sensors

MDPI

Article

Adaptive Aquila Optimizer Combining Niche Thought with Dispersed Chaotic Swarm

Yue Zhang, Xiping Xu *, Ning Zhang, Kailin Zhang, Weida Dong and Xiaoyan Li

School of Opto-Electronic Engineering, Changchun University of Science and Technology, Changchun 130022, China
* Correspondence: xxp_educust@163.com

Abstract: The Aquila Optimizer (AO) is a new bio-inspired meta-heuristic algorithm inspired by Aquila's hunting behavior. Adaptive Aquila Optimizer Combining Niche Thought with Dispersed Chaotic Swarm (NCAAO) is proposed to address the problem that although the Aquila Optimizer (AO) has a strong global exploration capability, it has an insufficient local exploitation capability and a slow convergence rate. First, to improve the diversity of populations in the algorithm and the uniformity of distribution in the search space, DLCS chaotic mapping is used to generate the initial populations so that the algorithm is in a better exploration state. Then, to improve the search accuracy of the algorithm, an adaptive adjustment strategy of de-searching preferences is proposed. The exploration and development phases of the NCAAO algorithm are effectively balanced by changing the search threshold and introducing the position weight parameter to adaptively adjust the search process. Finally, the idea of small habitats is effectively used to promote the exchange of information between groups and accelerate the rapid convergence of groups to the optimal solution. To verify the optimization performance of the NCAAO algorithm, the improved algorithm was tested on 15 standard benchmark functions, the Wilcoxon rank sum test, and engineering optimization problems to test the optimization-seeking ability of the improved algorithm. The experimental results show that the NCAAO algorithm has better search performance and faster convergence speed compared with other intelligent algorithms.

Keywords: Aquila Optimizer (AO); chaotic distribution; adaptive adjustment; Niche; global optimization; swarm intelligence

check for updates

Citation: Zhang, Y.; Xu, X.; Zhang, N.; Zhang, K.; Dong, W.; Li, X. Adaptive Aquila Optimizer Combining Niche Thought with Dispersed Chaotic Swarm. *Sensors* **2023**, *23*, 755. https://doi.org/10.3390/s23020755

Received: 28 November 2022
Revised: 23 December 2022
Accepted: 7 January 2023
Published: 9 January 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

With the continuous development of science and technology, the effective integration of life sciences and engineering sciences has become the main feature of modern science and technology [1], and meta-heuristic algorithms have flourished under this trend. Currently, optimization objectives in practical engineering applications range from single objective to multi-objective, from continuous to discrete, and from constrained to unconstrained, with complex problems. The drawbacks of traditional optimization algorithms are obviously due to their heavy reliance on initial values and their tendency to fall into local optima. Meta-heuristic optimization algorithms are easy to implement, bypass local optima, are non-derivative mechanisms, and do not require gradient information [2–5]. Compared with the two types of algorithms, the performance of meta-heuristic algorithms is outstanding and has received widespread attention from scholars; they have not only played an important role in the field of computing, but have also shown powerful problem-solving capabilities in the fields of military [6], agriculture [7] and hydraulic engineering [8].

Since the series of algorithms such as Particle Swarm Optimization [9] (PSO), Genetic Algorithm [10] (GA) and Artificial Bee Colony [11] (ABC) have been proposed, they are continuously applied in many engineering examples and provide better solutions to the problems [12]. In the process of rapid development, scholars mainly focus on population

initialization, population exploration and how to quickly converge to the global optimum to study meta-heuristic algorithms in depth. Abualigah L et al. proposed a new meta heuristic algorithm in 2021: Aquila Optimizer (AO) [13]. The algorithm, inspired by the prey-catching behavior of the Aquila, has demonstrated in the generalized arithmetic test that AO has a high performance and high local optimum capability in the exploration and development stages and in the process of handling composite functions compared to 11 algorithms such as GOA, EO, DA, PSO, ALO, GWO, and SSA. However, its local exploitation capability is insufficient and easy to fall into a local optimum and slow convergence, and many scholars have improved it. Wang S et al. [14] combined the development phase of Harris Hawks Optimization (HHO) with the exploration phase of AO to propose an improved hybrid Aquila Optimizer and Harris Hawks Optimization (IHAOHHO), combining both nonlinear escape energy parameters and stochastic opposition-based learning strategies to enhance the exploration and development of algorithms in standard test functions and industrial engineering design problems, demonstrating strong superior performance and good promise. Verma M et al. [15] generated a population by a standard AO and a new population by a single-stage genetic algorithm based on the concept of evolution, where binary tournament selection, roulette wheel selection, shuffle crossover and replacement mutations occur to generate a new population. The chaotic mapping criterion is then applied to obtain various variants of the standard AO technique. The standard AO is further improved, yields better results and is applied to engineering design problems. However, the effect of the homogeneity of chaotic systems on population initialization is not considered. Akyol S [16] used the tangent search algorithm with an intensive phase (TSA) to optimize the AO and proposes the Skyhawk Optimizer tangent search algorithm (AO-TSA). The enhancement phase (TSA) of the tangent search algorithm is used instead of the limited exploration phase to improve the exploitation capability of the Aquila Optimizer (AO), while the local minimum escape phase of the TSA is applied in the AO-TSA to avoid the local minimum stagnation problem. Mahajan S et al. [17] proposed a new hybrid approach using Aquila Optimizer (AO) and Arithmetic Optimization Algorithm (AOA) by combining AO with Arithmetic Optimization Algorithm (AOA). Efficient search is achieved in high- and low-dimensional problems, further showing that the population-based approach achieves efficient search results in high-dimensional problems. Zhang Y J et al. [18] similarly combined AO with Arithmetic Optimization Algorithm (AOA) and proposed The Hybrid Algorithm of Arithmetic Optimization Algorithm with Aquila Optimizer (AOAAO). An Individual exploration and development process in the energy parameter E equilibrium population is also introduced, and segmented linear mappings are introduced to reduce the randomness of the energy parameters. The proposed AOAAO is experimentally validated to have a faster convergence speed and higher convergence accuracy.

In addition, AIRassas A M et al. developed an improved Adaptive Neuro-Fuzzy Inference System (ANFIS) via AO: AO-ANFIS [19] for predicting oil production and experimentally demonstrated that AO significantly improved the prediction accuracy of ANFIS. Abd Elaziz M et al. [20] proposed a new image of COVID-19 classification framework using DL and a hybrid swarm-based algorithm that use AO as a feature selector to reduce the dimensionality of the image. The experimental results showed that the performance metrics of feature extraction and selection stages are more accurate than other methods. Jnr E O N et al. [21] proposed a new DWT-PSR-AOA-BPNN model for wind speed prediction and for efficient grid operation by combining the Skyhawk Optimization Algorithm (AOA) with Discrete Wavelet Transform (DWT), Phase Space Reconfiguration (PSR) with Chaos Theory, and Back Propagation Neural Network (BPNN). Meanwhile, AO has been applied to population prediction [22], PID parameter optimization [23], power generation allocation [24], path planning [25] and other fields. Due to the short time of AO proposal, its slow convergence speed and its tendency to fall into local optimum still need further optimization and improvement. It is important to further improve its performance and make it applicable to more practical problems.

The literature shows that these algorithms can effectively approach the true expectation of a multi-objective problem. The problem of zero search preferences, however, has not attracted much attention from scholars. The effective solution of this problem is the basis for improving the robustness of the AO algorithm and allows the algorithm to be applied to new classes of problems or to propose new optimization algorithms. This is the purpose and significance of this work, in which we propose a new algorithm called Adaptive Aquila Optimizer Combining Niche Thought with Dispersed Chaotic Swarm (NCAAO), based on the recently proposed Aquila optimizer (AO). The research gaps and contributions of the paper can be summarized as follows:

- The NCAAO has great potential in solving various engineering problems such as feature selection, image segmentation, image enhancement and engineering design.
- The AO and NCAAO have rarely been combined with other available meta-heuristics; therefore, the NCAAO has a great potential to be combined with other algorithms available in the literature.
- Hence, a new chaotic mapping is introduced which has better uniformity, better initializes the population, and effectively enhances the efficiency of the algorithm.
- This new algorithm is further investigated in terms of search thresholds, which allows the population to adaptively adjust the search space by changing the search threshold. The introduced adaptive weight parameter perturbs the population of individual positions and improves the algorithm local exploitation ability.
- Moreover, for the algorithm to obtain the optimal solution with high accuracy, a communication exchange strategy based on the idea of Niche thought is proposed to ensure the optimization-seeking accuracy and convergence speed of the algorithm by screening elite individuals.

In summary, in order to solve the problem that the convergence of AO is slow and easy to fall into local optimum, firstly and as inspired by the literature [26], the method of initializing population is adopted to improve the robustness of the algorithm. The Aquila population is initialized by DLCS chaotic mapping. The approach we have used in this study aims to enhance the homogeneity of the Aquila population in the search space and improve the randomness and ergodicity of the population individuals. Second, an adaptive adjustment search strategy for de-searching preferences is proposed to balance global exploitation and local search by the adaptive threshold adjustment method. On this basis, the adaptive location weighting method is introduced to perturb the update of the group individuals' location and improve the local exploitation ability. The de-search preference adaptive adjustment search strategy can remove the search preference problem effectively to a certain extent and help the algorithm escape from the local optimum quickly. Finally, inspired by literature [27] and combined with the elite solution mechanism, a communication exchange strategy based on the idea of Niche thought is used to ensure that the goodness of the population is maintained during the iterative process. Through the information exchange among individuals, the global optimum is better selected and the rapid convergence of the scout individuals to the global optimum is promoted to enhance the robustness of the algorithm.

The rest of the paper is structured as follows: Section 2 describes the principle of AO and studies its module performance. Section 3 describes the proposed algorithm and analyzes the time complexity of the proposed algorithm. Section 4 verifies the robustness and applicability of the algorithm through numerical experiments and engineering experiments. Section 5 summarizes the full text and examines the future research direction.

2. Aquila Optimizer

2.1. Mathematical Model of AO

The AO algorithm simulates the different hunting styles of Aquila for different prey. For fast-moving prey, the Aquila needs to acquire the prey in a fast and precise way, by which the global exploration ability of the algorithm is reflected. Similarly, the local ex-

ploitation ability of the algorithm is reflected by the hunting way for slow-moving prey. The optimization process is represented by simulating four types of Aquila hunting behaviors.

First, the population needs to be generated randomly between the upper bound (UB) and lower bound (LB), specified based on the problem, as shown in Equation (1). The approximate optimal solution during the iterative process up to each iteration is determined as the optimal solution. The current set of candidate solutions X is randomly generated by Equation (2).

