHHOSVM that combines HHO with Support Vector Machine (SVM). Adaptive relative reflection HHO (ARHHO) was proposed by Zuo and Wang [24]. A hybrid Harris Hawk Optimization algorithm with Differential Evolution (HHO-DE) is proposed in [25]. Modified Harris Hawk Optimization algorithm (MHOO) is proposed in [26]. In [27], various Harris Hawk Optimization (HHO) algorithm techniques are proposed. These techniques, which include Improved HHO Opposition-Based Learning (OBL) (IHHOOBL), Improved HHO Lévy Flight (IHHOLF), and Improved HHO Chaotic Map (IHHOCM) to improve search efficiency and exploration and exploitation capabilities. In [28], the authors introduced an improved version of the Harris Hawk Optimization (IHHO) algorithm by incorporating elite opposite-based learning and a new search mechanism.

The theory of nonlinear dynamics chaos has found widespread application, and several metaheuristic algorithms have used chaos to enhance performance [29]. Chaos theory involves the study of chaotic dynamical systems, which are highly sensitive to initial conditions and comprise an infinite number of unstable periodic motions. In particular, the use of chaotic mapping techniques was developed to determine random parameters in metaheuristic algorithms, leading to faster convergence and reduced chance of being trapped in local optima [30,31]. In this context, a new variant of HHO is presented in this paper to further enhance its performance. This variant incorporates chaos theory and a self-adaptive technique to improve the algorithm's exploration and exploitation capabilities.

In recent years, wireless sensor networks (WSNs) have spread rapidly. This is a consequence of its ubiquitous daily usage in several fields. Applications for WSNs include monitoring the environment, surveillance systems, healthcare, the Internet of Things (IoT), and public safety [32,33]. Power scarcity is an important issue that negatively affects the WSNs' performance. The coverage problem of WSNs is a fundamental problem that directly affects the energy consumption of WSNs [34]. WSN's coverage problem of a specific region refers to finding the minimum number of sensors to cover the maximum area [32]. Designing a WSN with a high coverage rate and a reasonable number of sensors leads to efficient energy consumption and a longer lifetime. The maximum coverage problem for wireless sensor networks is an NP-hard optimization problem.

There has been a lot of interest in studying the coverage problem of WSNs, and several research results have been issued in recent years. In the literature, Hu et al. [35] proposed a method to optimize large radius sensors' deployment using binary PSO (binary-PSO). Aziz et al. [36] used a PSO-Voronoi algorithm to minimize the coverage hole. Ngatchou et al. [37] used a modified version of PSO, sequential PSO. In [38], Wang, X et al. proposed virtual force-directed co-evolutionary PSO to deploy WSN with good efficiency. Wang et al. [39] proposed a novel head wolf mutation strategy to enhance Wolf Pack Algorithm (WPA) to optimize coverage. Herein [40], Alshaqqawi et al. proposed an optimization approach that combined the PSO algorithm with a greedy method. Liang et al. [41] presented the MTCLM problem to maximize the number of covered targets. In [42], Wang, G proposes a positioning method based on a random PSO algorithm (SPSO). By integrating the locally convergent PSO with the gravity search algorithm, Mehta and Malik [43] introduced a new hybrid optimization algorithm that modifies the PSO into a global optimization algorithm. Tarnaris et al. [44] used GA, PSO, grid-based PSO, and a Voronoi-based PSO method to maximize area coverage and area k-coverage. Bonnah et al. [45] proposed a combination of the computed minimal exposed path and PSO algorithm using the ratio of covered to uncovered grids as a fitness function. Li and Liu [46] proposed an optimization algorithm for monitoring area coverage based on controlling the node position, which can rapidly improve the coverage effect. Zhang [47] simulated the coverage optimization of WSNs based on the Improved PSO algorithm. Gökalp [48] used the self-adaptive DE algorithm (SADE) to optimize the connected target coverage for the first time in the literature. In [49], Zhang, Y. and Zhang, Z. used DBSCAN and TDADA-II. Mobile Assisted Coverage Hole Patching Scheme (MACHPS) is introduced by Wang et al. [50] who compared it with two other algorithms, CMPA and CHHA. Mnasri et al. [51] proposed a hybrid approach relying on a novel bird's accent-based many-objective PSO algorithm (named acMaPSO) and another variant (named acMaMaPSO), and compared their performance with the standard PSO and the NSGA-III algorithm. Weighted Salp Swarm Algorithm (WSSA) is a technique proposed by M. A. Syed and R. Syed in [52] to improve the performance and convergence rate of the SSA and to compare WSSA's performance with PSO, MFO, GWO, WOA, MVO, and MGWO.

In this paper, we propose a new version of HHO. The proposed algorithm (SHHOIRC) is a hybrid algorithm consisting of Harris Hawk optimization, three chaos optimization methods (Chaotic mapped initialization, Chaotic mapped randomness, and Chaotic mapped controlled parameter), and a self-adaptive method. SHHOIRC is tested and compared with genuine HHO, SHHO [21], and four versions of HHO, which are pro-

posed in [31]. In [31], Harun and Haydar proposed forty HHO variants in which HHO is hybridized with ten distinct chaotic maps to modify its critical parameters, using four distinct hybridization techniques. The four methods used combinations of two chaos optimization techniques, which are Chaotic mapped randomness, and Chaotic mapped controlled parameter. Here, SHHOIRC is compared with four variants that use quietly comparable processes. The proposed algorithm and the other six HHO versions are applied to 16 Benchmark functions. Furthermore, the proposed algorithm is tested on a real-life problem, which is the maximum coverage problem, and compared with pure HHO, and two well-known algorithms, GWO [3] and WOA [4].

This document is structured as follows. Section 2 describes Harris Hawk optimization. The SHHOIRC algorithm and the used strategies are described in Section 3. Section 4 contains the results of the experiment, their simulation, and statistical analysis. Section 5 contains the conclusion.

2. Harris Hawk Optimization

HHO is a population-based metaheuristic technique inspired by the cooperative behavior and hunting style (surprise pounce) of Harris Hawks, as well as their natural predation and reproductive behaviors [14]. Harris Hawks forage with other family members, which distinguishes their foraging behavior from that of other birds. This algorithm has three phases: the Exploration phase, the Transportation phase, and the Exploitation phase. In the Exploration phase, the hawks perch randomly according to other members or at a random tall tree until they detect a prey. In the next phase, the Transportation phase, the hawks determine whether to chase the prey or not according to the prey's energy. In the Exploitation phase, depending on the dynamic character of situations and the evasion strategies of their prey, Harris Hawks pursue their prey in a variety of ways. Algorithm 1 explains pseudocode for HHO. The HHO phases can be modeled as follows:

2.1. Exploration Phase

In HHO, Harris Hawks represent candidate solutions, and the best solution in each phase is regarded as the intended prey or as close to the optimal solution. Based on two strategies, Harris Hawks settle at random in various locations and wait to detect prey. Considering an equal probability q for each perching technique: if $q < 0.5$, then the position of hawks is determined in regards to other hawks in the population and rabbit positions; if $q \geq 0.5$, then the hawk perch on a random tall tree inside the range. The position of each hawk can be modeled as in Equation (1), considering the stated perching strategies:

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 |X_{rand}(t) - 2r_2 X(t)| & \text{if } q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)) & \text{if } q < 0.5 \end{cases} \quad (1)$$

Such that $X(t+1)$ is the position vector of hawks in the next iteration, $X_{rabbit}(t)$ is the rabbit position in the current iteration t , $X(t)$ is the current position vector of hawks, $X_{rand}(t)$ is a randomly selected hawk from the current population, r_1 , r_2 , r_3 , and r_4 are random numbers in the interval of $(0, 1)$, LB and UB are the lower and upper bounds of the problem, and $X_m(t)$ is the average position of the current population of N hawks (Equation (2)):

$$X_m(t) = \frac{1}{N} \sum_{i=1}^N X_i(t) \quad (2)$$

2.2. Transportation from Exploration to Exploitation

The transition from exploration to exploitation is dependent on the prey's escaping energy. During its fleeing behavior, the prey's energy decreases drastically. Exploration

occurs when $|E| \geq 1$, whereas exploitation occurs in subsequent phases when $|E| < 1$. The prey's energy can be represented by Equation (3):

$$E = 2E_0 \left(1 - \frac{t}{T}\right) \quad (3)$$

where E indicates the escaping energy of the prey, T represents the maximum number of iterations, and E_0 is the initial state of the rabbit's energy, which randomly changes inside the interval $(-1, 1)$ iteratively.

2.3. Exploitation Phase

In this phase, the Harris Hawks perform a surprise pounce (seven kills) by attacking the intended prey detected in the previous phase. In the attacking stage, there are four possible strategies proposed in regard to the escaping behavior and the chasing strategies. It is supposed that r is the chance of a prey successfully escaping ($r < 0.5$) or not successfully escaping ($r \geq 0.5$) before the surprise pounce. The four strategies are as follows:

2.3.1. Soft Besiege

If $r \geq 0.5$ and $|E| \geq 0.5$, then the prey still has energy but cannot escape successfully. In this case, the hawks proceed with a soft besiege. The position update formula can be modeled as Equations (4) and (5):

$$X(t+1) = \Delta X(t) - E|JX_{rabbit}(t) - X(t)| \quad (4)$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \quad (5)$$

Such that $\Delta X(t)$ represents the difference between the rabbit's position vector and its current location in iteration t . J is the random jump strength of the rabbit throughout the escaping procedure. The J value changes randomly in each iteration to simulate the rabbit motion. The jump strength can be modeled as in Equation (6), where r_5 is a random number inside $(0, 1)$.

$$J = 2(1 - r_5) \quad (6)$$

2.3.2. Hard Besiege

The prey lacks sufficient energy and cannot effectively flee when $r \geq 0.5$ and $|E| < 0.5$. In this case, the hawks proceed to hard besiege. The position update formula can be modeled as Equation (7):

$$X(t+1) = X_{rabbit}(t) - E|\Delta X(t)| \quad (7)$$

2.3.3. Soft Besiege with Progressive Rapid Dives

If $r < 0.5$ and $|E| \geq 0.5$, then the prey still has enough energy and can escape successfully. This case is inspired by the actual behavior of hawks; it is assumed that, in competitive situations, they can choose the optimal dive toward prey in order to capture it. Consequently, the raptors can evaluate their next move based on the following rule, which is represented by Equation (8):

$$Y = X_{rabbit}(t) - E|JX_{rabbit}(t) - X(t)| \quad (8)$$

If the hawk will dive based on the LF-based patterns, the next move is to proceed using Equation (9):

$$Z = Y + S \times LF(D) \quad (9)$$

Such that D is the dimension of the problem, S is a random vector by size $1 \times D$, and LF is the Levy Flight function (Equation (10)) where u, v are random values inside $(0,1)$ and β is a default constant set to 1.5.

$$LF(D) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \quad \sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \quad (10)$$

In this phase, the next location is set as the better between Y and Z . Hence, the position formula of this phase can be modeled as Equation (11):

$$X(t+1) = \begin{cases} Y & \text{if } F(Y) < F(X(t)) \\ Z & \text{if } F(Z) < F(X(t)) \end{cases} \quad (11)$$