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,D} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,D} \end{bmatrix} \tag{1}$$

$$X_{i,j} = rand \times (UB_j - LB_j) + LB_j, i = 1, 2, \ldots, N \; j = 1, 2, \ldots D \tag{2}$$

where n denotes the total number of candidate solutions, D denotes the dimensionality of the problem and $x_{n,D}$ denotes the position of the nth solution in the Dth dimensional space. Rand is a random number, UB_j denotes the jth dimensional upper bound of the given problem and LB_j denotes the jth dimensional lower bound of the given problem.

2.1.1. Expanded Exploration(X_1)

The first way is to select the search space by flying high in a vertical bend. Aquila flies high to recognize prey areas and quickly select the best prey area. This behavior is shown by Equation (3).

$$X_1(t+1) = X_{best}(t) \times (1 - \frac{t}{T}) + (X_M(t) - X_{best}(t)) \times rand \tag{3}$$

$$X_M(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t), \forall j = 1, 2, \ldots, D \tag{4}$$

where $X_1(t+1)$ denotes the position of an individual at time $t+1$, $X_{best}(t+1)$ denotes the current global best position at the tth iteration, T and t denote the maximum number of iterations and the current number of iterations, respectively, $X_M(t)$ denotes the average position of an individual in the current iteration, and *Rand* is a random number between 0 and 1 in the Gaussian distribution.

2.1.2. Narrowing the Scope of Exploration (X_2)

The second approach is short gliding attacks in isometric flight. Aquilas hover above the target prey in preparation for an attack when they spot the prey area from a high altitude. This behavior is represented by Equation (5).

$$X_2(t+1) = X_{best}(t) \times levy(D) + X_R(t) + (y - x) \times rand \tag{5}$$

$$levy(D) = s \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \tag{6}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right) \tag{7}$$

where $X_2(t+1)$ is the generated solution for the next iteration of t, D denotes the spatial dimension, *levy(D)* is the Lévy flight distribution function, $X_R(t)$ is the random position of the Aquila in [1,N], s takes the value of 1.5, and y and x present a spiral situation in the search space as shown in the following equation:

$$y = r \times \cos(\theta) \tag{8}$$

$$x = r \times \sin(\theta) \tag{9}$$

$$r = r_1 + 0.00565 \times D_1 \tag{10}$$

$$\theta = -0.005 \times D_1 + \frac{3 \times \pi}{2} \tag{11}$$

where r_1 takes a fixed index between 1 and 20, and D_1 is an integer from 1 to the length of the search space.

2.1.3. Expanded Development (X_3)

The third way is a low-flying and slow-descent attack. The Aquila locks onto a hunting target in the hunting area, and with the attack ready, makes an initial attack in a vertical descent, thus further testing the prey's response. This behavior is represented by Equation (12).

$$X_3(t+1) = (X_{best}(t) - X_M(t)) \times \alpha - rand + ((UB - LB) \times rand + LB) \times \delta \tag{12}$$

where $X_3(t + 1)$ is the solution of the next iteration of t generated, δ and α are both mining adjustment parameters in the range of (0,1), and UB and LB represent the upper and lower bounds of the given problem.

2.1.4. Narrowing the Development Area(X_4)

The fourth way is walking and grabbing prey. When the Aquila approaches the prey, it attacks the prey according to the random movement of the prey. This behavior is represented by Equation (13).

$$X_4(t+1) = QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times levy(D) \tag{13}$$

$$QF(t) = t^{\frac{2 \times rand - 1}{(1-T)^2}} \tag{14}$$

$$G_1 = 2 \times rand - 1 \tag{15}$$

$$G_1 = 2 \times (1 - \frac{t}{T}) \tag{16}$$

where $X_4(t + 1)$ is the generated solution for the next iteration of t; QF denotes the mass function used to balance the search strategy and $F \in (0, 1)$; G_1 denotes the different methods used by the Aquila for prey escape; G_2 represents the slope value from the first position to the last position during the Aquila's prey chase, taking a decreasing value from 2 to 0; *Rand* is a random number between 0 and 1 in the Gaussian distribution; and T and t denote the maximum number of iterations and the current number of iterations, respectively.

2.2. AO module Performance Split Study

In order to extract the modules with more prominent optimization performance, the standard AO modules are separated and tested. The other modules are masked in the code source, one module is selected to update the individual positions, and the typical arithmetic cases F1, F10, and F13 in the literature [5] are selected to run the AO after the separation optimization process for ten times; the data are shown in Table 1, where *F_fit* is the average fitness of the ten times.

Table 1. Standard AO optimization process performance splitting.

Test Module	F_1_fit	F_{10}_fit	F_{13}_fit
Equation (3)	2.21×10^{-160}	8.4767	3.3478
Equation (5)	86.42499	0.996	3.2516
Equation (12)	1.53×10^{-4}	12.6705	5.3837
Equation (13)	0.0018084	1.998	3.0893

Two phases in the AO algorithm introduce a greedy algorithm to perturb the individual positions to ensure that the new position of the individual is better than the original position.

The four positions' update formulas are observed by the module separation test described above. In Equation (3), when the optimal solution is 0, it converges to 0. When the optimal solution is not 0, it converges to a constant; in Equation (5), it barely converges to the optimal solution; and in Equations (12) and (13), it mostly converges to a constant. Through the analysis of Equation (3), it is found that the optimal value of 0 creates an incentive for the algorithm to converge to the optimal solution quickly during the iterative process. Therefore, the algorithm easily falls into the local optimum, and the probability that the local optimum is 0 is larger, which will be called the zero-point search preference. The zero-point preference interference and incentive schematic are shown in Figure 1. There is a zero-point preference corollary as follows.

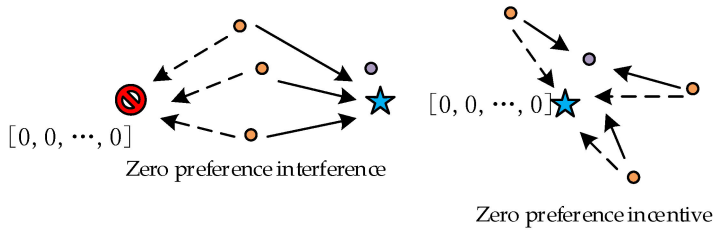


Figure 1. Zero preference interference and incentive.

There exist N feasible solution vectors in each dimension in the search domain $\subset [LB, UB]_D$ defined by the function $F(x)$; then, there is a total of N_D feasible solution vectors in the search domain. If there exist K (K is a constant) solutions of $F(x)$ with the same global optimal fitness and N tends to infinity, the probability that any feasible solution vector in the search domain corresponds to a global optimal fitness solution is as in Equation (17).

$$P\{fitness(X) = \min(fitness)\} = \lim_{N\to\infty} \frac{K}{N^D} \tag{17}$$

If the algorithm has no preference, the search probability for each feasible solution should be of the same order as $P\{fitness(X) = min(fitness)\}$. Preference appears when a feasible solution vector X^p in the search domain is attractive to the population, then there is Equation (18).

$$P\{fitness(X) = \min(fitness)\} = \lim_{N\to\infty} \frac{K}{N^D} \tag{18}$$

Find the fitness probability of the global optimum, and then determine whether there is interference in the search process; if there is no interference, then it is expressed by Equation (19).

$$P\{fitness(X_{best})\} < \lim_{N\to\infty} \frac{K}{N^D} \tag{19}$$

Therefore, it can be judged that the probability of the population converging to X_{best} tends to be a higher order infinitesimal.

3. Improved Algorithm

3.1. Generation of Group Based on Chaotic System

The goodness of the initial population directly affects the solution accuracy and convergence speed of the algorithm, and a well-diversified initial population can largely improve the performance of the algorithm. The problem of using a stochastic system to generate the initial population in optimization problems is that there is a lack of correlation between previous and upcoming states; so, the upcoming states cannot be predicted, resulting in an uneven distribution of the initialized population.

Compared with random systems, chaotic systems are more sensitive to initial values and have the characteristics of mixing properties, randomness, ergodicity and regularity [28–30], which can produce uniform and stable orbits. Given a long enough time, a chaotic system can traverse all states of the space, and the traversal process never repeats in a certain range, so it

can effectively circumvent the local optimum. Lin Z B and Liu Y H [31] proposed the DLCS chaotic mapping by adding a piecewise model to the Lorenz chaotic mapping and introducing a greedy strategy. The DLCS chaotic mapping has extremely complex kinematic properties with outstanding initial value sensitivity, high distribution uniformity and stability, and an enhanced search uniformity of the algorithm for neighborhood solutions. The expressions are shown in Equations (20)–(23), and Z_{sn+1} is the main chaotic sequence generated.

$$X_{n+1} = X_n + a \cdot (X_n - Y_n) \cdot \Delta s \tag{20}$$

$$Y_{n+1} = Y_n + (b \cdot X_n - X_n \cdot Z_n - Y_n) \cdot \Delta s \tag{21}$$

$$Z_{n+1} = Z_n + (X_n^2 + \sin(X_n \cdot Y_n) - c \cdot Z_n) \cdot \Delta s \tag{22}$$

$$Zs_{n+1} = Z_{n+1} \cdot 16 \mathrm{mod} Gear \tag{23}$$

where a = 10, b = 28, c = 6.2 and Δs is the step size taken as 0.01; the signal amplifier control parameter is taken as 16, and Gear = 105 for the sequence slice mode parameter.

The logistic mapping [32], cubic mapping [33], tent mapping [34] and DLCS mapping were used to generate sequences of a length of 1000 each on initial values with small differences and normalized to form a two-dimensional planar distribution in the interval [0, 1] for 30 uniformity comparison experiments; the metrics that constitute the chaotic model are shown in Table 2. When the ratio of both adjacent statistical values of a chaotic distribution graph is closer to 1, the uniformity of the chaotic system is higher. To observe the uniformity of the chaotic orbits generated by the two chaotic systems, two perturbed orbits were generated using the four chaotic systems mentioned above to form a chaotic distribution map, as shown in Figure 2. It is obvious that logistic and cubic have an edge over the density phenomenon, which leads to a weaker ergodicity of the chaotic system. Although most of the segments of the tent map converge normally, there are still uneven distributions. Combined with Table 2, it can be seen that DLCS has the relatively best performance with a stable and uniform distribution, which is suitable for generating the initial chaotic repulsive group of individuals.

Table 2. Planar uniformity comparison.

U	Logistic	Cubic	Tent	DLCS
U_{max}	1.4215	1.3726	1.0259	1.0238
U_{min}	1.3201	1.1836	0.9931	0.9825
U_{avg}	1.3698	1.2145	1.0241	1.0026

3.2. Adaptive Adjustment Strategy for De-Searching Preferences

The fitness in the AO algorithm reflects the difference between the individual and global optimum. In the AO algorithm, it is shown how to quickly choose a search strategy that causes the Aquila to reach the area where the prey is located in the fastest way and thus obtain the prey (optimal solution). The adaptive adjustment strategy of the de-searching preference uses two methods: the adaptive probability threshold method and the adaptive position weight method. First, to further improve the convergence speed of AO while balancing the ability of global exploration and local exploitation, an adaptive probability threshold method is proposed. In this method, the adaptive probability threshold adaptively changes between 0 and 1 with the iterations of the algorithm, guiding individuals to quickly select the hunting strategy more suitable for the current population at different times. The mathematical expression of the adaptive probability threshold *adaptive_p* is shown in Equation (24).

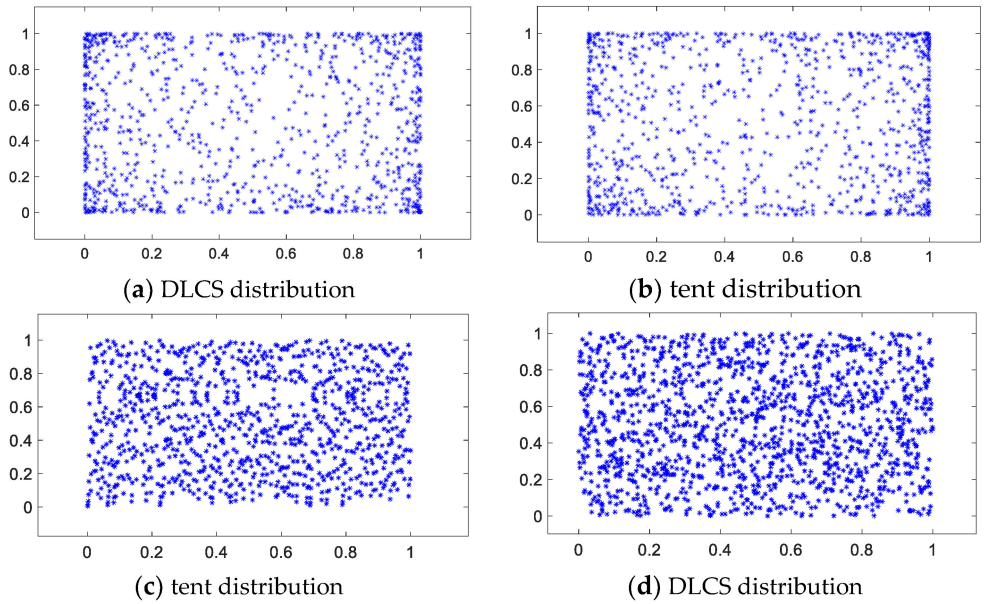


Figure 2. Chaotic System Planar Distribution.

$$adaptive_p = 1 - \left[\frac{1}{1+\lambda} \times \left(\lambda \times \frac{t^{\lambda}}{T^{\lambda}} + \mu \times \frac{t^{\mu}}{T^{\mu}}\right)\right] \tag{24}$$

where λ and μ are control parameters and take the values $\lambda = 3$, $\mu = 2$, and T and t denote the maximum number of iterations and the current number of iterations, respectively.

Therefore, the adaptive probability threshold is introduced into the Gaussian distributed random number *Rand* to further and precisely select the appropriate strategy for hunting. The mathematical expression for the hunting mode of AO can therefore be re-expressed as Equation (25).

$$X(t+1) = \begin{cases} X_{best}(t) \times (1 - \frac{t}{T}) + (X_M(t) - X_{best}(t)) \times rand & ,if rand < adaptive_p \\ X_{best}(t) \times levy(D) + X_R(t) + (y - x) \times rand & \\ (X_{best}(t) - X_M(t)) \times \alpha - rand + ((UB - LB) \times rand + LB) \times \delta & ,if rand \geq adaptive_p \\ QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times levy(D) & \end{cases} \tag{25}$$

At the beginning of the algorithm iteration, the adaptive probability threshold is larger, guiding the Aquila population for global exploration. At the later stage of algorithm iteration, the adaptive probability threshold is smaller to guide the populations for local exploration. This enables the algorithm to adaptively adjust the search strategy and reduce the search preference while effectively improving the convergence speed of the algorithm.

Second, the adaptive weight method is introduced to adjust the position update of AO. The adaptive weight coefficients are shown in Equation (26).

$$\xi = \frac{1}{\tau + \mu} \times \left(\tau \times \frac{t^{\tau}}{T^{\tau}} + \mu \times \frac{t^{\mu}}{T^{\mu}}\right) \tag{26}$$

where $\tau = 2$, $\mu = 2$ and $\xi \in [0, 1]$.

When ξ increases with the number of iterations, it indicates that the prey selected after each iteration is also the optimal solution for the current population with its own adaptation degree, which produces a stronger attraction to the Aquila in the population. At the same time, the credibility of the information conveyed by the randomly selected individuals in the population increases under the iteration, thus making the Aquila able to find the prey more accurately according to the adaptive weight change, thus improving the algorithm convergence speed and the ability to find the best. However, when the local exploitation is carried out in the late iteration of the algorithm, the Aquila approaches the prey; a relatively

small weight coefficient should be used at this time so that the individuals update their position while finely searching whether there is a more optimal solution around the prey, thus improving the local exploitation capability of the algorithm. The introduction of adaptive weight coefficients to update the position is shown in Equations (27)–(30).

$$X_1(t+1) = \xi \times X_{best}(t) \times (1 - \frac{t}{T}) + (X_M(t) - X_{best}(t)) \times rand \quad (27)$$

$$X_2(t+1) = \xi \times X_{best}(t) \times levy(D) + X_R(t) + (y - x) \times rand \quad (28)$$

$$X_3(t+1) = ((1-\xi) \times X_{best}(t) - X_M(t)) \times \alpha - rand + ((UB - LB) \times rand + LB) \times \delta \quad (29)$$

$$X_4(t+1) = QF \times (1-\xi) \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times levy(D) \quad (30)$$

3.3. Communication Strategy Based on Niche Thought

Niche thought comes from biology, where microhabitat refers to the functions or roles of organizations in a particular environment, and organizations with common characteristics are called species. Niche can help maintain diversity in the biological community and help form new species in nature. A schematic diagram of Niche thought is shown in Figure 3.

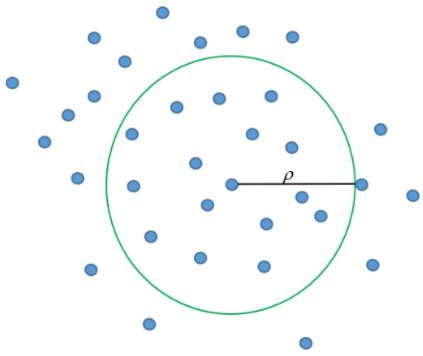


Figure 3. Diagram of Niche Thought.

Niche thought is introduced in the AO algorithm [35], which uses a sharing mechanism to compare the distance between individuals in a habitat. A specific threshold is set to increase the fitness of individuals with high fitness, which is used to ensure that the individual state is at the optimum. For individuals with low fitness, a penalty is given to make them update and further search for optimal values in other regions to ensure the diversity of the population in the iterative process and further obtain the location of the optimal solution. In this process, the distance between the individuals of the small habitat population is calculated by Equation (31).

$$d_{ij} = |X_i - X_j| \quad (31)$$

The information exchange function between individual Xi and Xj is shown in Equation (32).

$$sh(d_{ij}) = \begin{cases} 1 - \frac{d_{ij}}{\rho}, d_{ij} < \rho \\ 0, d_{ij} > \rho \end{cases} \quad (32)$$

where ρ is the radius of information sharing in the microhabitat, and $d_{ij} < \rho$ ensures that individuals survive in the microhabitat environment.

After sharing information, the optimal adaptation is adjusted in time, as shown in Equation (33).

$$F_i_best = \frac{F_i}{sh}, i = 1, 2, \ldots, N \quad (33)$$

where F_i_best is the optimal adaptation after sharing, and F_i is the original adaptation.

3.4. Algorithm General Framework

In summary, the improved modules are reorganized to constitute the new algorithm after the improved AO, which is called the Adaptive Aquila optimizer Combining Niche Thought with Dispersed Chaotic Swarm(NCAAO). The overall flow chart of the algorithm is shown in Figure 4. In NCAAO, a set of candidate solutions is generated by chaotic mapping to initiate the improved algorithm. Through repeated iterations, the search strategy of NCAAO gradually converges toward the optimal solution or obtains the location of the optimal solution. The adaptive adjustment strategy and the communication exchange strategy help the algorithm to obtain the optimal position according to the optimization process, balancing global exploration with local exploitation. Until the algorithm reaches the end criteria, the NCAAO search process is terminated. The Algorithm 1 pseudo-code is as follows.

Algorithm 1. Pseudo-code of NCAAO

Initialization phase:
Initialization of the parameters of the AO (i.e., D, UB, LB, t, T)
Generating initial populations using DCLS chaotic mapping with a length of ChaoticTrackLength_no and a range of $[LB, UB]_D$ X_i = (i = 1,2,...0., N)
while (t < T + 1)
Calculate the fitness function values.
$X_{best}(t)$ = Determine the best obtained solution according to the fitness values.
Update x, y, G_1, G_2, Levy(D), etc.
Update adaptive_p and ξ using Equations (24) and (26)
For i = 1:N^D
if rand<adaptive_p
Update the position of X(t + 1) using Equation (27)
Else
Update the position of X(t + 1) using Equation (28)
End if
Else
if rand>adaptive_p
Update the position of X(t + 1) using Equation (29)
Else
Update the position of X(t + 1) using Equation (30)
End if
Calculate distance from X_i to X_j using Equation (31)
Update the best fitness using Equation (33)
End for
End

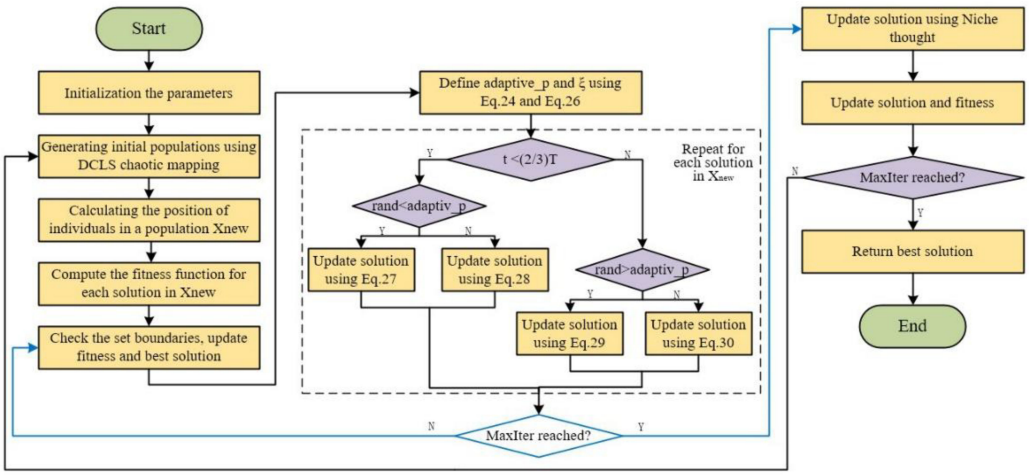


Figure 4. NCAAO algorithm flow chart.