2.3.4. Hard Besiege with Progressive Rapid Dives

If $r < 0.5$ and $|E| < 0.5$, then the prey still does not have enough energy and cannot escape successfully. In this case, the hawks proceed to hard besiege with progressive rapid dives. The position update formula can be modeled as Equation (12):

$$X(t+1) = \begin{cases} Y & \text{if } F(Y) < F(X(t)) \\ Z & \text{if } F(Z) < F(X(t)) \end{cases} \quad (12)$$

where Y and Z are obtained using new rules (Equations (13) and (14)):

$$Y = X_{rabbit}(t) - E|JX_{rabbit}(t) - X_m(t)| \quad (13)$$

$$Z = Y + S \times LF(D) \quad (14)$$

Algorithm 1 Pseudo-code of HHO Algorithm

```

1: Initialize the parameters (Popsiz (N), MaxIter (T), LB, UB, and Dim)
2: Initialize a population  $X_0$ 
3: While ( $iter \leq MaxIter$ ) do
4:   Compare the fitness Function for hawk  $x_i$ 
5:    $X_{rabbit}$  = the best search agent
6:   for each hawk ( $x_i$ ) do
7:     Update the escaping energy  $E$  using Equation (3)
8:     if ( $|E| \geq 1$ ) then
9:       Update hawk position using Equation (1)
10:    end if
11:    if ( $|E| < 1$ ) then
12:      if ( $r \geq 0.5$  and  $|E| \geq 0.5$ ) then
13:        Update hawk position using Equation (4)
14:      else if ( $r \geq 0.5$  and  $|E| < 0.5$ ) then
15:        Update hawk position using Equation (7)
16:      else if ( $r < 0.5$  and  $|E| \geq 0.5$ ) then
17:        Update hawk position using Equation (11)
18:      else if ( $r < 0.5$  and  $|E| < 0.5$ ) then
19:        Update hawk position using Equation (12)
20:      end if
21:    end if
22:  end for
23: End
24: Return  $X_{rabbit}$ 

```

3. SHHOIRC Algorithm

The proposed algorithm is Self-adaptive Harris Hawk Optimization with three Chaotic Optimization methods (Chaotic mapped initialization, Chaotic mapped randomness, and Chaotic mapped controlled parameter) called (SHHOIRC).

3.1. Chaos Optimization

Chaos is a random-like common phenomenon that occurs in nonlinear systems. It is considered as a source of stochasticity [53]. The chaotic map can generate a huge number of non-repetitive sequences instead of random sequences by only changing its initial values. The following chaotic maps are examples of chaotic maps that can be used to generate non-repetitive random numbers within specified limits, as detailed in Table 1, where z_k and z_{k+1} are the chaotic variables of k th and $(k + 1)$ th iteration, respectively. For initial conditions of the two chaotic maps, z_k is set as a random number belonging to $[0, 1]$. The chaotic variable is generated using the previous chaotic variable in the sequence. In the proposed algorithm, the Circle map is used to generate all the chaotic variables. The theory of chaos was applied to HHO in order to enhance convergence speed and prevent it from falling into the local minimum trap by utilizing a chaotic map to explore unvisited regions in the search space.

Table 1. Chaotic maps' details.

No.	Map Name	Map Equations	Parameters
1	Circle map	$z_{k+1} = z_k + \varphi - \frac{K}{2\pi} \sin(2\pi z_k) \bmod(1)$	$\varphi = 2.5, K = 5$
2	Logistic map	$z_{k+1} = \mu (1 - z_k)$	$\mu = 4$
3	Gaussian map	$z_{k+1} = 0, \quad \text{if } z_k = 0;$ $z_{k+1} = \frac{1}{z_k} \bmod(1), \quad \text{if } z_k \neq 0$	
4	Chebyshev map	$z_{k+1} = \cos(a \cos^{-1}(z_k))$	$a = 5$
5	Sinusoidal map	$z_{k+1} = \sin(\pi z_k)$	

The chaotic maps have been introduced to improve nature-inspired algorithms for many years. As stated in [54], there are four ways for chaotic applications to improve the performance of the existing nature-inspired algorithms. The chaotic improvements can be categorized as follows: Chaotic-mapped initialization, Chaotic-mapped randomness, Chaotic-mapped iterations, and Chaotic-mapped controlled parameters. In the literature, chaotic-mapped initialization [55] enhances PSO, chaotic-mapped randomness [56] improves the Bat algorithm, chaotic-mapped iterations [53] enhance the Grey Wolf optimizer, and chaotic-mapped controlled parameters [57] optimize GWO algorithm performance. These approaches significantly contribute to their respective algorithms. In the proposed algorithm, three types of strategies are used.

The first way is chaotic-mapped initialization; in this method, the chaotic variables are used to generate the initial population of HHO. This modification can be applied by generating random initial positions for the hawks using the chaotic variables. The generated positions can then be limited within the upper and lower boundaries of the search space according to the optimization problem. The initial population positions $X_{initial}$ can be modeled as follows (Equation (15)):

$$X_{initial} = LB + X_0 z_{k+1} \times (UB - LB) \quad (15)$$

where X_0 is the random positions generated, and z_{k+1} is the chaotic variable generated using the circle chaotic map. UB and LB are the upper and lower limits of the problem.

The second method is chaotic-mapped randomness; in this method, the chaotic variables are used to generate the random numbers used in Equation (1) and to choose $X_{rand}(t)$. This modification can be modeled as follows (Equation (16)):

$$X(t+1) = \begin{cases} X_{rand}(t) - z_{k+1} r_1 |X_{rand}(t) - 2z_{k+1} r_2 X(t)| & \text{if } q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - z_{k+1} r_3 (LB + z_{k+1} r_4 (UB - LB)) & \text{if } q < 0.5 \end{cases} \quad (16)$$

The third one is the chaotic-mapped controlled parameter; in this method, the chaotic variables are used to generate a random number used to edit the control parameter E in Equation (3). This modification can be modeled as follows (Equation (17)):

$$E = 2z_{k+1} E_0 \left(1 - \frac{t}{T}\right) \quad (17)$$

3.2. Self-Adaptive Technique

To improve the diversity and exploration capabilities of the HHO, a self-adaptive technique is used. The self-adaptive technique is integrated with HHO and a new solution is generated using Equation (18):

$$v_i(t+1) = X_i(t+1) + S_R (X_{CurrBest} - X_i(t)) \quad (18)$$

Equation (18) updates the state of solution $X_i(t+1)$. $X_i(t+1)$ are the solutions obtained in iteration t , which are prepared for use in the upcoming iteration $t+1$, using a proposed search equation based on self-adaptation. $X_i(t)$ is the i th hawk solution before starting iteration t , and $X_{CurrBest}$ is the best solution found so far for $X_i(t)$. $X_{CurrBest}$ can be described as the local best for the i th hawk after iteration t is performed, and S_R is a self-adaptive rate that is a random number between 0 and 1.

3.3. Proposed Algorithm

The proposed algorithm integrates the introduced three chaotic strategies and a self-adaptive technique with HHO. The first chaotic strategy enhances HHO by allowing individuals to spread throughout the domain of the given problem as equally as possible. In this step, either the global or local optimum has a greater chance of gaining access by some individuals. The second type enhances HHO by replacing pseudo-random numbers with a non-repetitive sequence of random numbers generated by the circle chaotic map, which enhances the exploration phase diversity and exploits all previously visited promising areas. The third one is used to balance the ratio of exploration and exploitation for individuals during iterations. The self-adaptive method helps control the balance between exploration and exploitation, and the proposed search equation helps the algorithm explore previously unvisited promising areas. The pseudocode of SHHOIRC is shown in Algorithm 2.

In SHHOIRC, the integration of the three strategies and the self-adaptive method with HHO is performed by using the four Equations (15)–(18), as in the flowchart in Figure 1. The initial population is generated using Equation (15) by considering the problem limitations. At each iteration, the positions of the hawks are updated using Equation (16). To update the control parameter E for each hawk, use Equation (17). After generating the hawks' solutions and obtaining HHO's best solution for the current iteration, a new solution is generated using Equation (18), and then the best solution among the HHO solution and self-adaptive solution is selected as the new solution for the current iteration.

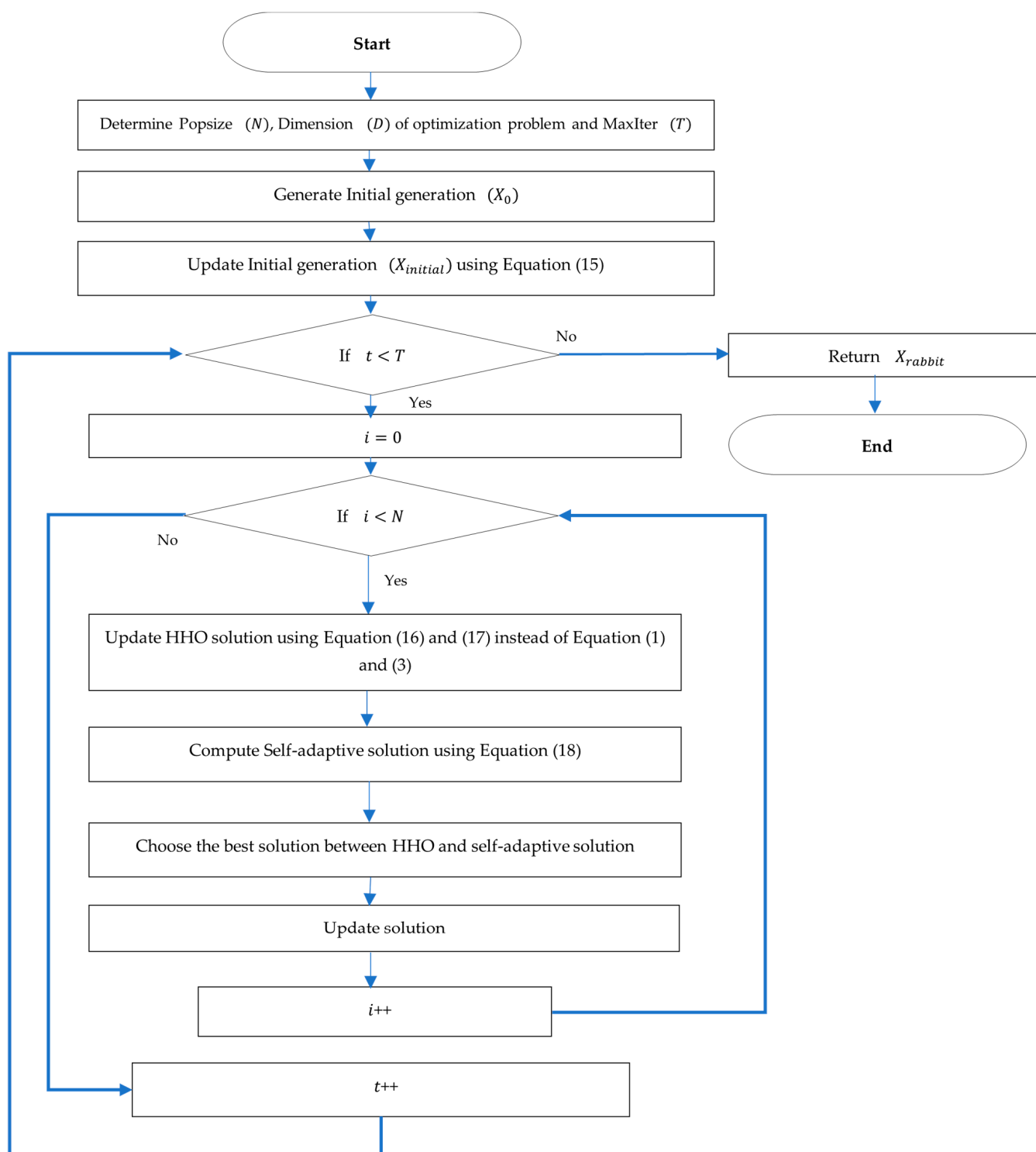


Figure 1. The flowchart of SHHOIRC.