3.5. Time Complexity Analysis

The computational complexity of the improved algorithm is based on the original algorithm by adding chaotic mapping to generate the initial population and updating the fitness of the population through Niche thought. The computational complexity of the algorithm still depends on the initialization of the solution, the calculation of the function and the update of the solution. First, if the number of individuals in the initialized population is N, the dimension of the objective function is D and the maximum number of iterations is T. The time complexity is expressed in terms of "*O*", and the time complexity of solving the initial population of Aquila once per solution is Equation (34).

$$O(\mathrm{T} \times D) \tag{34}$$

The process of updating the solutions includes updating the positions of all solutions as well as exploring the best positions with a time complexity of Equation (35).

$$O(N \times T) + O(N \times T \times D) \tag{35}$$

Through the small habitat idea to ensure the excellence of the population in the iterative process and repeated iterations to compare the mutual distance between the population particles and adjust the optimal adaptation in time, the time complexity is as in Equation (36).

$$O((N-1) \times T \times D) \tag{36}$$

Therefore, the total algorithm time complexity is shown in Equation (37). Since the chaotic orbit length and the number of iterations need to be determined based on the solution problem, the actual complexity of the algorithm is influenced by the actual problem.

$$O(D \times (N \times T)) \tag{37}$$

4. Algorithm Performance Experiments

In this section, the performance of the proposed NCAAO in solving optimization problems is studied. For this purpose, fifteen objective functions of NCAAO were selected from the literature [36,37] for solving different types of unimodal, high-dimensional multimodal and fixed-dimensional multimodal modes. The benchmark functions used are detailed in Table 3. Similarly, three engineering design problems were selected to test the applicability of NCAAO.

Table 3. Unimodal benchmark functions.

Function	Dim	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	10,30,100,500	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	10,30,100,500	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	10,30,100,500	$[-100, 100]$	0
$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	10,30,100,500	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	10,30,100,500	$[-1.28, 1.28]$	0
$F_6 = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	10,30,100,500	$[-5.12, 5.12]$	0
$F_7(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	10,30,100,500	$[-600, 600]$	0

Table 3. *Cont.*

$F_8 = \frac{\pi}{n}\{10\sin(\pi y_1)\}+ \sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})+ \sum_{i=1}^{n}u(x_i,10,100,4)], where \quad y_i = 1+\frac{x_i+1}{4}, u(x_i,a,k,m)\begin{cases} K(x_i-a)^m, x_i > a \\ 0, -a \le x_i \le a \\ K(-x_i-a)^m, x_i < -a \end{cases}$	10,30,100,500	$[-50, 50]$	0
$F_9(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i-1)^2 \times[1+\sin^2(3\pi x_i+1)] + (x_n-1)^2[1+\sin^2(2\pi x_n)]\} + \sum_{i=1}^{n}u(x_i,5,100,4)$	10,30,100,500	$[-50, 50]$	0
$F_{10} = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})}\right)^{-1}$	2	$[-65, 65]$	1
$F_{11}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	$[-5, 5]$	0.00030
$F_{12} = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$F_{13}(x) = [1+(x_1+x_2+1)^2(19-14x_1+3x_1^2 -14x_2+6x_1x_2+3x_2^2)] \times [30+(2x_1-3x_2)^2 \times(18-32x_1+12x_1^2-48x_2+36x_1x_2+27x_2^2)]$	2	$[-2, 2]$	3
$F_{14} = -\sum_{i=1}^{4}c_i\exp(-\sum_{i=1}^{3}a_{ij}(x_j-p_{ij})^2)$	3	$[-1, 2]$	-3.86
$F_{15}(x) = -\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	[0, 1]	-10.5363

In addition, the obtained optimization results from the proposed NCAAO are compared with five well-known optimization algorithms. These competing algorithms include popular methods: Gray Wolf Optimization (GWO) [38], Whale Optimization Algorithm(WOA) [39], Harris Hawks Optimization(HHO) [40], Ant Lion Optimizer(ALO) [41] and Aquila Optimizer(AO) [13]. Table 4 shows the values of the control parameters of these algorithms.

Table 4. Parameter settings.

Algorithm	Setting
GWO	a was linearly decreased from 2 to 0, rate = 3
WOA	α decreased from 2 to 0, b = 1
HHO	t = 0
ALO	I ratio = 10^{ω}, ω = [2, 6]
AO	S = 1.5, r_1 take a fixed index between 1 and 20, G_2 decreased from 2 to 0
MFO	Convergence constant α from 2 to 0, Spiral factor b = 1
PSO	$C_1 = -0.2$, $C_2 = 1.49445$, w = 0.8 $P_c = 0.95$,
GA	$P_c = 0.95$, $P_m = 0.001$, $E_r = 0.2$
TSA	t = 0
SSA	P_percent = 0.2

To evaluate the performance of optimization algorithms, each of the competing algorithms as well as the proposed NCAAO in 30 independent implementations (with each independent implementation containing 500 iterations) have been implemented on the objective functions. The experimental environment is Windows 10, a 64-bit operating system, the CPU is Inter Core i7-10750H, the main frequency is 2.60 GHz and the memory is 16GB. The algorithm is based on MATLAB R2019a. The simulation results are reported using two criteria: the average of the best solutions obtained (AVG) and the standard deviation of the best solutions obtained (STD).

4.1. Benchmark Set and Compared Algorithms

Table 3 includes five unimodal functions (F1–F5) to test the algorithm development capability, four high-dimensional multimodal functions (F6–F9) to test the algorithm search capability as well as the local optimum avoidance capability, and six fixed-dimensional multimodal modes (F10-F15) that are considered to be a combination of the first two sets of random rotations, shifts and biases. The composite test functions are more similar to the real search space, facilitating the simultaneous benchmarking of algorithm exploration and development [42]. The 'w/t/l' in Table 3 indicates the number of wins, draws and failures of each algorithm. The algorithms in Table 3 are shown in Figure 5 in the graph of the effect of adaptation in two dimensions.

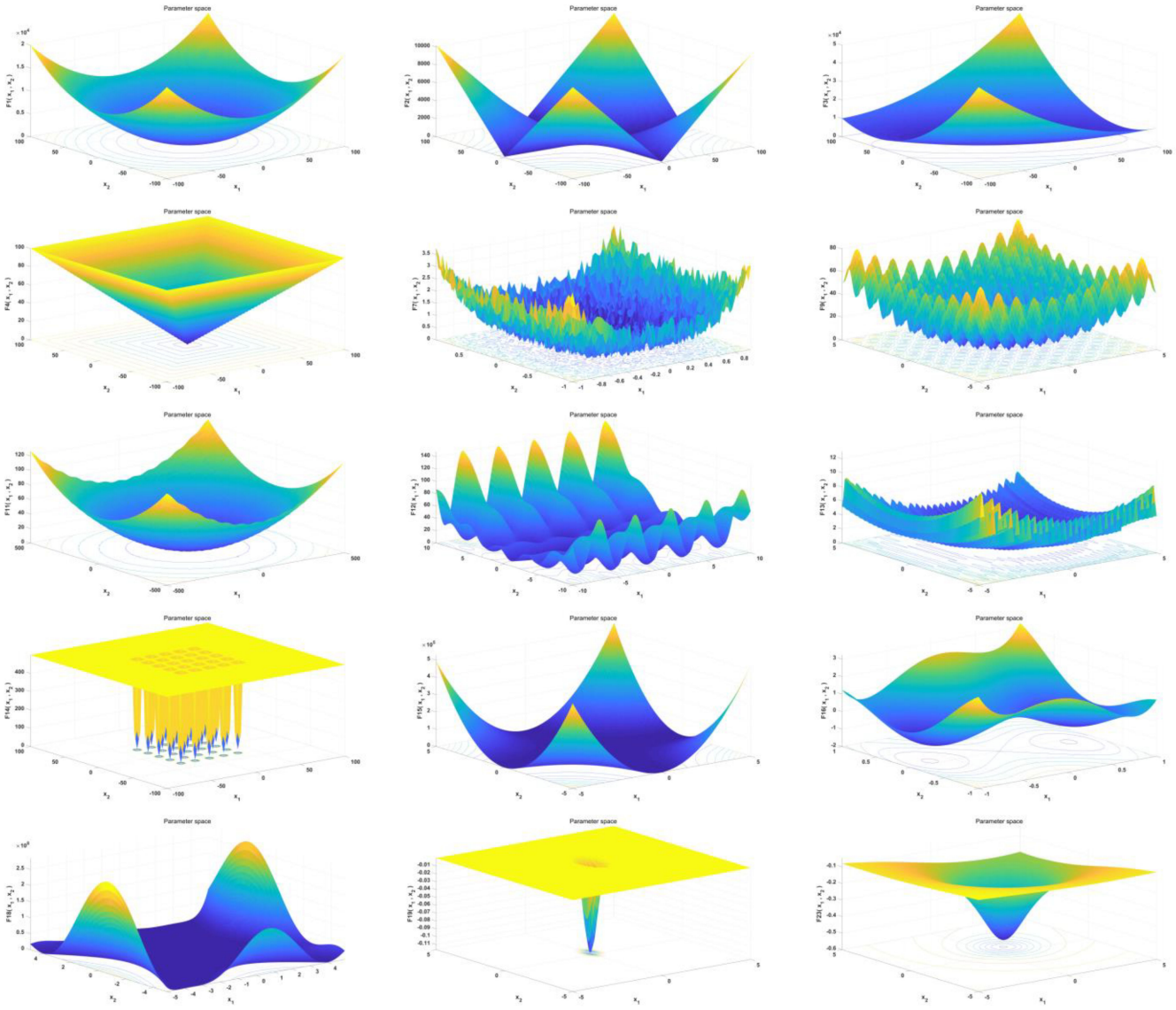


Figure 5. Fitness charts of 15 cases in two dimensions.

The NCAAO algorithm was tested by the unimodal functions, and the results are shown in Table 5. As can be seen from the table, the NCAAO algorithm is able to provide very competitive results on the single-peak test function, especially on the F2 function for all dimensions with significant improvements. The proposed algorithm converges to the global optimum of this function, i.e., zero. This proves that the proposed algorithm has a high utilization capacity, which is due to the role of initializing the population with the

DLCS chaotic map, helping the NCAAO algorithm to provide high utilization. Therefore, the NCAAO algorithm has a strong development capability.

Table 5. The comparison of obtained solutions for unimodal functions.

F	D	Index	GWO	WOA	HHO	ALO	AO	NCAAO
F1	10	AVG	5.6575×10^{-131}	4.8691×10^{-122}	9.5173×10^{-122}	3.2913×10^{-10}	5.6144×10^{-174}	**0**
		STD	1.4541×10^{-130}	1.0843×10^{-121}	2.6643×10^{-121}	1.1628×10^{-10}	0	0
	30	AVG	5.4833×10^{-57}	1.3682×10^{-115}	5.8876×10^{-118}	4.8172×10^{-8}	4.2005×10^{-176}	**0**
		STD	1.3376×10^{-56}	2.0943×10^{-115}	1.2566×10^{-117}	2.0373×10^{-8}	0	0
	100	AVG	1.1543×10^{-25}	2.6843×10^{-114}	3.2317×10^{-115}	3.8028×10^{-2}	1.381×10^{-177}	**0**
		STD	2.5685×10^{-25}	3.8666×10^{-113}	5.6943×10^{-114}	2.6591×10^{-2}	0	0
F2	10	AVG	7.7136×10^{-72}	1.0135×10^{-67}	3.6286×10^{-67}	1.9342×10^{-2}	5.2349×10^{-87}	$\mathbf{5.8837 \times 10^{-169}}$
		STD	1.4544×10^{-71}	1.60484×10^{-67}	7.06325×10^{-67}	4.3151×10^{-2}	1.1556×10^{-86}	0
	30	AVG	3.3720×10^{-33}	4.99×10^{-67}	4.3415×10^{-69}	2.2323	1.0200×10^{-88}	$\mathbf{1.9000 \times 10^{-168}}$
		STD	3.94207×10^{-33}	9.9288×10^{-67}	4.30699×10^{-67}	1.5552	1.86559×10^{-88}	0
	100	AVG	2.7785×10^{-14}	1.2822×10^{-63}	4.1000×10^{-68}	8.0225×10^{2}	1.4752×10^{-88}	$\mathbf{5.8922 \times 10^{-167}}$
		STD	3.5943×10^{-14}	2.4739×10^{-63}	3.5430×10^{-67}	1.1591×10^{2}	1.7100×10^{-88}	0
F3	10	AVG	6.7187×10^{-76}	4.3830×10^{-6}	1.4170×10^{-106}	1.6350×10^{-6}	1.4380×10^{-178}	**0**
		STD	2.3663×10^{-75}	1.36690×10^{-6}	0	7.3425×10^{-6}	0	0
	30	AVG	4.1523×10^{-22}	4.2241×10^{-3}	4.4823×10^{-111}	2.1156×10^{1}	2.3572×10^{-173}	**0**
		STD	1.2303×10^{-22}	5.0500×10^{-3}	0	7.6463×10^{1}	0	0
	100	AVG	1.4000×10^{-3}	4.1766×10^{5}	1.8609×10^{-107}	1.2330×10^{4}	1.3141×10^{-177}	**0**
		STD	5.2473×10^{-2}	2.2046×10^{5}	5.5540×10^{-107}	8.1439×10^{4}	0	0
F4	10	AVG	2.5343×10^{-44}	1.1055×10^{-8}	6.3666×10^{-60}	1.5402×10^{-5}	1.9843×10^{-89}	$\mathbf{7.9443 \times 10^{-167}}$
		STD	3.4655×10^{-44}	2.4773×10^{-8}	7.6522×10^{-60}	3.0500×10^{-6}	2.5100×10^{-89}	0
	30	AVG	8.1471×10^{-15}	4.4538×10^{-14}	4.4883×10^{-59}	3.4741	1.1948×10^{-86}	$\mathbf{2.0741 \times 10^{-167}}$
		STD	7.7544×10^{-15}	8.5664×10^{-14}	9.9273×10^{-59}	3.7564	2.6700×10^{-86}	0
	100	AVG	4.4000×10^{-5}	1.7645×10^{-1}	7.6788×10^{-60}	2.0600×10^{1}	1.1767×10^{-86}	$\mathbf{5.0800 \times 10^{-168}}$
		STD	1.5946×10^{-5}	3.4955×10^{-1}	8.6634×10^{-60}	2.5973	2.3443×10^{-86}	0
F5	10	AVG	2.7156×10^{-4}	2.0000×10^{-2}	1.6343×10^{-5}	3.0100×10^{-3}	9.2146×10^{-5}	$\mathbf{4.3222 \times 10^{-12}}$
		STD	3.0725×10^{-4}	3.5740×10^{-2}	2.0440×10^{-5}	3.3140×10^{-3}	1.3600×10^{-4}	0
	30	AVG	6.8043×10^{-5}	2.6225×10^{-3}	1.7400×10^{-4}	3.8846×10^{-3}	1.5622×10^{-4}	$\mathbf{2.0510 \times 10^{-6}}$
		STD	2.0579×10^{-5}	2.8950×10^{-3}	3.2803×10^{-4}	1.9460×10^{-3}	2.0864×10^{-4}	0
	100	AVG	2.2536×10^{-4}	1.9441×10^{-4}	1.8113×10^{-5}	3.7800×10^{-1}	1.8843×10^{-4}	$\mathbf{5.2014 \times 10^{-6}}$
		STD	1.8141×10^{-4}	3.3940×10^{-4}	1.8235×10^{-5}	3.1397×10^{-1}	3.8878×10^{-4}	1.5420×10^{-6}
RANK	10	w/t/l	0/0/5	0/0/5	0/0/5	0/0/5	0/0/5	5/0/0
	30	w/t/l	0/0/5	0/0/5	0/0/5	0/0/5	0/0/5	5/0/0
	100	w/t/l	0/0/5	0/0/5	0/0/5	0/0/5	0/0/5	5/0/0

After proving the exploitation of NCAAO, we will discuss the exploration of this algorithm. The NCAAO algorithm was tested by high-dimensional multimodal functions, and the results are shown in Table 6. From the table, it can be seen that the NCAAO algorithm is equally competitive with the classical algorithm in finding the optimal values for different dimensions of the F6 and F7 functions. However, the results of finding the optimum for F8 and F9 functions show that the NCAAO algorithm has a stronger search ability and local optimum avoidance ability than other classical algorithms. The excellent exploration of the proposed algorithm is due to the adjustment of adaptive search thresholds and the introduction of adaptive position weighting parameters. Directing a population from one search region into another promotes exploration. This is similar to the crossover operator in GA that highly emphasizes search space exploration.

Table 6. The comparison of obtained solutions for multimodal functions.

F	D	Index	GWO	WOA	HHO	ALO	AO	NCAAO
F6	10	AVG	4.2526	2.9536×10^{-15}	**0**	1.7724×10^{1}	**0**	**0**
		STD	5.9828	6.8724×10^{-15}	0	1.1325×10^{1}	0	0
	30	AVG	2.1300×10^{1}	**0**	**0**	3.7500×10^{2}	**0**	**0**
		STD	1.3500×10^{1}	0	0	6.8300×10^{1}	0	0
	100	AVG	1.4028×10^{1}	**0**	**0**	3.3645×10^{2}	**0**	**0**
		STD	4.1562	0	0	4.0162×10^{1}	0	0

Table 6. *Cont.*

F	D	Index	GWO	WOA	HHO	ALO	AO	NCAAO
F7	10	AVG	4.9425×10^{-3}	4.4737×10^{-2}	**0**	1.7114×10^{-1}	**0**	**0**
		STD	6.7643×10^{-3}	6.2962×10^{-2}	0	1.0108×10^{-1}	0	0
	30	AVG	4.1982×10^{-2}	**0**	**0**	2.2600×10^{-2}	**0**	**0**
		STD	3.6423×10^{-2}	0	0	2.4700×10^{-1}	0	0
	100	AVG	2.7342×10^{-2}	1.3320×10^{-1}	**0**	2.3934×10^{-1}	**0**	**0**
		STD	2.2163×10^{-2}	1.5432×10^{-1}	0	1.5643×10^{-1}	0	0
F8	10	AVG	2.0436×10^{-2}	1.1674×10^{-2}	1.4803×10^{-4}	2.8749×10^{-2}	3.1236×10^{-4}	$\mathbf{1.8510 \times 10^{-5}}$
		STD	8.8876×10^{-3}	8.9764×10^{-2}	5.2234×10^{-4}	1.2356×10^{-3}	4.2360×10^{-5}	3.2664×10^{-5}
	30	AVG	6.1270×10^{-1}	4.7881×10^{-1}	2.0856×10^{-6}	1.5716×10^{1}	1.0767×10^{-6}	$\mathbf{1.2786 \times 10^{-7}}$
		STD	1.5289×10^{-1}	2.3845×10^{-1}	1.1722×10^{-5}	2.1337×10^{1}	1.4585×10^{-6}	3.2361×10^{-7}
	100	AVG	5.2230×10^{-2}	1.3180×10^{-1}	4.2344×10^{-5}	2.4432×10^{1}	7.3494×10^{-6}	$\mathbf{6.7628 \times 10^{-6}}$
		STD	1.3462×10^{-2}	1.5785×10^{-1}	5.4672×10^{-5}	1.3258×10^{1}	8.7032×10^{-6}	1.3640×10^{-6}
F9	10	AVG	4.0664×10^{-2}	7.3654×10^{-2}	1.5774×10^{-4}	5.0367×10^{-3}	9.4682×10^{-5}	$\mathbf{5.4800 \times 10^{-7}}$
		STD	9.0056×10^{-2}	6.3332×10^{-2}	2.5761×10^{-4}	5.9886×10^{-3}	0.3021×10^{-2}	2.6473×10^{-6}
	30	AVG	4.0500	3.7500	1.5736×10^{-4}	9.6700×10^{1}	6.4589×10^{-5}	$\mathbf{1.3468 \times 10^{-6}}$
		STD	4.8600	5.4500×10^{-1}	1.8932×10^{-4}	7.5600×10^{1}	6.6508×10^{-5}	5.6430×10^{-6}
	100	AVG	3.6753×10^{-1}	4.3929×10^{-1}	9.1331×10^{-4}	1.3463×10^{1}	1.8243×10^{-5}	$\mathbf{2.8756 \times 10^{-4}}$
		STD	1.0467×10^{-1}	3.1817×10^{-1}	1.2332×10^{-3}	1.2789×10^{1}	1.6473×10^{-5}	7.6312×10^{-4}
RANK	10	w/t/l	0/0/4	0/0/4	0/2/2	0/0/4	0/2/2	2/2/0
	30	w/t/l	0/0/4	0/1/3	0/2/2	0/0/4	0/2/2	2/2/0
	100	w/t/l	0/0/4	0/0/4	0/2/0	0/0/4	0/2/2	2/2/0

F10–F15 are the six objective functions that evaluate the ability of the optimization algorithm to handle fixed-dimensional multimodal mode problems. The results of optimizing these objective functions using the proposed NCAAO and the competing algorithms are shown in Table 7. The proposed NCAAO optimizing F10, F12, F14, and F15 is able to converge to the global optimum of these functions. NCAAO is the first best optimizer for solving F14 and F15. In optimizing the functions of F10, F11, F12 and F13, although the average criterion of NCAAO is similar to some rival algorithms, its standard criterion is more outstanding. Therefore, the proposed NCAAO solves these objective functions more effectively. The analysis of the simulation results shows that the proposed NCAAO has a higher ability to solve the fixed-dimensional multimodal mode optimization problems from F10 to F15 compared with six competing algorithms.