Algorithm 2 Pseudo-code of SHHOIRC Algorithm

```

1: Initialize the parameters (Popsiz (N), MaxIter (T), LB, UB, and Dim)
2: Initialize a population  $X_0$ 
3: Update  $X_{initial}$  using Equation (15)
4: While ( $iter \leq MaxIter$ ) do
5:     Compare the fitness Function for hawk  $x_i$ 
6:      $X_{rabbit}$  = the best search agent
7:     for each hawk ( $x_i$ ) do
8:         Update the escaping energy  $E$  using Equation (17)
9:         if ( $|E| \geq 1$ ) then
10:            Update hawk position using Equation (16)
11:        end if
12:        if ( $|E| < 1$ ) then
13:            if ( $r \geq 0.5$  and  $|E| \geq 0.5$ ) then
14:                Update hawk position using Equation (4)
15:            else if ( $r \geq 0.5$  and  $|E| < 0.5$ ) then
16:                Update hawk position using Equation (7)
17:            else if ( $r < 0.5$  and  $|E| \geq 0.5$ ) then
18:                Update hawk position using Equation (11)
19:            else if ( $r < 0.5$  and  $|E| < 0.5$ ) then
20:                Update hawk position using Equation (12)
21:            end if
22:        end if
23:        Update hawk using Equation (18)
24:    end for
25: End
26: Return  $X_{rabbit}$ 

```

4. Simulation Experiments

In this article, the proposed SHHOIRC, the four CHHO that are proposed by Harun et al. [31], Self-adaptive HHO (SHHO), and genuine HHO are applied to 16 benchmark functions. Furthermore, SHHOIRC is tested on a real-life problem, which is the maximum coverage problem of WSN, and compared with pure HHO, and two well-known algorithms GWO and WOA. For the maximum coverage problem, two experiments are performed. All the experiments are applied on a PC with the following specifications: Intel Core i5-10400 CPU 2.90 GHz, 16.0 GB RAM. Table 2 shows the experimental parameter setting.

Table 2. Experimental parameter setting.

No.	Parameter Name	Benchmark Functions Experiments	Maximum Coverage Experiments
1	Population size	30	30
2	Number of Dimensions	According to function (Table 3)	30
3	Max iteration and run	500 and 5 runs	(Exp1: 500, Exp2: 1000) and 10 runs for each

Table 3. List of Benchmark Functions.

No.	Name	Function Expressions	Dimension (d)	Initial Range [LB, UB]	Type	Optimal Solution f_{min}
F1	Schwefel 2.22	$F(x) = \sum_{i=1}^d x_i + \prod_{i=1}^d x_i $	30	$[-10, 10]$	UM	0
F2	Schwefel 2.21	$F(x) = \max_{1 \leq i \leq d} x_i $	30	$[-100, 100]$	UM	0
F3	Generalized Rosenbrock	$F(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	UM	0

Table 3. Cont.

No.	Name	Function Expressions	Dimension (d)	Initial Range [LB, UB]	Type	Optimal Solution f_{min}
F4	Quartic with Noise	$F(x) = \sum_{i=1}^d ix_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	MM	0
F5	Generalized Rastrigin	$F(x) = \sum_{i=1}^d [x_i^2 - 10 \cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	MM	0
F6	Ackley 1	$F(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{d} \sum_{i=1}^d x_i^2}\right) - \exp\left(\frac{1}{d} \sum_{i=1}^d \cos(2\pi x_i)\right) + 20 + e$	30	$[-32, 32]$	MM	0
F7	Generalized Griewank Function	$F(x) = \sum_{i=1}^d \frac{x_i^2}{4000} - \prod_{i=1}^d \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$	MM	0
F8	Generalized Penalized 1	$F(x) = \frac{\pi}{d} \left\{ 10 \sin^2(\pi y_i) + \sum_{i=1}^d \frac{(y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})]}{(y_d - 1)^2} \right\} + \sum_{i=1}^d u(x_i, 10, 100, 4)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & , x_i > a \\ 0 & , -a < x_i < a \\ k(-x_i - a)^m & , x_i < -a \end{cases}$	30	$[-50, 50]$	MM	0
F9	Generalized Penalized 2	$F(x) = 0.1 \left\{ \sin^2(3\pi x_i) + \sum_{i=1}^d \frac{(x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)]}{(x_i - 1)^2 [1 + \sin^2(2\pi x_i)]} \right\} + \sum_{i=1}^d u(x_i, 5, 100, 4)$	30	$[-50, 50]$	MM	0
F10	Six-Hump Camel-Back Function	$F(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (4x_2^2 - 4)x_2^2$	2	$[-5, 5]$	MM	-1.0316285
F11	Goldstein–Price Function	$F(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	$[-5, 5]$	MM	3
F12	Hartman 3	$F(x) = -\sum_{i=1}^4 c_i \exp\left[-\sum_{j=1}^3 a_{ij} (x_j - p_{ij})^2\right]$	3	$[0, 1]$	MM	-3.86278
F13	Hartman 6	$F(x) = -\sum_{i=1}^4 c_i \exp\left[-\sum_{j=1}^6 a_{ij} (x_j - p_{ij})^2\right]$	6	$[0, 1]$	MM	-3.32237
F14	Shekel 5	$F(x) = -\sum_{i=1}^5 \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	MM	-10.1532
F15	Shekel 7	$F(x) = -\sum_{i=1}^7 \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	MM	-10.4029
F16	Shekel 10	$F(x) = -\sum_{i=1}^{10} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	MM	-10.5364

4.1. Measuring Performance

- Average is the mean of the values obtained in different runs:

$$AVG = \frac{1}{N} \sum_{i=1}^N F_i \quad (19)$$

where N and F_i are the number of runs and the value obtained in i th run, respectively.

- Standard deviation is calculated as follows:

$$STD = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (F_i - Mean)^2} \quad (20)$$

- Best is the best (minimum or maximum according to the optimization problem) solution obtained in different runs.
- Worst is the worst (maximum or minimum according to the optimization problem) solution obtained in different runs.

4.2. Testing HHO Versions on 16 Benchmark Functions

To evaluate the efficiency of the proposed metaheuristic SHHOIRC method, 16 well-known benchmark functions are subjected to a series of experiments. These functions are separated into three Unimodal functions and 13 Multimodal functions to evaluate the algorithm's capacity to avoid local optima and locate the global optimum. The expressions of each test function are presented in Table 3, and specific parameters for some functions are shown in Tables 4–6.

Each algorithm is run independently five times with 500 iterations per run and a population size of 30 agents. Table 7 compares the proposed HHO version with six other HHO versions (genuine HHO, SHHO, and four CHHO versions proposed in [31]) in terms of the average solution, standard deviation values, best solutions obtained, worst solutions obtained, and average time consumed (seconds). The four CHHO versions are categorized into two types, PM1 and PM2, each using two different chaotic maps (Circle and Sinusoidal, which are presented in Table 1). This study aims to demonstrate the superiority of the proposed SHHOIRC method over other HHO variants and compare its performance on various benchmark functions.

Table 4. Parameters of F13 (Hartman 6-Dimensional function).

i	$a_{ij}, j = 1, \dots, 6$						c_i	$p_{ij}, j = 1, \dots, 6$					
1	10	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10	17	0.1	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	10	17	8	3	0.2348	0.1415	0.3522	0.2883	0.3047	0.6650

Table 5. Parameters of Shekel functions: F14 (Shekel-5), F15 (Shekel-7), and 16 (Shekel-10).

i	$a_{ij}, j = 1, \dots, 4$				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

Table 6. Parameters of F12 (Hartman 3-Dimensional function).

i	$a_{ij}, j = 1, \dots, 3$			c_i	$p_{ij}, j = 1, \dots, 3$		
1	3.0	10	30	1.0	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3.0	10	30	3.0	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.0381	0.5743	0.8828

Table 7. HHO versions' results (500 iterations, 30 agents).

	Metric	HHO	SHHOIRC	CHHO PM1C2	CHHO PM1C9	CHHO PM2C2	CHHO PM2C9	SHHO
F1	AVG	9.86×10^{-52}	1.04×10^{-59}	1.18×10^{-49}	1.19×10^{-44}	2.69×10^{-52}	9.04×10^{-45}	1.03×10^{-58}
	STD	1.53×10^{-51}	2.18×10^{-59}	2.64×10^{-49}	2.66×10^{-44}	2.77×10^{-52}	2.00×10^{-44}	1.31×10^{-58}
	Best	8.06×10^{-55}	4.95×10^{-59}	5.90×10^{-49}	5.94×10^{-44}	6.64×10^{-52}	4.47×10^{-44}	2.82×10^{-64}
	Worst	3.66×10^{-51}	7.47×10^{-63}	4.58×10^{-54}	3.76×10^{-52}	1.33×10^{-54}	8.85×10^{-50}	3.06×10^{-58}
	AVG Time	1.19×10^{-1}	2.30×10^0	5.54×10^{-1}	5.21×10^{-1}	1.35×10^0	1.43×10^0	7.08×10^{-1}
F2	AVG	5.57×10^{-48}	1.73×10^{-49}	1.75×10^{-50}	1.01×10^{-41}	8.54×10^{-48}	9.84×10^{-48}	3.62×10^{-48}
	STD	1.24×10^{-47}	3.84×10^{-49}	3.59×10^{-50}	1.56×10^{-41}	1.91×10^{-47}	1.94×10^{-47}	8.08×10^{-48}
	Best	1.12×10^{-57}	8.59×10^{-49}	8.16×10^{-50}	3.53×10^{-41}	4.27×10^{-47}	4.43×10^{-47}	7.04×10^{-55}
	Worst	2.78×10^{-47}	3.13×10^{-54}	6.30×10^{-57}	6.18×10^{-48}	8.31×10^{-55}	6.47×10^{-54}	1.81×10^{-47}
	AVG Time	1.36×10^{-1}	1.98×10^0	5.51×10^{-1}	5.38×10^{-1}	1.39×10^0	1.43×10^0	5.67×10^{-1}
F3	AVG	2.43×10^{-2}	8.35×10^{-04}	8.59×10^{-03}	6.02×10^{-03}	3.97×10^{-03}	1.33×10^{-03}	8.59×10^{-04}
	STD	3.19×10^{-2}	5.21×10^{-4}	8.12×10^{-3}	4.37×10^{-3}	5.16×10^{-3}	2.80×10^{-3}	5.94×10^{-4}
	Best	3.21×10^{-4}	1.58×10^{-3}	2.02×10^{-2}	1.13×10^{-2}	1.29×10^{-2}	6.34×10^{-3}	2.99×10^{-4}
	Worst	7.78×10^{-2}	4.09×10^{-4}	6.35×10^{-4}	1.74×10^{-3}	9.27×10^{-5}	3.42×10^{-5}	1.82×10^{-3}
	AVG Time	2.12×10^{-1}	1.72×10^0	6.39×10^{-1}	6.04×10^{-1}	1.53×10^0	1.56×10^0	6.59×10^{-1}
F4	AVG	5.34×10^{-5}	2.09×10^{-4}	1.16×10^{-4}	9.04×10^{-5}	1.42×10^{-4}	1.77×10^{-4}	9.11×10^{-5}
	STD	4.20×10^{-5}	2.47×10^{-5}	7.69×10^{-5}	1.56×10^{-4}	5.80×10^{-5}	2.58×10^{-4}	5.78×10^{-5}
	Best	5.68×10^{-6}	2.45×10^{-4}	1.79×10^{-4}	3.68×10^{-4}	1.97×10^{-4}	6.23×10^{-4}	5.00×10^{-5}
	Worst	1.21×10^{-4}	1.82×10^{-4}	8.17×10^{-6}	5.62×10^{-6}	7.19×10^{-5}	9.03×10^{-6}	1.93×10^{-4}
	AVG Time	3.35×10^{-1}	2.00×10^0	6.95×10^{-1}	6.74×10^{-1}	1.61×10^0	1.67×10^0	1.10×10^0
F5	AVG	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	STD	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Best	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Worst	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	AVG Time	1.77×10^{-1}	1.51×10^0	6.16×10^{-1}	6.30×10^{-1}	1.51×10^0	1.53×10^0	4.62×10^{-1}
F6	AVG	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	STD	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Best	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	Worst	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	AVG Time	1.89×10^{-1}	1.92×10^0	6.40×10^{-1}	6.53×10^{-1}	1.49×10^0	1.53×10^0	4.69×10^{-1}
F7	AVG	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	STD	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Best	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Worst	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	AVG Time	2.29×10^{-1}	1.80×10^0	7.01×10^{-1}	6.21×10^{-1}	9.42×10^{-1}	9.90×10^{-1}	5.06×10^{-1}

Table 7. Cont.