Table 7. The comparison of obtained solutions for fixed-dimension multimodal functions.

F	D	Index	GWO	WOA	HHO	ALO	AO	NCAAO
F10	2	AVG	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$
		STD	0	0	0	0	0	0
F11	4	AVG	3.1666×10^{-4}	3.2614×10^{-4}	$\mathbf{3.1002 \times 10^{-4}}$	5.6676×10^{-4}	5.1303×10^{-4}	5.4825×10^{-4}
		STD	1.9293×10^{-5}	3.5372×10^{-5}	3.3390×10^{-6}	2.2070×10^{-4}	4.0083×10^{-4}	2.0710×10^{-6}
F12	2	AVG	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
		STD	0	0	0	0	0	0
F13	2	AVG	**3**	**3**	**3**	**3**	**3**	3.0053
		STD	0	0	0	0	0	1.3000×10^{-4}
F14	3	AVG	−3.8623	−3.8596	−3.8347	−3.8213	−3.8425	**−3.8604**
		STD	1.6897×10^{-3}	2.1255×10^{-2}	4.0300×10^{-2}	0.4283×10^{-2}	0.1978×10^{-2}	7.7618×10^{-4}
F15	4	AVG	-1.0543×10^{1}	-2.1105×10^{4}	−7.2953	−7.2943	-2.1134×10^{4}	$\mathbf{-1.0500 \times 10^{1}}$
		STD	1.0832×10^{-4}	4.8980×10^{-5}	2.9620	2.9620	1.0496×10^{-4}	7.4736×10^{-4}
RANK	D	w/t/l	0/3/2	0/3/2	1/3/2	0/3/3	0/3/3	2/2/1

Although the use of AVG and STD indices from the experimental evaluation showed that the NCAAO algorithm outperformed the comparison algorithm, it was not sufficient to demonstrate the superiority of the NCAAO algorithm. Due to the random nature of the metaheuristic, and for a fair comparison, the Wilcoxon rank sum test statistical analysis

was used to draw statistically significant conclusions [43]. Simulation results for testing the proposed NCAAO using five competing algorithms are presented in Table 8. In this table, when the *p*-value is less than 0.05, the proposed NCAAO has a significant advantage over the competing algorithms in this group of objective functions. The *p*-values in the table also prove that the advantage is significant in most cases. This proves the high exploitation capability of the proposed algorithm, which is due to a proposed adaptive adjustment strategy of de-searching preferences that allows the algorithm to balance effectively in development and exploration. The effective combination of small habitat techniques allows the algorithm to speed up the process of preference seeking. In addition, the *p*-value also shows that the NCAAO algorithm effectively avoids the zero-search preference problem, proving that the results of the algorithm are very competitive.

Table 8. The *p*-values obtained from Wilcoxon sum rank test. ($p \geq 0.05$ have been underlined).

Functions Type	GWO and NCAAO	WOA and NCAAO	HHO and NCAAO	ALO and NCAAO	AO and NCAAO
Unimodal	0.000639	0.000248	0.001553	0.000010	0.009578
High-dimensional multimodal	0.028571	0.065714	0.065714	0.028571	0.082857
Fixed-dimensional multimodal	0.011893	0.007800	0.003965	0.007800	0.068436

To evaluate the convergence performance of the NCAAO algorithm, the convergence curves were plotted by selecting the best average value among 30 iterations, as shown in Figure 6. The convergence curves for the unimodal functions, high-dimensional multimodal functions and fixed-dimensional multimodal functions are shown in the figure. The curves show that for the single-peaked and multi-peaked functions, the NCAAO algorithm has a better ability to find the best and converges faster, which further proves that the algorithm has a better ability to develop as well as explore. As for the multimodal functions, the NCAAO algorithm fluctuates more during the initial iterations and relatively less so in the later stages. The significant decrease of the curve during the iterative process shows that the proposed communication exchange strategy based on the idea of small habitats increases the ability to update the optimal adaptation of the population. The overall iterative curve analysis concludes that the NCAAO algorithm has a stronger convergence behavior relative to the comparison algorithm. It is further demonstrated that NCAAO has a better ability to balance exploitation and exploration.

In summary, the NCAAO algorithm has significantly superior performance compared to other more classical optimization algorithms (e.g., GWO, WOA, HHO, ALO, AO). From the data in Tables 5 and 6, we can see that the performance of the optimization search is better on the unimodal functions as well as the high-dimensional multimodal functions, and there is no significant decrease with the increase of the dimensionality. At the same time, through the convergence analysis in Figure 6, the NCAAO algorithm has good convergence overall; although the convergence speed is slower in function F8 compared with other functions, it can still find the optimal solution. It can be seen through the fixed-dimensional multimodal function's test data in Table 7 that the optimal solution can still be found accurately when the optimal solution is not zero, which reduces the existence of zero search preference. Compared with other algorithms, the NCAAO algorithm has a strong balance of development ability and search ability.

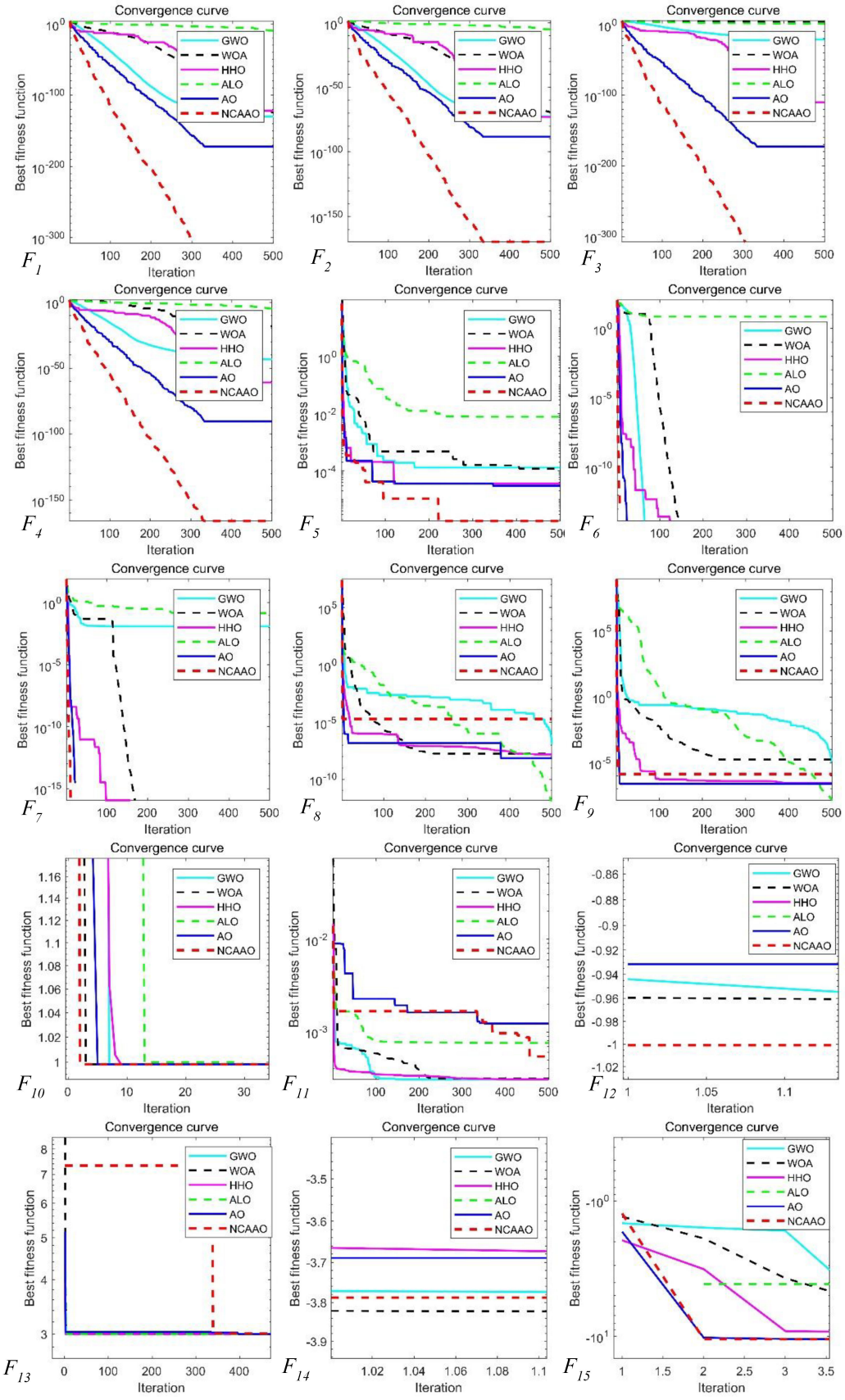


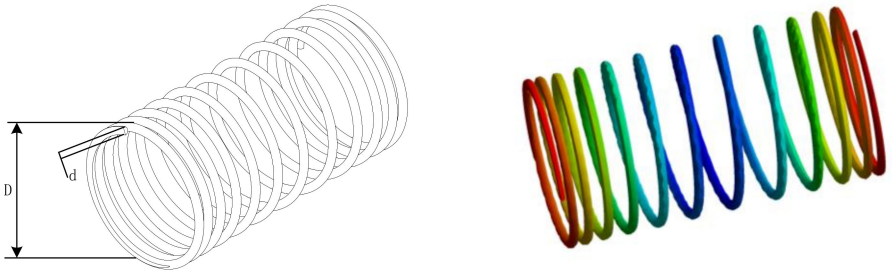
Figure 6. Convergence comparison curves of experiment.