	Metric	HHO	SHHOIRC	CHHO PM1C2	CHHO PM1C9	CHHO PM2C2	CHHO PM2C9	SHHO
F8	AVG	1.18×10^{-5}	<u>2.33×10^{-7}</u>	4.76×10^{-6}	2.16×10^{-5}	6.84×10^{-6}	2.16×10^{-6}	3.17×10^{-7}
	STD	1.18×10^{-5}	<u>2.18×10^{-7}</u>	4.06×10^{-6}	2.89×10^{-5}	7.02×10^{-6}	2.55×10^{-6}	3.22×10^{-7}
	Best	8.71×10^{-7}	5.96×10^{-7}	8.60×10^{-6}	6.92×10^{-5}	1.87×10^{-5}	6.29×10^{-6}	3.53×10^{-8}
	Worst	3.04×10^{-5}	6.17×10^{-8}	3.30×10^{-7}	2.02×10^{-8}	4.20×10^{-7}	6.61×10^{-9}	8.48×10^{-7}
	AVG Time	7.67×10^{-1}	2.60×10^0	1.02×10^0	9.72×10^{-1}	1.31×10^0	1.26×10^0	1.51×10^0
F9	AVG	7.51×10^{-5}	<u>1.71×10^{-6}</u>	7.58×10^{-5}	5.53×10^{-5}	1.83×10^{-4}	6.89×10^{-6}	4.54×10^{-6}
	STD	7.61×10^{-5}	<u>4.64×10^{-7}</u>	1.36×10^{-4}	8.37×10^{-5}	2.65×10^{-4}	7.49×10^{-6}	4.46×10^{-6}
	Best	2.14×10^{-6}	2.15×10^{-6}	3.18×10^{-4}	1.97×10^{-4}	6.28×10^{-4}	1.84×10^{-5}	9.91×10^{-7}
	Worst	1.84×10^{-4}	1.10×10^{-6}	6.82×10^{-6}	5.34×10^{-7}	1.92×10^{-6}	3.68×10^{-9}	9.90×10^{-6}
	AVG Time	7.73×10^{-1}	2.95×10^0	1.05×10^0	1.03×10^0	1.32×10^0	1.30×10^0	1.52×10^0
F10	AVG	-1.03×10^0	<u>-1.03×10^0</u>	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0
	STD	1.03×10^{-9}	<u>1.57×10^{-16}</u>	5.84×10^{-10}	1.51×10^{-8}	7.08×10^{-11}	5.53×10^{-9}	1.92×10^{-16}
	Best	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0
	Worst	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0
	AVG Time	1.30×10^{-1}	1.95×10^0	7.23×10^{-1}	7.12×10^{-1}	9.67×10^{-1}	9.82×10^{-1}	6.79×10^{-1}
F11	AVG	3.00×10^0	<u>3.00×10^0</u>	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0
	STD	1.63×10^{-8}	6.08×10^{-15}	8.88×10^{-8}	7.16×10^{-7}	4.69×10^{-7}	8.83×10^{-7}	<u>4.32×10^{-15}</u>
	Best	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0
	Worst	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0
	AVG Time	1.10×10^{-1}	$1.89 \times 10^{+0}$	7.04×10^{-1}	6.94×10^{-1}	9.84×10^{-1}	1.02×10^0	6.96×10^{-1}
F12	AVG	-3.86×10^0	<u>-3.86×10^0</u>	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0
	STD	1.11×10^{-3}	<u>3.85×10^{-16}</u>	4.39×10^{-3}	4.73×10^{-4}	5.61×10^{-4}	3.31×10^{-3}	5.44×10^{-16}
	Best	-3.86×10^0	-3.86×10^0	-3.85×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0
	Worst	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0
	AVG Time	1.54×10^{-1}	$2.50 \times 10^{+0}$	7.39×10^{-1}	7.66×10^{-1}	1.02×10^0	1.06×10^0	7.14×10^{-1}
F13	AVG	-3.15×10^0	<u>-3.32×10^0</u>	-3.17×10^0	-3.11×10^0	-3.14×10^0	-3.16×10^0	-3.27×10^0
	STD	9.45×10^{-2}	<u>2.17×10^{-14}</u>	8.44×10^{-2}	1.54×10^{-1}	7.01×10^{-2}	3.50×10^{-2}	6.51×10^{-2}
	Best	-3.22×10^0	-3.32×10^0	-3.06×10^0	-2.89×10^0	-3.03×10^0	-3.12×10^0	-3.32×10^0
	Worst	-2.99×10^0	-3.32×10^0	-3.27×10^0	-3.30×10^0	-3.22×10^0	-3.22×10^0	-3.20×10^0
	AVG Time	1.57×10^{-1}	1.82×10^0	7.45×10^{-1}	7.31×10^{-1}	1.02×10^0	1.08×10^0	7.13×10^{-1}
F14	AVG	-5.05×10^0	<u>-6.07×10^0</u>	-5.05×10^0	-5.05×10^0	-5.05×10^0	-5.04×10^0	-5.06×10^0
	STD	3.06×10^{-3}	2.28×10^0	3.28×10^{-3}	4.27×10^{-4}	1.28×10^{-3}	1.50×10^{-2}	<u>7.69×10^{-16}</u>
	Best	-5.05×10^0	-5.06×10^0	-5.05×10^0	-5.05×10^0	-5.05×10^0	-5.02×10^0	-5.06×10^0
	Worst	-5.05×10^0	-1.02×10^1	-5.05×10^0	-5.06×10^0	-5.05×10^0	-5.05×10^0	-5.06×10^0
	AVG Time	1.91×10^{-1}	2.27×10^0	7.60×10^{-1}	7.49×10^{-1}	1.07×10^0	1.06×10^0	7.74×10^{-1}
F15	AVG	-5.09×10^0	<u>-6.15×10^0</u>	-5.08×10^0	-5.09×10^0	-5.09×10^0	-4.41×10^0	-6.15×10^0
	STD	2.40×10^{-3}	2.38×10^0	4.47×10^{-3}	<u>2.60×10^{-4}</u>	2.67×10^{-3}	1.46×10^0	2.38×10^0
	Best	-5.09×10^0	-5.09×10^0	-5.08×10^0	-5.09×10^0	-5.08×10^0	-1.79×10^0	-1.04×10^1
	Worst	-5.08×10^0	-1.04×10^1	-5.09×10^0	-5.09×10^0	-5.09×10^0	-5.08×10^0	-5.09×10^0
	AVG Time	2.17×10^{-1}	$2.13 \times 10^{+0}$	7.98×10^{-1}	7.61×10^{-1}	1.07×10^0	1.08×10^0	8.10×10^{-1}

Table 7. Cont.

	Metric	HHO	SHHOIRC	CHHO PM1C2	CHHO PM1C9	CHHO PM2C2	CHHO PM2C9	SHHO
F16	AVG	-5.12×10^0	-8.37×10^0	-6.17×10^0	-7.29×10^0	-5.12×10^0	-4.57×10^0	-8.37×10^0
	STD	5.29×10^{-3}	2.96×10^0	2.33×10^0	2.96×10^0	4.07×10^{-3}	1.21×10^0	2.96×10^0
	Best	-5.13×10^0	-5.13×10^0	-5.12×10^0	-5.13×10^0	-5.12×10^0	-2.41×10^0	-1.05×10^1
	Worst	-5.12×10^0	-1.05×10^1	-1.03×10^1	-1.05×10^1	-5.13×10^0	-5.11×10^0	-5.13×10^0
	AVG Time	2.62×10^{-1}	2.08×10^0	8.02×10^{-1}	7.85×10^{-1}	1.10×10^0	1.13×10^0	8.30×10^{-1}

The results presented in Table 7 demonstrate that the proposed HHO version (SHHOIRC) outperformed the other six algorithms in terms of average solutions obtained for 13 out of the 16 benchmark functions. HHO outperformed the other algorithms for four functions, while CHHO PM1C2 and SHHO performed the best for three functions, respectively. All seven HHO versions obtained the same best average solution for F5 and F7. In terms of the best average solutions with the lowest standard deviation, SHHOIRC outperformed the other algorithms for nine benchmark functions, while CHHO PM1C2, SHHO, and pure HHO performed the best for three functions. All seven HHO versions obtained the same best average solution with the same lowest standard deviation values for F5 and F7.

HHO consumed the least time for 16 benchmark functions. Regarding the best solutions obtained, SHHOIRC outperformed the other algorithms for 6 benchmark functions, while genuine HHO performed the best for 5, and SHHO for 10 functions. All seven HHO versions obtained the same best solution for F5 and F7. Finally, in terms of the lowest worst solutions obtained, SHHOIRC outperformed the other algorithms for 10 benchmark functions, CHHO PM2C9 for 5, and genuine HHO, CHHO PM1C9, and SHHO for 3 functions. All seven HHO versions obtained the same lowest worst solution for F5 and F7. According to the results presented in Table 7, the proposed hybrid algorithm, SHHOIRC, demonstrated superior performance compared with the other seven algorithms tested. The study concludes that SHHOIRC is a competitive algorithm. Additionally, when compared to the genuine HHO algorithm, SHHOIRC outperformed it by obtaining better average solutions on 14 benchmark functions. When compared to the two CHHO PM1 versions, SHHOIRC outperformed all of them for 13 benchmark functions in terms of average solutions achieved. Similarly, when compared with the five CHHO PM2 versions, SHHOIRC outperformed all of them for 15 benchmark functions. By comparing the proposed algorithm with the SHHO algorithm, SHHOIRC outperformed it by obtaining better average solutions on 15 benchmark functions.

Figure 2 displays the convergence curve of the average solutions obtained by the 7 HHO versions for 16 benchmark functions (F1–F16) in 500 iterations. Upon comparing the 7 HHO versions, the proposed algorithm outperformed the other HHO versions in terms of obtaining the best average solution for 13 benchmark functions out of the 16 tested. Therefore, the study concludes that the proposed SHHOIRC algorithm outperforms both the genuine HHO and other HHO versions.

According to the number of the best average solutions obtained in the experiment on benchmark functions, it may be concluded that SHHOIRC outperformed other algorithms by 81.25 percent, compared to 18.75 percent for the prior algorithms.

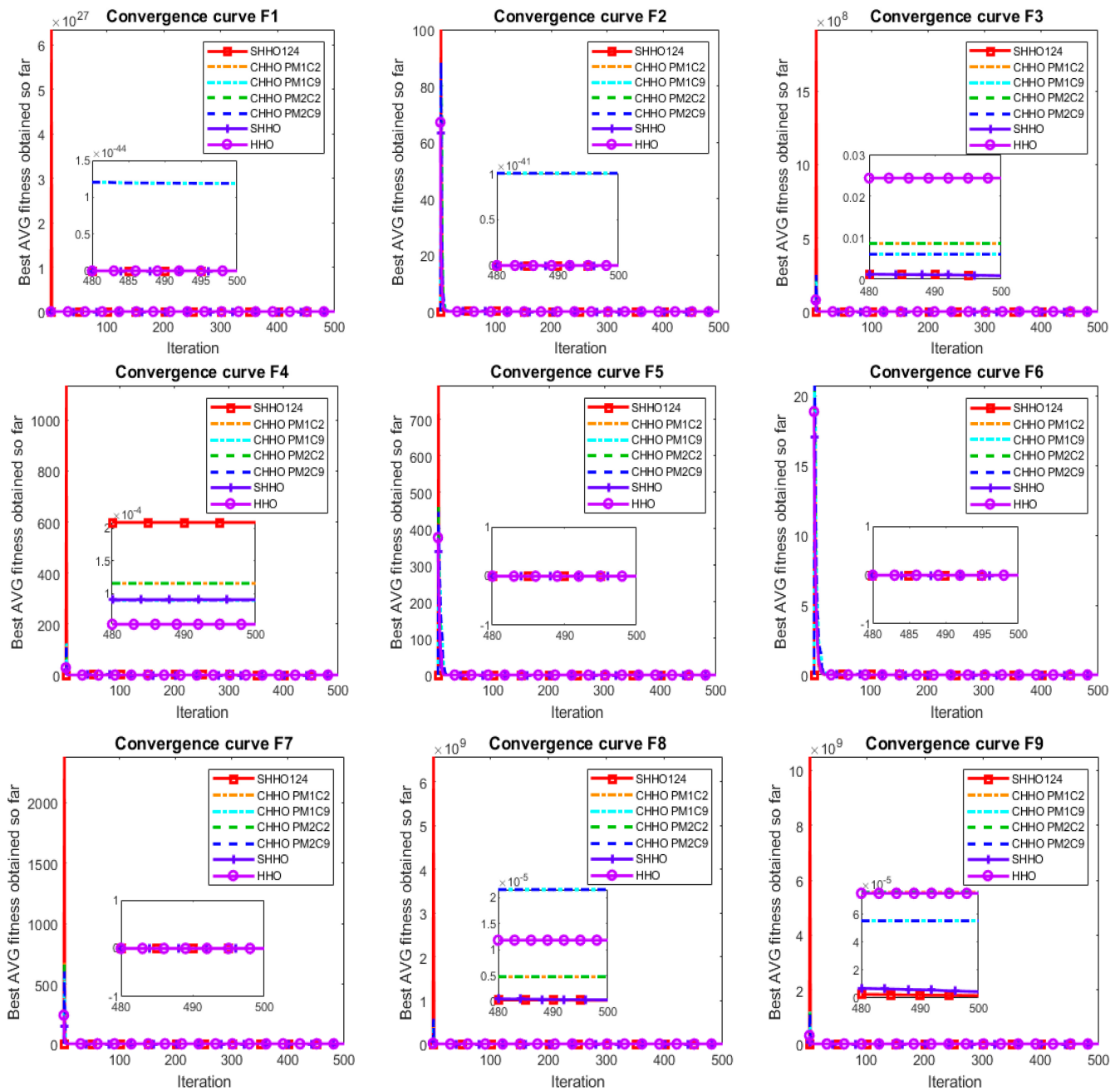


Figure 2. Cont.

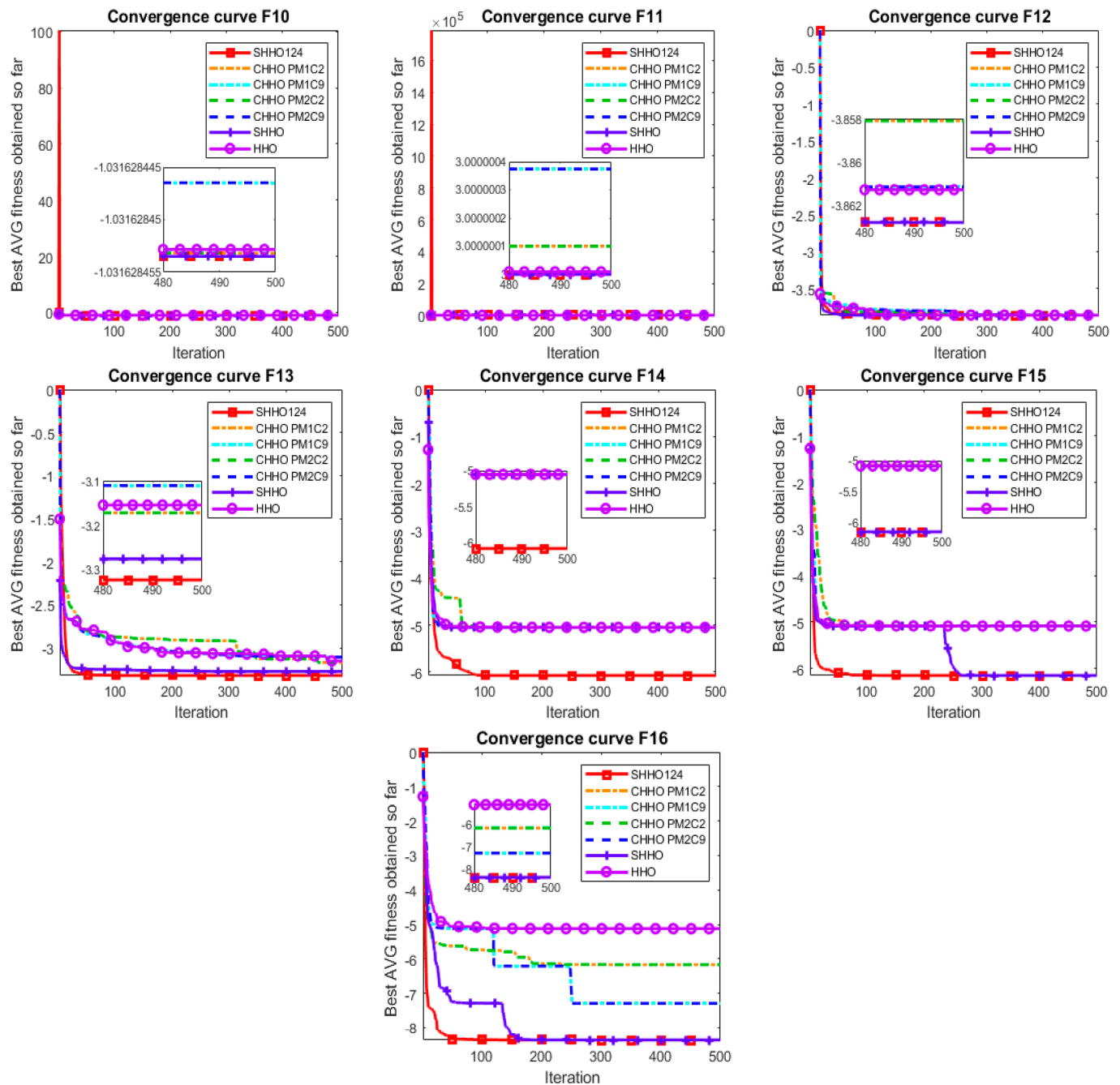


Figure 2. Average convergence curve obtained by 7 HHO versions for 16 benchmark functions (500 iterations).

Statistical Analysis

Statistical tests should be conducted to evaluate the performance of metaheuristic algorithms [58]. Statistical analysis is required to determine the best algorithm among the proposed algorithm and the other six HHO versions. An evaluation of an algorithm's performance based on the number of test functions on which it performs better is not an appropriate method for selecting the best algorithm. Instead, the success order of the proposed algorithms should be considered across all test functions. The well-known non-parametric Friedman test is used to evaluate the efficiency of the proposed algorithm. The rank values of the proposed algorithms derived from the Friedman test are shown in Table 8. The rank for algorithm i for a specific problem j is denoted as r_i^j , such that $1 \leq i \leq n$ and $1 \leq j \leq k$, where n is the number of benchmark problems and k is the

number of tested algorithms. The average values are used to evaluate the performance of the algorithms more precisely. The average rank for each algorithm is denoted as R_j , which is the sum of the ranks for each algorithm across all performance metrics divided by the number of performance metrics (benchmark functions), which is calculated using Equation (21):

$$R_j = \frac{1}{n} \sum_i^n r_i^j \quad (21)$$

To determine which algorithm is more successful in the HHO algorithm, Table 8 provides the success scores of the HHO variants. During success scoring, for example, the HHO version that solves the F1 test function best, in terms of the mean value metric, receives one point, while the HHO version that solves the F1 test function worst receives seven points. With 33 total points, the proposed SHHOIRC algorithm is the most successful by obtaining the smallest number of total points. To apply the Friedman test, the average rank for each algorithm is calculated. The average ranks for each algorithm are shown in Table 8 in the last row. The best average rank is obtained by SHHOIRC, with a value of 4.2539.

The Friedman test is used to determine whether there are significant differences between the performances of multiple algorithms on multiple tasks. In this case, the test compares the performance of seven algorithms ($k = 7$) on 16 benchmark functions ($n = 16$). We can then use the Friedman test formula (Equation (22)) to calculate the test statistic:

$$F = \frac{12n}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \quad (22)$$

Friedman's measure is obtained with a value of 21.2076.

Table 8. Ranks of the proposed algorithm and the other HHO versions.

	HHO	SHHOIRC	CHHO PM1C2	CHHO PM1C9	CHHO PM2C2	CHHO PM2C9	SHHO
F1	4	1	2	5	7	3	6
F2	4	2	3	1	7	5	6
F3	7	1	2	6	5	4	3
F4	1	7	3	4	2	5	6
F5	4	4	4	4	4	4	4
F6	1.5	5	1.5	5	5	5	5
F7	4	4	4	4	4	4	4
F8	6	1	2	4	7	5	3
F9	5	1	2	6	4	7	3
F10	6	1	6	3	5	2	4
F11	2	1	2	4	6	5	7
F12	4	1	4	7	3	2	6
F13	5	1	2	3	7	6	4
F14	6	1	2	5	3	4	7
F15	3	1	2	6	4	5	7
F16	6	1	2	4	3	5	7
Total	69	33	44	71	76	71	82
Average rank	18.3291	4.2539	7.3916	19.6914	22.5625	19.6914	26.2656

To determine whether the calculated test statistic is significant, we need to compare it to the critical value from the χ^2 -distribution with df degree of freedom. The degrees of freedom df for the Friedman test are $k - 1$, where k is the number of algorithms being compared. In this case, $k = 7$, so $df = 6$. Using a significance level of 0.05, the critical value for $df = 6$ is 12.59. Since the calculated test statistic ($F = 21.2076$) is greater than the critical value (12.59), we can reject the null hypothesis that all algorithms perform equally well. We can conclude that there is at least one algorithm that performs significantly better or worse than the others.

To determine the best algorithm, the average ranks obtained in Table 8 are examined. The algorithm with the lowest average rank is considered the best. In this case, the algorithm with the lowest average rank is SHHOIRC, with an average rank of 4.2539. Therefore, SHHOIRC is considered the best algorithm based on the Friedman test.