4.2. Engineering Benchmark Sets

In order to further demonstrate that the NCAAO algorithm has better performance in finding the optimum, and to prove the application value of the algorithm in solving engineering problems, the tension/compression spring design problem [44], the pressure vessel design problem [45] and the three-bar truss design problem [46] were selected for solution testing. Salp Swarm Algorithm (SSA) [47], Whale Optimization Algorithm (WOA) [39], Gray Wolf Optimization (GWO) [38], Moth-flame Optimization MFO [48], Gravitational Search Algorithm GSA [49], Particle Swarm Optimization (PSO) [9], Genetic Algorithm (GA) [10], Tunicate Swarm Algorithm (TSA) [50] and NCAAO were also selected for comparison with the NCAAO algorithm for optimization testing of engineering examples. The values of the control parameters for these algorithms are likewise described in Table 4.

4.2.1. Tension/Compression Spring Design Problem

The purpose of the tension/compression spring design problem is to minimize the weight of the pull-pressure spring, which is shown schematically in Figure 7. The main considerations in the design process are the cross-sectional diameter of the spring (d), the average coil diameter (D) and the number of active coils (Q).



(**a**) (**b**)

Figure 7. Schematic diagram of a tension/compression spring. (**a**) Schematic of the spring, (**b**) stress distribution evaluated at the optimum design.

The mathematical model of this problem is as follows:

Consider

$$x = [x_1, x_2, x_3] = [d, D, Q] \tag{38}$$

Minimize

$$f(x) = (x_3 + 2) \cdot x_2 \cdot x_1^2 \tag{39}$$

Subject to

$$\begin{cases} g_1(X) = 1 - \frac{x_2^3 \cdot x_3}{717854 x_1^4} \leq 0 \\ g_2(X) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} \leq 0 \\ g_3(X) = 1 - \frac{140.45 x_1}{x_2^2 \cdot x_3} \leq 0 \\ g_4(X) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \end{cases} \tag{40}$$

Variable range

$$\begin{cases} 0.05 \leq x_1 \leq 2.00 \\ 0.25 \leq x_2 \leq 1.30 \\ 2.00 \leq x_3 \leq 15.00 \end{cases}$$

The NCAAO is applied to this case based on 30 independent runs with 500 group individuals and 500 iterations in each run. Since this benchmark case has some constraints, we need to integrate the NCAAO with a constraint handling technique. The results of NCAAO are compared to those reported for eight algorithms in the previous literature. Table 9 shows the detailed results of the proposed NCAAO compared to other techniques. Based on the results in Table 9, it is observed that NCAAO can reveal very competitive

results compared to the MFO algorithm. Additionally, the NCAAO outperforms other optimizers significantly. The results obtained show that the NCAAO is capable of dealing with a constrained space.

Table 9. Comparison of results for tension/compression spring problem.

Algorithm	Optimum Variables			Optimum Weight
	d	D	N	
SSA [47]	0.051197	0.345219	12.00402	0.0126821
WOA [39]	0.05119	0.357236	12.00309	0.0126828
GWO [38]	0.05171	0.354382	11.28698	0.0125433
MFO [48]	0.051843	0.364107	11.24036	0.0126734
GSA [49]	0.05019	0.353697	14.25410	0.126963
PSO [9]	0.05031	0.310225	14.00324	0.013108
GA [10]	0.05034	0.315979	15.24316	0.012833
TSA [50]	0.050109	0.341599	12.07350	0.012655
NCAAO	0.051836	0.360026	11.13659	0.0126649

4.2.2. Pressure Vessel Design Problem

The pressure vessel design problem is a minimization problem, which is shown schematically in Figure 8. The variables of this case are thickness of shell (*Ts*), thickness of the head (T_h), inner radius (r) and length of the section without the head (L).

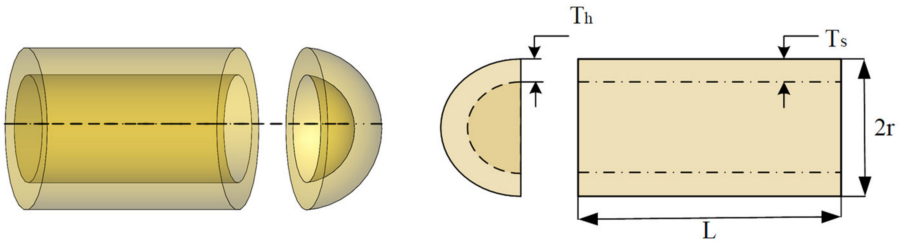


Figure 8. Pressure vessel design problem.

The formulation of this test case is as follows:
Consider

$$\mathrm{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \tag{41}$$

Minimize

$$f(x) = 0.6224x_1x_2x_3 + 1.778x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{42}$$

Subject to

$$\begin{cases} g_1(x) = -x_1 + 0.0193x_3 \le 0 \\ g_2(x) = -x_2 + 0.00954x_3 \le 0 \\ g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\ g_4(x) = x_4 - 240 \le 0 \end{cases} \tag{43}$$

Variable range

$$\begin{cases} 0 \le x_1, x_2 \le 100 \\ 10 \le x_3, x_4 \le 200 \end{cases}$$

Table 10 gives the optimal design results for the pressure vessels. The data in the table shows that NCAAO has given the relatively best results compared with other algorithms. Therefore, NCAAO is proven to be the best optimizer to deal with this problem in this test.

Table 10. Comparison of results for pressure vessel design problem.

Algorithm	Optimum Variables				Optimum Cost
	T_s	T_h	R	L	
SSA [47]	0.812500	0.437516	42.09517	176.9635	6059.7294
WOA [39]	0.783694	0.391106	40.60043	200	5923.497
GWO [38]	0.85257	0.421394	44.23575	175.5324	6142.5731
MFO [48]	0.81150	0.441350	42.10036	176.3492	6058.3024
GSA [49]	1.08996	0.956638	49.73246	183.6672	8766.9234
PSO [9]	0.75536	0.424861	41.65367	179.1068	5919.763
GA [10]	1.095612	0.920064	44.67365	182.5634	6587.639
TSA [50]	0781681	0.386526	40.31256	200	5919.263
NCAAO	0.817624	0.417563	42.98364	177.8329	6000.3776

4.2.3. Three-Bar Truss Design Problem

The three-bar truss design problem is a more classical problem that is used to test the performance of numerous algorithms. The schematic diagram of its structure and the relationship between the forces of each part are shown in Figure 9. The mathematical model of the problem is as follows:

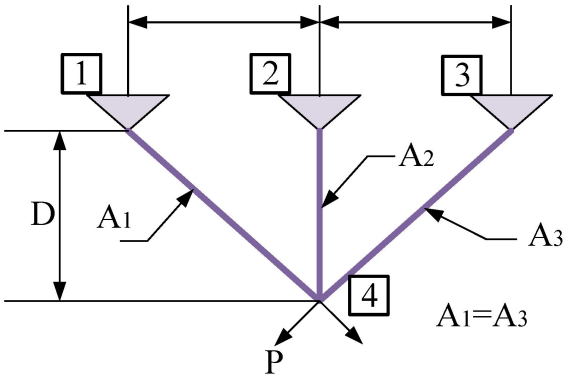


Figure 9. Three-bar truss design problem.

Consider

$$\mathrm{x} = [x_1, x_2] = [A_1, A_2] \tag{44}$$

Minimize

$$f(x) = (2\sqrt{2}x_1 + x_2) \times 1 \tag{45}$$

Subject to

$$\begin{cases} g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \\ g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \\ g_3(x) = \frac{1}{\sqrt{2}x_2 + 2x_1x_2} P - \sigma \leq 0, \end{cases} \tag{46}$$

Variable range

$$0 \leq x_1, x_2 \leq 1$$

where, l = 100 cm, P = 2 KN/cm^2, σ = 2 KN/cm^2.

Table 11 reports the optimum designs attained by NCAAO and the listed optimizers. Inspecting the results in Table 11, we detected that the NCAAO is the best optimizer in dealing with problems and can attain superior results compared to other techniques.

Table 11. Comparison of results for tree-bar truss design problem.

Algorithm	Optimum Variables		Optimum Weight
	x_1	x_2	
SSA [47]	0.791462	0.407936	264.10593
WOA [39]	0.786629	0.406775	263.89534
GWO [38]	0.786513	0.408248	264.01956
MFO [48]	0.787236	0.406624	263.89511
GSA [49]	0.790254	0.407993	263.97544
PSO [9]	0.788522	0.408346	263.89602
GA [10]	0.794366	0.395462	264.00395
TSA [50]	0.789560	0.408011	262.97637
NCAAO	0.788653	0.408294	263.89582

5. Conclusions and Prospect

In this work, a new meta-heuristic algorithm called NCAAO is proposed for the AO algorithm, which is prone to fall into local optima and has the property of zero-point search preference and slow convergence. In fact, three new modules are integrated into the AO algorithm. The first one is the DLCS chaotic mapping chosen as the initial Skyhawk population generator. The resulting population has a more uniform distribution of individuals, which further clarifies the convergence direction of the algorithm effectively. The second module goes to the adaptive adjustment strategy of the search preferences, which is used to balance global search and local probing by changing the threshold of the Aquila selection work mechanism. Then, the adaptive position weight parameter is introduced to give a perturbation to the group individuals, which in turn updates their positions and improves the algorithm's local exploitation capability. Finally, in the third module we propose a communication exchange strategy based on the idea of small habitats to better select the global optimum through the information exchange between individuals and to promote the rapid convergence of population individuals toward the global optimum and reduce the search error. Although chaotic mappings have been used in other algorithms, this work took into consideration the homogeneity of chaotic mappings to produce populations. Similarly, the existence of zero search preferences in the AO algorithm was investigated and the new algorithm was further improved.

The following improvements from the theoretical aspect demonstrate that the NCAAO algorithm shows excellent performance in the unified combination of theory and practice:

- Chaotic mapping has the characteristics of chaotic characteristics, randomness, ergodicity and regularity; the use of DLCS mapping produces a more uniform population distribution, which is conducive to an efficient search of the overall search space and further clarifies the convergence direction of the algorithm.
- Changing the threshold *adaptive_p* of the Aquila selection work and introducing the dynamic adjustment strategy in the development and exploration process help the algorithm to choose a reasonable strategy for the search area to seek the best and enhance the utilization behavior of NCAAO in the iterative process.
- The introduced adaptive location weight parameter ζ, which makes the group individual locations more diverse, further promotes the exploration behavior of NCAAO in the iterative process and has a constructive impact on balancing the exploitation and exploration trends.
- The proposed communication exchange strategy based on the idea of small habitats better promotes the optimization of group adaptation. Among them, a penalty mechanism is given to individuals with lower fitness to promote the NCAAO algorithm to better solve the problems of difficult high-dimensional search, local optimal incentive and unclear convergence direction.

The experiments further investigated the development performance, search performance, and local optimum avoidance performance of NCAAO by fifteen objective functions.

The results show that NCAAO has good optimization performance. Meanwhile, three engineering application examples further verify that the NCAAO algorithm provides a better solution for practical engineering applications. Statistical analysis by the Wilcoxon rank sum test concludes that NCAAO has statistical significance. Based on the comprehensive study, it can be concluded that the proposed algorithm has significant advantages in solving practical problems and thus can provide different areas for other researchers. The results demonstrate that the NCAAO algorithm has a more intelligent balancing mechanism in the development and exploration process, which greatly circumvents the local optimal solutions and converges quickly toward the optimal solution when solving many different types of problems. At the same time, the existence of search preferences is suppressed, and the ability to jump out of the local optimal solution is significantly improved.

In future research, we plan to investigate the application of the proposed algorithm to problems such as image enhancement, feature extraction and camera calibration. In addition, we will further investigate in depth the effect of introducing chaotic mappings to evolve the initial population on the convergence performance, which is essential to enhance the performance of the algorithm.

Author Contributions: Conceptualization, Y.Z. and X.X.; methodology, Y.Z.; validation, N.Z., Y.Z and X.L.; formal analysis, Y.Z.; investigation, W.D.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Y.Z.; visualization, K.Z.; supervision, X.X.; project administration, X.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the 111 Project of China (Grant No. D21009) and the State Key Laboratory Fund Project of China: 4D Millimeter Wave Radar Key Technology Research (Grant No. 20210102).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to Project Confidential.

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

References

1. Teng, Z.-J.; Lv, J.-L.; Guo, L.-W. An improved hybrid grey wolf optimization algorithm. *Soft Comput.* **2019**, *23*, 6617–6631. [CrossRef]
2. Neumann, F.; Witt, C. Bioinspired computation in combinatorial optimization: Algorithms and their computational complexity. In Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation, Amsterdam, The Netherlands, 6–10 July 2013; pp. 567–590.
3. Liu, E.; Yao, X.; Liu, M.; Jin, H. AGV path planning based on improved grey wolf optimization algorithm and its implementation prototype platform. *Comput. Integr. Manuf. Syst.* **2018**, *24*, 2779–2791.
4. Shi, C.; Zeng, Y.; Hou, S. Summary of the application of swarm intelligence algorithms in image segmentation. *Comput. Eng. Appl.* **2021**, *57*, 36–47.
5. Li, S.-Y.; He, Q.; Chen, J. Application of improved equilibrium optimizer algorithm to constrained optimization problems. *J. Front. Comput. Sci. Technol.* **2021**, *9*, 1–14.
6. Zheng, Y.J.; Wang, Y.; Ling, H.F.; Xue, Y.; Chen, S.Y. Integrated civilian–military pre-positioning of emergency supplies: A multi-objective optimization approach. *Appl. Soft Comput.* **2017**, *58*, 732–741. [CrossRef]
7. Zou, S.; Yang, F.; Tang, Y.; Xiao, L.; Zhao, X. Optimized algorithm of sensor node deployment for intelligent agricultural monitoring. *Comput. Electron. Agric.* **2016**, *127*, 76–86.
8. Le Quiniou, M.; Mandel, P.; Monier, L. Optimization of drinking water and sewer hydraulic management: Coupling of a genetic algorithm and two network hydraulic tools. *Procedia Eng.* **2014**, *89*, 710–718. [CrossRef]
9. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 7 November–1 December 1995; pp. 1942–1948.
10. Holland, J.H. Genetic algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
11. Karaboga, D. Artificial bee colony algorithm. *Scholarpedia* **2010**, *5*, 6915. [CrossRef]
12. Forestiero, A. Heuristic recommendation technique in Internet of Things featuring swarm intelligence approach. *Expert Syst. Appl.* **2022**, *187*, 115904. [CrossRef]

13. Abualigah, L.; Yousri, D.; Elaziz, M.A.; Ewees, A.A.; Al-Qaness, M.A.; Gandomi, A.H. Aquila optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
14. Wang, S.; Jia, H.; Abualigah, L.; Liu, Q.; Zheng, R. An improved hybrid aquila optimizer and harris hawks algorithm for solving industrial engineering optimization problems. *Processes* **2021**, *9*, 1551. [CrossRef]
15. Verma, M.; Sreejeth, M.; Singh, M.; Babu, T.S.; Alhelou, H.H. Chaotic Mapping Based Advanced Aquila Optimizer with Single Stage Evolutionary Algorithm. *IEEE Access* **2022**, *10*, 89153–89169. [CrossRef]
16. Akyol, S. A new hybrid method based on Aquila optimizer and tangent search algorithm for global optimization. *J. Ambient. Intell. Humaniz. Comput.* **2022**, 1–21. [CrossRef]
17. Mahajan, S.; Abualigah, L.; Pandit, A.K.; Altalhi, M. Hybrid Aquila optimizer with arithmetic optimization algorithm for global optimization tasks. *Soft Comput.* **2022**, *26*, 4863–4881. [CrossRef]
18. Zhang, Y.-J.; Yan, Y.-X.; Zhao, J.; Gao, Z.-M. AOAAO: The hybrid algorithm of arithmetic optimization algorithm with aquila optimizer. *IEEE Access* **2022**, *10*, 10907–10933. [CrossRef]
19. AlRassas, A.M.; Al-qaness, M.A.; Ewees, A.A.; Ren, S.; Abd Elaziz, M.; Damaševičius, R.; Krilavičius, T. Optimized ANFIS model using Aquila Optimizer for oil production forecasting. *Processes* **2021**, *9*, 1194. [CrossRef]
20. Abd Elaziz, M.; Dahou, A.; Alsaleh, N.A.; Elsheikh, A.H.; Saba, A.I.; Ahmadein, M. Boosting COVID-19 image classification using MobileNetV3 and aquila optimizer algorithm. *Entropy* **2021**, *23*, 1383. [CrossRef]
21. Jnr, E.O.N.; Ziggah, Y.Y.; Rodrigues, M.J.; Relvas, S. A hybrid chaotic-based discrete wavelet transform and Aquila optimisation tuned-artificial neural network approach for wind speed prediction. *Results Eng.* **2022**, *14*, 100399.
22. Ma, L.; Li, J.; Zhao, Y. Population Forecast of China's Rural Community Based on CFANGBM and Improved Aquila Optimizer Algorithm. *Fractal Fract.* **2021**, *5*, 190. [CrossRef]
23. Aribowo, W.; Supari, B.S.; Suprianto, B. Optimization of PID parameters for controlling DC motor based on the aquila optimizer algorithm. *Int. J. Power Electron. Drive Syst. (IJPEDS)* **2022**, *13*, 808–2814. [CrossRef]
24. Ali, M.H.; Salawudeen, A.T.; Kamel, S.; Salau, H.B.; Habil, M.; Shouran, M. Single-and multi-objective modified aquila optimizer for optimal multiple renewable energy resources in distribution network. *Mathematics* **2022**, *10*, 2129. [CrossRef]
25. Yao, J.; Sha, Y.; Chen, Y.; Zhang, G.; Hu, X.; Bai, G.; Liu, J. IHSSAO: An Improved Hybrid Salp Swarm Algorithm and Aquila Optimizer for UAV Path Planning in Complex Terrain. *Appl. Sci.* **2022**, *12*, 5634. [CrossRef]
26. Alkayem, N.F.; Shen, L.; Al-hababi, T.; Qian, X.; Cao, M. Inverse Analysis of Structural Damage Based on the Modal Kinetic and Strain Energies with the Novel Oppositional Unified Particle Swarm Gradient-Based Optimizer. *Appl. Sci.* **2022**, *12*, 11689. [CrossRef]
27. Alkayem, N.F.; Cao, M.; Shen, L.; Fu, R.; Šumarac, D. The combined social engineering particle swarm optimization for real-world engineering problems: A case study of model-based structural health monitoring. *Appl. Soft Comput.* **2022**, *123*, 108919. [CrossRef]
28. Li, J.; Zang, H.; Wei, X. On the construction of one-dimensional discrete chaos theory based on the improved version of Marotto's theorem. *J. Comput. Appl. Math.* **2020**, *380*, 112952. [CrossRef]
29. Zandi-Mehran, N.; Jafari, S.; Golpayegani, S.M.R.H. Signal separation in an aggregation of chaotic signals. *Chaos Solitons Fractals* **2020**, *138*, 109851. [CrossRef]
30. Anand, P.; Arora, S. A novel chaotic selfish herd optimizer for global optimization and feature selection. *Artif. Intell. Rev.* **2020**, *53*, 1441–1486. [CrossRef]
31. Lin, Z.-B.; Liu, Y.-H. Divided chaotic oscillatory annealing TSP optimization algorithm based on greedy strategy. *Appl. Res. Comput.* **2021**, *38*, 2359–2364.
32. Fan, J.-L.; Zhang, X.-F. Piecewise Logistic Chaotic Map and Its Performance Analysis. *Acta Electron. Sin.* **2009**, *37*, 720–725.
33. Wang, C.; Di, Y.; Tang, J.; Shuai, J.; Zhang, Y.; Lu, Q. The Dynamic Analysis of a Novel Reconfigurable Cubic Chaotic Map and Its Application in Finite Field. *Symmetry* **2021**, *13*, 1420. [CrossRef]
34. Zhang, H.; Zhang, T.N.; Shen, J.H.; Li, Y. Research on decision-makings of structure optimization based on improved Tent PSO. *Control Decis.* **2008**, *8*, 857–862.
35. Wang, K.; Mao, W. Simulation of Vertical Temperature Distribution in Green Building Space Based on the Niche Genetic Algorithm. In Proceedings of the 2021 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI), Harbin, China, 24–26 December 2021; pp. 123–130.
36. Yao, X.; Liu, Y.; Lin, G. Evolutionary programming made faster. *IEEE Trans. Evol. Comput.* **1999**, *3*, 82–102.
37. Digalakis, J.G.; Margaritis, K.G. On benchmarking functions for genetic algorithms. *Int. J. Comput. Math.* **2001**, *77*, 481–506. [CrossRef]
38. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
39. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
40. 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]
41. Mirjalili, S. The ant lion optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98. [CrossRef]
42. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S. An improved grey wolf optimizer for solving engineering problems. *Expert Syst. Appl.* **2021**, *166*, 113917. [CrossRef]
43. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]

44. Zhang, M.-J.; Zhang, H.; Chen, X.; Yang, J. A grey wolf optimization algorithm based on Cubic mapping and its application. *Comput. Eng. Sci.* **2021**, *43*, 2035–2042.
45. Kannan, B.; Kramer, S.N. An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. In Proceedings of the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Albuquerque, NM, USA, 19–22 September 1993; pp. 103–112.
46. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper optimization algorithm: Theory and application. *Adv. Eng. Softw.* **2017**, *105*, 30–47. [CrossRef]
47. 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]
48. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
49. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
50. Kaur, S.; Awasthi, L.K.; Sangal, A.; Dhiman, G. Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103541. [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.