In conclusion, the Friedman test indicates that there are significant differences in the performances of the seven algorithms on the 16 benchmark functions. The best-performing algorithm on average is SHHOIRC, followed by SHHO, HHO, CHHO PM1C2, CHHO PM2C2, CHHO PM1C9, and CHHO PM2C9.

4.3. Wireless Sensor Network Coverage Optimization

The maximum coverage problem of a WSN can be modeled as follows: It is assumed that we have m sensors that will be used to cover the monitoring region, and the set S of m sensors as $S = \{s_1, s_2, \dots, s_i, \dots, s_m\}$, where s_i is the i th sensor node, and its coordinate is (x_i, y_i) . And the coordinate of target point t is (x_t, y_t) , which is in the monitoring region.

In this paper, we used the Boolean (deterministic) sensing model, which is the most commonly used sensing model. In this model, if a point (target) t in the monitoring area is located within sensing range r_s of sensor node s_i , then it is assumed that t is covered by s_i . Therefore, the coverage function, $C(s_i, t)$, of sensor s_i and point t can be modeled as in Equation (23):

$$C(s_i, t) = \begin{cases} 1, & \text{if } d(s_i, t) \leq r_s \\ 0, & \text{otherwise} \end{cases} \quad (23)$$

where $d(s_i, t)$ is the Euclidean distance between sensor node s_i and target point t , and r_s is the sensing range.

The coverage of single target point t in the monitoring region by all the sensor nodes of the WSN is in Equation (24):

$$C(S, t) = \bigvee_{i=1}^m C(s_i, t) \quad (24)$$

To calculate the network's net coverage, we divide the observing area into g equal squares grids. We determine the number of grid points in row gh , and columns gw . Therefore, the number of grid points g is $gw \times gh$. The grid points are set as $G = \{t_1, t_2, \dots, t_j, \dots, t_g\}$, and the coordinate of target point t_j is (x_{t_j}, y_{t_j}) . Consequently, the definition of the WSN coverage rate is the ratio of the covered grid points, and the number of all grid points is modeled in Equation (25):

$$C(S, G) = \frac{\sum_{x_i=1}^{gw} \sum_{y_i=1}^{gh} C(S, t)}{gw \times gh} \quad (25)$$

For the sensor nodes set $S = \{s_1, s_2, \dots, s_i, \dots, s_m\}$ in the WSN, (x_i, y_i) , representing s_i coordinates. In HHO, each hawk represents a solution (sensor distribution) for the WSN maximum coverage problem. The encoding of each seahorse is:

$$X = \{x_1, y_1, x_2, y_2, \dots, x_i, y_i, \dots, x_m, y_m\} \quad (26)$$

The objective/fitness function used to evaluate each sensor distribution is modeled as (Equation (27)):

$$f(X) = -C = -\frac{\sum_{x_i=1}^{gw} \sum_{y_i=1}^{gh} C(s_{all}, t)}{gw \times gh} \quad (27)$$

Therefore, the coverage optimization problem can be defined as follows: the optimization variable is X , and in this optimization problem, we search for the best solution X to minimize $f(X)$, to find the maximal coverage.

Herein, we used a WSN that includes 30 sensor nodes, with sensing radius $r_s = 2.5$ m. These sensors are distributed in a 20×20 m² monitoring region. The HHO population size $PS = 30$. The number of grid points is set to 40^2 for a more accurate coverage rate. In this paper, we tested SHHOIRC, genuine HHO, GWO, and WOA on two maximum coverage experiments. In the first experiment, the maximum number of iterations in each run is 500, with a maximum number of runs of 10. In the second one, the maximum number of iterations in each run is 1000, with a maximum number of runs of 10. The results of the first and second experiments are shown in Tables 9 and 10, respectively.

The two experiments compare the proposed HHO version (SHHOIRC) with pure HHO and two well-known swarm intelligence algorithms (GWO and WOA) in terms of average solution, standard deviation values, best solutions obtained, worst solutions obtained, and average time consumed. The four algorithms are used to solve the stated maximum coverage problem for wireless sensor networks.

4.3.1. Experiment 1 (500 Iterations)

Table 9 shows the results obtained by the four algorithms for the stated maximum coverage problem (500 iterations). Table 1 shows comparisons between the solutions obtained in terms of average, best, and worst solutions, standard deviation values, and average time.

Table 9. Results obtained for maximum coverage problem by the four algorithms (500 iterations).

	AVG	Best	Worst	STD	AVG Time
SHHOIRC	92.45	95.4375	90.625	1.7194	1726.5590
HHO	90.0625	91.4375	88.625	1.1273	949.8674
GWO	87.1063	94.5	78.9375	6.0276	425.0602
WOA	87.7375	92.375	82.5625	2.6411	403.2416

Table 10. Results obtained for maximum coverage problem by the four algorithms (1000 iterations).

	AVG	Best	Worst	STD	AVG Time
SHHOIRC	93.8064	97.125	91.9375	1.6246	4183.2866
HHO	91.6313	93.125	90	0.9183	1947.2500
GWO	90.6313	96.125	79.125	4.6276	781.6393
WOA	89.2	92	85	2.2783	864.4674

In terms of average solutions obtained by the proposed algorithm SHHOIRC, HHO, GWO, and WOA, it is shown that the proposed algorithm SHHOIRC outperformed the other three algorithms. Based on this metric, SHHOIRC achieved the highest average score of 92.45%, followed by HHO (90.06%). The worst-performing algorithms in terms of average score were GWO (87.11%) and WOA (87.74%).

In terms of the best solution, the proposed algorithm SHHOIRC achieved the highest best solution with a value of 95.44%, followed by GWO (94.5%), and the lowest best solution was obtained by HHO with a value of 91.44%. The highest worst solution was obtained by

SHHOIRC with a value of 90.625%, and the lowest worst solution was obtained by GWO with a value of 78.94%.

The standard deviation metric shows the variability in fitness values achieved by each algorithm across multiple runs. Based on this metric, GWO had the highest standard deviation of 6.03, followed by WOA (2.64). The algorithms with the lowest standard deviation were HHO (1.13), and SHHOIRC (1.72). In terms of the average time consumed, SHHOIRC had the longest runtime of 1726.56 s. The algorithms with the shortest runtime were WOA (403.24) and GWO (425.06).

Figure 3a shows the convergence curve of average solutions obtained so far by the four algorithms, and Figure 3b shows the convergence curve of the best solutions obtained so far by the four algorithms. As shown in Figure 3, the highest solutions are obtained by the proposed algorithm SHHOIRC.

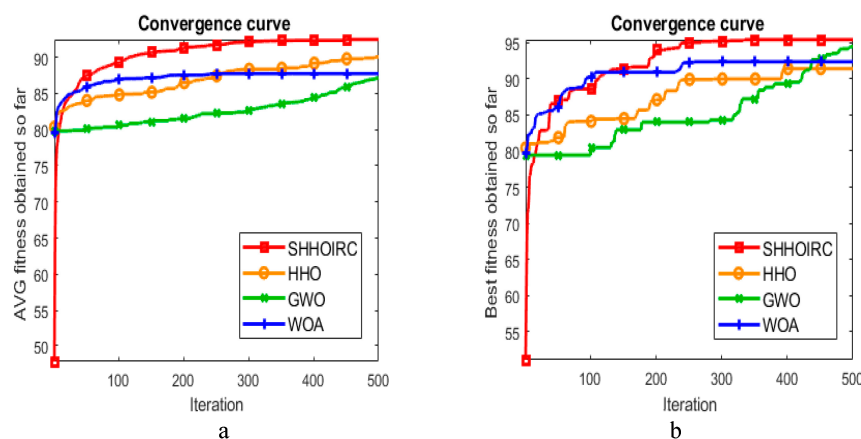


Figure 3. Maximum coverage convergence curve of average fitness obtained by the 4 algorithms (500 iterations): (a) for the average solutions and (b) for the best solutions.

Figure 4a shows the initial distribution of sensor nodes for the best solution obtained by SHHOIRC, and Figure 4b shows the final distribution of sensor nodes for the best solution obtained by SHHOIRC with a coverage rate of 95.4375%.

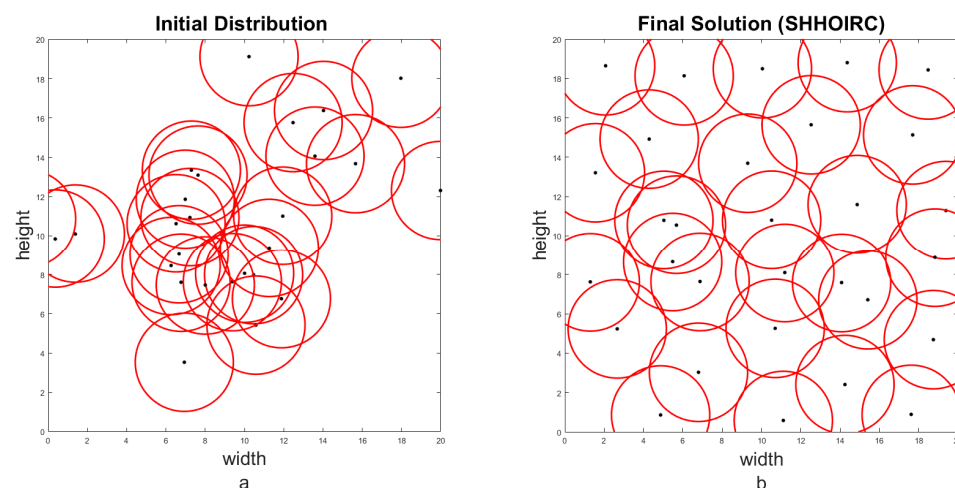


Figure 4. (a,b) Initial and final distribution of the best solution obtained by SHHOIRC (500 iterations).

4.3.2. Experiment 2 (1000 Iterations)

Table 10 shows the results obtained by SHHOIRC, HHO, GWO, and WOA for the stated maximum coverage problem (1000 iterations). Table 10 shows comparisons between the solutions obtained in terms of average, best, and worst solutions, standard deviation values, and average consumed time.

In terms of average solutions obtained by the proposed algorithm SHHOIRC and the three algorithms, it can be seen that the proposed algorithm SHHOIRC outperformed the other algorithms. In terms of the average solution, SHHOIRC obtained the best value of 93.81%. The worst average solution was obtained by WOA with a value of 89.2%. In terms of the best solution, the proposed algorithm SHHOIRC achieved the highest best solution with a value of 97.13%, and the lowest best solution was obtained by WOA with a value of 92%. On the other hand, the highest worst solution was obtained by SHHOIRC with a value of 91.94%, and the lowest worst solution was obtained by GWO with a value of 79.13%.

In terms of average time consumed, the highest time consumption was obtained by SHHOIRC with a value of 4183.29 s, and the lowest time consumption was obtained by GWO with a value of 781.64 s. Based on the STD metric, the lowest standard deviation value of 0.92 was obtained by the HHO algorithm, and the highest standard deviation value of 4.63 was obtained by GWO.

Figure 5a shows the convergence curve of average solutions obtained so far by the four algorithms, and Figure 5b shows the convergence curve of best solutions obtained so far by the four algorithms. As shown in Figure 5, the highest solutions are obtained by the proposed algorithm SHHOIRC. Figure 6a shows the initial distribution of sensor nodes for the best solution obtained by SHHOIRC, and Figure 6b shows the final distribution of sensor nodes for the best solution obtained by SHHOIRC with a coverage rate of 97.125%.

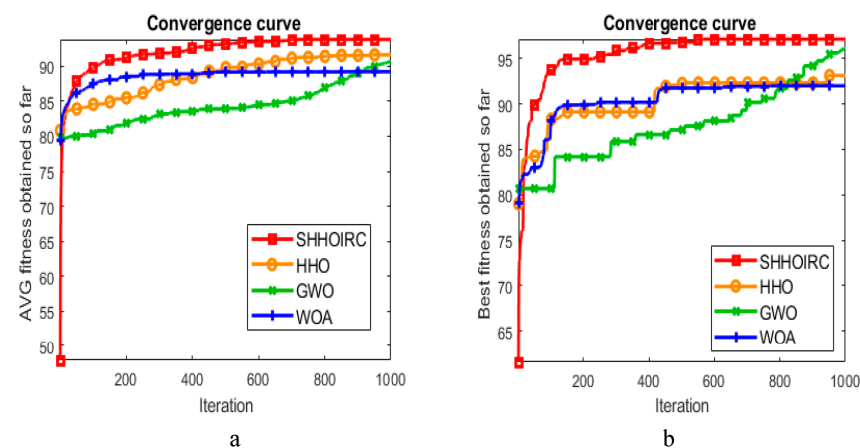


Figure 5. Maximum coverage convergence curve of average fitness obtained by the 4 algorithms (1000 iterations): (a) for the average solutions and (b) for the best solutions.

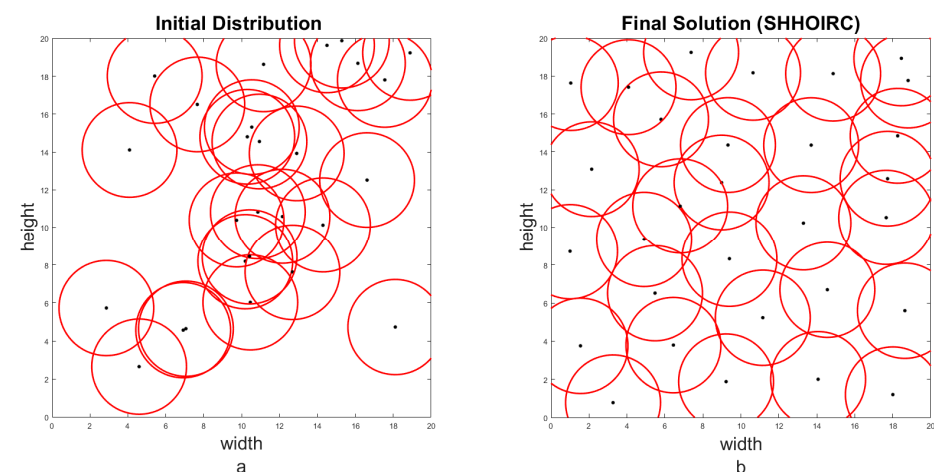


Figure 6. (a,b) Initial and final distribution of the best solution obtained by SHHOIRC (1000 iterations).

4.3.3. Statistical Analysis

For Experiment 1, we performed a one-way analysis of variance (ANOVA) to compare the mean performance of the algorithms. The results from one-way ANOVA for Experiment 1 are shown in Table 11. The one-way ANOVA test conducted on the data from Experiment 1 yielded an F-statistic of 4.98. To determine whether the calculated test statistic is significant, we need to compare it with the critical value from the F -distribution with $df1$ and $df2$ degrees of freedom. The degrees of freedom $df1$ and $df2$ for the one-way ANOVA test are $k - 1$ and $k(n - 1)$, respectively, where k is the number of algorithms being compared and n is the number of runs. In this case, $k = 4$ and $n = 10$, so $df1 = 3$ and $df2 = 36$. Using a significance level of 0.05, the critical value for the stated $df1$ and $df2$ is 2.87. Since the calculated test statistic ($F = 4.98$) is greater than the critical value (2.87), we can reject the null hypothesis that all algorithms perform equally well. We can conclude that there is at least one algorithm that performs significantly better or worse than the others. The corresponding p -value is 0.0054, which is less than 0.05. In this case, the p -value of 0.0054 suggests that the observed differences in mean performances among the algorithms in Experiment 1 are statistically significant.

Table 11. Results obtained by one-way ANOVA for Experiment 1.

Source of Variation	Sum of Squares (SS)	Degrees of Freedom (df)	Mean Sum of Squares (MSS)	F-Computed (Fc)	p -Value
Columns	$SST_r = 177.517$	$df1 = 3$	$MSSTr = 59.1725$	4.98	0.0054
Error	$SSE = 427.81$	$df2 = 36$	$MSSE = 11.8836$		
Total	$SST = 605.327$	39			

Moving on to Experiment 2, the results from one-way ANOVA for Experiment 2 are shown in Table 12. The one-way ANOVA test conducted on the data from Experiment 2 yielded an F-statistic of 4.98. By comparing the calculated test statistic ($F = 4.98$) with the critical value (2.87) for the stated $df1$ and $df2$ using a significance level of 0.05, it is notably greater. Therefore, we can reject the null hypothesis that all algorithms perform equally well and conclude that there is at least one algorithm that performs significantly better or worse than the others. Since the corresponding p -value is 0.0054, the low p -value (less than 0.05) indicates that the observed differences in mean performances among the algorithms in Experiment 2 are statistically significant.

Table 12. Results obtained by one-way ANOVA for Experiment 2.

Source of Variation	Sum of Squares (SS)	Degrees of Freedom (df)	Mean Sum of Squares (MSS)	F-Computed (Fc)	p -Value
Columns	$SST_r = 122.47$	$df1 = 3$	$MSSTr = 37.4902$	4.98	0.0054
Error	$SSE = 270.79$	$df2 = 36$	$MSSE = 7.522$		
Total	$SST = 383.261$	39			

From the boxplot of Experiments 1 and 2 shown in Figure 7a,b, it can be observed that the SHHOIRC boxplot is located higher than the other algorithms. The '+' marks represent the outliers.

Based on the statistical analysis performed on the data from Experiments 1 and 2, we can conclude that SHHOIRC has a significantly higher mean than the other algorithms tested. Specifically, SHHOIRC showed a significantly higher mean performance than pure HHO, GWO, and WOA. The significant differences in mean performances, supported by the calculated p -values, suggest that SHHOIRC is the best algorithm among the four algorithms in these experiments.

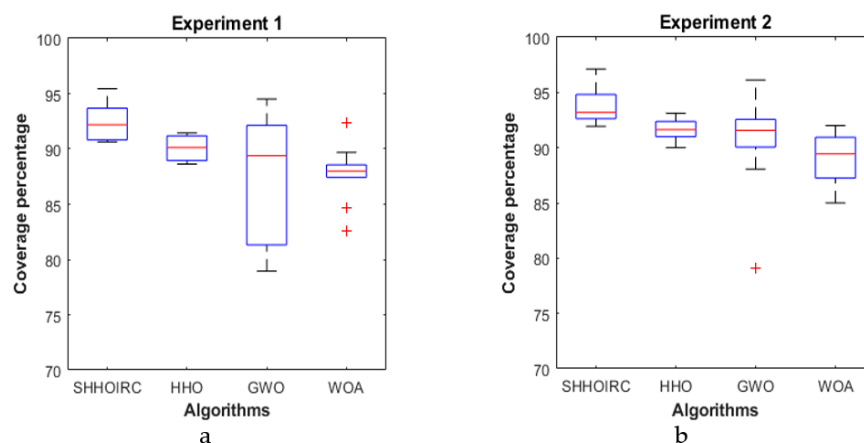


Figure 7. (a,b) boxplot of Experiments 1 and 2.

5. Conclusions

The proposed novel hybrid algorithm, SHHOIRC, incorporates self-adaptation and combines the HHO algorithm with three chaotic optimization techniques. On 16 well-known benchmark functions, the proposed algorithm was evaluated, and its performance was compared to that of authentic HHO and five other HHO variants. The proposed algorithm SHHOIRC significantly improves the performance of HHO. SHHOIRC obtained the best results in terms of average solution and best average solutions with the lowest standard deviation for 13 benchmark functions. In addition, statistical analyses employing the Friedman test and success scores confirmed that SHHOIRC is the most effective algorithm among the seven HHO variants evaluated. These results indicate that the proposed hybridization approach, which combines HHO with three chaotic optimization methods and incorporates self-adaptation, can be used to enhance the performance of other metaheuristic algorithms. The novel hybrid algorithm SHHOIRC and six other HHO variants are evaluated using sixteen well-known benchmark functions. The proposed algorithm SHHOIRC significantly improves the efficiency of HHO. The simulation experiment and statistical analysis demonstrate that the new hybrid algorithm SHHOIRC outperformed the other four CHHO variants, SHHO, and HHO. Furthermore, the proposed algorithm was tested on a real-life problem, which is the maximum coverage problem of WSN, and compared with genuine HHO, GWO, and WOA. The results of the experiments indicate that the proposed algorithm SHHOIRC performs better than the other algorithms tested in solving the maximum coverage problem for WSN. In both Experiments 1 and 2, SHHOIRC achieved the highest average solution and best solution values, while also having a relatively low standard deviation value. The statistical analysis of the results from both experiments shows that SHHOIRC has a significantly higher mean performance than the other algorithms tested. This indicates that SHHOIRC is a promising algorithm for solving maximum coverage problems in WSN and could be further developed and optimized for other related problems. The proposed algorithm SHHOIRC exhibits considerable improvement in HHO performance.

In the future, we intend to apply SHHOIRC to more real-world problems, and we anticipate achieving new breakthroughs in their resolution. In addition, we intend to introduce novel integration techniques to assist metaheuristic algorithms in escaping each local minimum.

Author Contributions: Conceptualization, S.A.; methodology, S.A.; software, E.B.; validation, E.B.; formal analysis, M.A.S.; investigation, M.A.S.; resources, A.D.; data curation, A.D.; writing—original draft preparation, A.D.; writing—review and editing, E.B.; visualization, A.D.; supervision, E.B.; project administration, M.A.S.; funding acquisition, S.A. All authors have read and agreed to the published version of the manuscript.

Funding: The author would like to thank Deanship of Scientific Research at Majmaah University for supporting this work under Project Number No. R-2023-629.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The author would like to thank Deanship of Scientific Research at Majmaah University for supporting this work under Project Number No. R-2023-629. We also thank the referees for suggestions to improve the presentation of this paper.

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

References

1. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science IEEE, MHS'95, Nagoya, Japan, 4–6 October 1995.
2. Holland, J. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: New York, NY, USA, 1992.
3. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [\[CrossRef\]](#)
4. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [\[CrossRef\]](#)
5. Yang, X.S.; Durand-Lose, J.; Jonoska, N. Flower Pollination Algorithm for Global Optimization. In *Unconventional Computation and Natural Computation, Proceedings of the 19th International Conference, UCNC 2021, Espoo, Finland, 18–22 October 2021*; Springer: Berlin/Heidelberg, Germany, 2012.
6. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [\[CrossRef\]](#)
7. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [\[CrossRef\]](#)
8. Karaboga, D.; Basturk, B.; Melin, P.; Castillo, O.; Aguilar, L.T.; Kacprzyk, J.; Pedrycz, W. Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems. In *Foundations of Fuzzy Logic and Soft Computing, Proceedings of the 12th International Fuzzy Systems Association World Congress, IFSA 2007, Cancun, Mexico, 18–21 June 2007*; Springer: Berlin/Heidelberg, Germany, 2007.
9. Yang, X.S.; González, J.R.; Pelta, D.A.; Cruz, C.; Terrazas, G.; Krasnogor, N. A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010.
10. Naruei, I.; Keynia, F. Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems. *Eng. Comput.* **2022**, *38*, 3025–3056. [\[CrossRef\]](#)
11. Abualigah, L.; Elaziz, M.A.; Hussien, A.G.; Alsabir, B.; Jalali, S.M.J.; Gandomi, A.H. Lightning search algorithm: A comprehensive survey. *Appl. Intell.* **2021**, *51*, 2353–2376. [\[CrossRef\]](#)
12. Dorigo, M.; Di Caro, G. Ant colony optimization: A new meta-heuristic. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999.
13. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [\[CrossRef\]](#)
14. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [\[CrossRef\]](#)
15. Tripathy, B.K.; Maddikunta, P.K.R.; Pham, Q.V.; Gadekallu, T.R.; Dev, K.; Pandya, S.; ElHalawany, B.M. Harris Hawk Optimization: A Survey on Variants and Applications. *Comput. Intell. Neurosci.* **2022**, *2022*, 2218594. [\[CrossRef\]](#)
16. Hussien, A.G.; Abualigah, L.; Abu Zitar, R.; Hashim, F.A.; Amin, M.; Saber, A.; Almotairi, K.H.; Gandomi, A.H. Recent Advances in Harris Hawks Optimization: A Comparative Study and Applications. *Electronics* **2022**, *11*, 1919. [\[CrossRef\]](#)
17. Too, J.; Abdullah, A.; Mohd Saad, N. A new quadratic binary harris hawk optimization for feature selection. *Electronics* **2019**, *8*, 1130. [\[CrossRef\]](#)
18. Hans, R.; Kaur, H.; Kaur, N. Opposition-based Harris Hawks optimization algorithm for feature selection in breast mass. *J. Interdiscip. Math.* **2020**, *23*, 97–106. [\[CrossRef\]](#)
19. Menesy, A.S.; Sultan, H.M.; Selim, A.; Ashmawy, M.G.; Kamel, S. Developing and applying chaotic harris hawks optimization Technique for Extracting Parameters of Several Proton Exchange Membrane Fuel Cell Stacks. *IEEE Access* **2020**, *8*, 1146–1159. [\[CrossRef\]](#)
20. Basha, J.; Bacanin, N.; Vukobrat, N.; Zivkovic, M.; Venkatachalam, K.; Hubálovský, S.; Trojovský, P. Chaotic Harris Hawks Optimization with Quasi-Reflection-Based Learning: An Application to Enhance CNN Design. *Sensors* **2021**, *21*, 6654. [\[CrossRef\]](#) [\[PubMed\]](#)
21. Hussien, A.G.; Amin, M. A self-adaptive Harris Hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection, *Int. J. Mach. Learn. Cybern.* **2022**, *13*, 1–28. [\[CrossRef\]](#)
22. Mirjalili, S.R. Multi-Objective Harris hawks optimization: A novel algorithm for multi-objective engineering design problems, *Eng. Comput.* **2020**, *36*, 879–891.

23. Shi, B.; Heidari, A.A.; Chen, C.; Wang, M.; Huang, C.; Chen, H.; Zhu, J. Predicting Di-2-Ethylhexyl Phthalate Toxicity: Hybrid Integrated Harris Hawks Optimization with Support Vector Machines. *IEEE Access* **2020**, *8*, 161188–161202. [\[CrossRef\]](#)
24. Zou, T.; Wang, C. Adaptive Relative Reflection Harris Hawks Optimization for Global Optimization. *Mathematics* **2022**, *10*, 1145. [\[CrossRef\]](#)
25. Birolgul, S. Hybrid Harris Hawk Optimization Based on Differential Evolution (HHODE) Algorithm for Optimal Power Flow Problem. *IEEE Access* **2019**, *7*, 184468–184488. [\[CrossRef\]](#)
26. Zhang, Y.; Zhou, X.; Shih, P.-C. Modified Harris Hawks Optimization Algorithm for Global Optimization Problems. *Arab. J. Sci. Eng.* **2020**, *45*, 10949–10974. [\[CrossRef\]](#)
27. Gharehchopogh, F.S. An Improved Harris Hawks Optimization Algorithm with Multi-strategy for Community Detection in Social Network. *J. Bionic Eng.* **2023**, *20*, 1175–1197. [\[CrossRef\]](#)
28. Sihwail, R.; Omar, K.; Ariffin, K.A.Z.; Tubishat, M. Improved Harris Hawks Optimization Using Elite Opposition-Based Learning and Novel Search Mechanism for Feature Selection. *IEEE Access* **2020**, *8*, 121127–121145. [\[CrossRef\]](#)
29. Kaur, G.; Arora, S. Chaotic whale optimization algorithm. *J. Comput. Des. Eng.* **2018**, *5*, 275–284. [\[CrossRef\]](#)
30. Yang, D.; Li, G.; Cheng, G. On the efficiency of chaos optimization algorithms for global optimization. *Chaos Solitons Fractals* **2007**, *34*, 1366–1375. [\[CrossRef\]](#)
31. Harun, G.; Haydar, L. Chaotic Harris hawks optimization algorithm. *J. Comput. Des. Eng.* **2022**, *9*, 216–245.
32. Xia, J. Coverage Optimization Strategy of Wireless Sensor Network Based on Swarm Intelligence Algorithm. In Proceedings of the 2016 International Conference on Smart City and Systems Engineering (ICSCSE), Hunan, China, 25–26 November 2016.
33. Yang, S.H. *Wireless Sensor Networks*; Springer: London, UK, 2014.
34. Elhabyan, R.; Shi, W.; St-Hilaire, M. Coverage Protocols for Wireless Sensor Networks: Review and Future Directions. *J. Commun. Netw.* **2019**, *21*, 45–60. [\[CrossRef\]](#)
35. Hu, J.; Song Jingyan, J.; Zhang, M.; Kang, X. Topology Optimization for Urban Traffic Sensor Network. *Tsinghua Sci. Technol.* **2008**, *13*, 2. [\[CrossRef\]](#)
36. Aziz, M.; Tayarani, N.M.-H.; Meybodi, M.R. A two-objective memetic approach for the node localization problem in wireless sensor networks. *Genet. Program. Evolvable Mach.* **2016**, *17*, 4. [\[CrossRef\]](#)
37. Ngatchou, P.N.; Fox, W.L.J.; El-Sharkawi, M.A. Distributed sensor placement with sequential particle swarm optimization. In Proceedings of the 2005 IEEE Swarm Intelligence Symposium, SIS 2005, Pasadena, CA, USA, 8–10 June 2005.
38. Wang, X.; Wang, S.; Ma, J.-J. An Improved Co-evolutionary Particle Swarm Optimization for Wireless Sensor Networks with Dynamic Deployment. *Sensors* **2007**, *7*, 354–370. [\[CrossRef\]](#)
39. Wang, S.; You, H.; Yue, Y.; Cao, L. A novel topology optimization of coverage-oriented strategy for wireless sensor networks. *Int. J. Distrib. Sens. Netw.* **2021**, *17*, 15501477219. [\[CrossRef\]](#)
40. Alshaqqawi, B.; Haque, S.A.; Alreshoodi, M.; Alsukayti, I. Enhanced Particle Swarm Optimization for Effective Relay Nodes Deployment in Wireless Sensor Networks. *Int. J. Comput. Netw. Commun.* **2021**, *13*, 53–73. [\[CrossRef\]](#)
41. Liang, D.; Shen, H.; Chen, L. Maximum Target Coverage Problem in Mobile Wireless Sensor Networks. *Sensors* **2021**, *21*, 184. [\[CrossRef\]](#) [\[PubMed\]](#)
42. Wang, G. Research on Wireless Sensor Network Technology Based on Particle Swarm Algorithm. *J. Phys. Conf. Ser.* **2021**, *1865*, 4. [\[CrossRef\]](#)
43. Mehta, S.; Malik, A. A Swarm Intelligence Based Coverage Hole Healing Approach for Wireless Sensor Networks. *EAI Endorsed Trans. Scalable Inf. Syst.* **2020**, *7*, 26. [\[CrossRef\]](#)
44. Tarnaris, K.; Preka, I.; Kandris, D.; Alexandridis, A. Coverage and k-Coverage Optimization in Wireless Sensor Networks Using Computational Intelligence Methods: A Comparative Study. *Electronics* **2020**, *9*, 675. [\[CrossRef\]](#)
45. Bonnah, E.; Ju, S.; Cai, W. Coverage Maximization in Wireless Sensor Networks Using Minimal Exposure Path and Particle Swarm Optimization. *Sens. Imaging* **2020**, *21*, 1–16. [\[CrossRef\]](#)
46. Li, Q.; Liu, N. Coverage optimization algorithm based on control nodes position in wireless sensor networks. *Int. J. Commun. Syst.* **2020**, *35*, e4599. [\[CrossRef\]](#)
47. Zhang, Y. Coverage Optimization and Simulation of Wireless Sensor Networks Based on Particle Swarm Optimization. *Int. J. Wirel. Inf. Netw.* **2020**, *27*, 307–316. [\[CrossRef\]](#)
48. Gökalp, O. Optimizing Connected Target Coverage in Wireless Sensor Networks Using Self-Adaptive Differential Evolution. *Balk. J. Electr. Comput. Eng.* **2020**, *8*, 325–330. [\[CrossRef\]](#)
49. Zhang, Y.; Zhang, Z. Wireless Sensor Network Node Deployment Based On Regular Tetrahedron. *J. Phys. Conf. Ser.* **2020**, *1453*, 012113. [\[CrossRef\]](#)
50. Wang, J.; Ju, C.; Kim, H.-J.; Sherratt, R.S.; Lee, S. A Mobile Assisted Coverage Hole Patching Scheme based on Particle Swarm Optimization for WSNs. *Clust. Comput.* **2019**, *22*, 1. [\[CrossRef\]](#)
51. Mnasri, S.; Nasri, N.; Van den Bossche, A.; Val, T. A new multi-agent particle swarm algorithm based on birds accents for the 3D indoor deployment problem. *ISA Trans.* **2019**, *91*, 262–280. [\[CrossRef\]](#) [\[PubMed\]](#)
52. Syed, M.A.; Syed, R. Weighted Salp Swarm Algorithm and its applications towards optimal sensor deployment. *J. King Saud Univ. Comput. Inf. Sci.* **2019**, *34*, 4. [\[CrossRef\]](#)
53. Kohli, M.; Arora, S. Chaotic grey wolf optimization algorithm for constrained optimization problems. *J. Comput. Des. Eng.* **2017**, *5*, 458–472. [\[CrossRef\]](#)

54. Gao, Z.M.; Zhao, J.; Zhang, Y.J. Review of chaotic mapping enabled nature-inspired algorithms. *Math. Biosci. Eng. MBE* **2022**, *19*, 8215–8258. [[PubMed](#)]
55. Xiang, T.; Liao, X.; Wong, K.W. An improved particle swarm optimization algorithm combined with piecewise linear chaotic map. *Appl. Math. Comput.* **2007**, *190*, 1637–1645. [[CrossRef](#)]
56. Gandomi, A.H.; Yang, X.-S. Chaotic bat algorithm. *J. Comput. Sci.* **2014**, *5*, 224–232. [[CrossRef](#)]
57. Köker, R.; Gao, Z.-M.; Zhao, J. An Improved Grey Wolf Optimization Algorithm with Variable Weights. *Comput. Intell. Neurosci.* **2019**, *2019*, 2981282.
58. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.