Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi
Niğde Ömer Halisdemir University Journal of Engineering Sciences

Araştırma makalesi / Research article

www.dergipark.org.tr/tr/pub/ngumuh / www.dergipark.org.tr/en/pub/ngumuh

An advanced Salp Swarm Algorithm for optimization problems

Optimizasyon problemleri için Gelişmiş Salp Sürüsü Algoritması

Bahaeddin Türkoğlu[1,*]

[1] *Niğde Ömer Halisdemir University, Computer Engineering Department, 51240, Niğde, Türkiye*

Abstract

The Salp Swarm Algorithm (SSA) is a metaheuristic optimization algorithm inspired by Salp swarms' biological characteristics and colony strategies. There is a wide variety of studies conducted with SSA in the literature. These studies have revealed some significant disadvantages of SSA, the most critical being the imbalance of exploration and exploitation. In this study, an equilibrium operator has been developed using the Ikeda map. Thanks to this enhancement, the performance of the SSA algorithm has increased, and issues such as premature convergence and local optima have been overcome. To evaluate the proposed method, ten fixed-dimension benchmark problems and three engineering design optimization problems were solved. The proposed method is validated by comparing four well-known metaheuristic approaches and the original SSA. Experimental results demonstrated that the proposed method outperforms the compared methods.

Keywords: Salp Swarm Algorithm, Engineering design problem, Global optimization

Öz

Salp Sürüsü Algoritması (SSA), Salp sürülerinin biyolojik özelliklerinden ve koloni stratejilerinden ilham alarak geliştirilmiş metasezgisel bir optimizasyon algoritmasıdır. Literatürde SSA ile yapılmış çok çeşitli çalışmalar vardır. Bu çalışmalarda SSA'nın temel dezavantajlarının olduğu vurgulanmıştır. Bunlardan en önemlisi keşif ve sömürü dengesizliğidir. Bu çalışmada Ikeda haritası kullanılarak bir denge operatörü geliştirilmiştir. Bu geliştirme sayesinde SSA algoritmasının performansı artırılarak erken yakınsama ve lokal minimumlara takılma sorunu giderilmeye çalışılmıştır. Önerilen yöntemin başarısını değerlendirmek için on sabit boyutlu benchmark seti ve üç iyi bilinen mühendislik optimizasyon problemi çözülmüştür. Geliştirilen yöntemin güvenilirliği dört iyi bilinen metasezgisel yaklaşımla ve orijinal SSA ile kıyaslanarak doğrulanmıştır. Deneysel çalışma sonuçları, önerilen yöntemin kıyaslanan yöntemlerden daha performanslı olduğunu göstermiştir.

Anahtar kelimeler: Salp Sürüsü Algoritması, Mühendislik tasarım problemleri, Global optimizasyon

1 Introduction

Optimization is the process of determining the most suitable solution among the possible solutions, considering given constraints. In today's world, optimization is widely applied in various domains aiming for maximum efficiency with minimum cost [1]. Many real-world problems such as vehicle and flight route planning [2], traveling salesman problems [3, 4], economic load dispatch [5], wind turbine [6] and facility layout problems [7], land consolidation [8], energy forecasting analysis [9], and engineering design problems, can be formulated as optimization problems [10-12].

The literature introduces various strategies to solve optimization problems, and one prominent approach is metaheuristic methods. Metaheuristic methods initiate the search process with random initial solutions and utilize two fundamental search behaviors: exploration and exploitation, aiming to find the optimal solution. Exploration represents the ability of a method to search the solution space, while the exploitation mechanism refers to the capacity to improve a solution. These two processes are crucial for metaheuristic methods and must be carefully designed to strike an ideal balance [13].

In metaheuristic approaches, especially in the last decade, nature-inspired optimization methods have been developed and become popular. Swarm intelligence optimization is an artificial intelligence optimization technique that models the life behaviors of swarms, where individuals in the swarm, such as cheetahs, vultures, snakes, gorillas, and fruit flies, interact by sharing information. The popularity of swarm intelligence optimization algorithms stems from their effective performance in solving complex real-world problems. One of the key factors contributing to their successful performance is the utilization of collective location update mechanisms and processes inspired by natural events and living organisms in nature. These mechanisms aid in exploring the solution space and improving existing solutions [14].

In the literature, several nature-inspired metaheuristic optimization algorithms have been introduced, such as the cheetah optimizer (CO) [15], elephant clan optimization (ECO) [16], artificial gorilla troops optimizer (GTO) [17], snake optimizer (SO) [18], African vultures optimization algorithm (AVOA) [19], remora optimization algorithm (ROA) [20], artificial hummingbird algorithm (AHA) [21], white shark optimizer (WSO) [22], marine predators

* Sorumlu yazar / Corresponding author, e-posta / e-mail: bturkoglu@ktun.edu.tr (B. Türkoğlu)

Geliş / Recieved: 26.10.2022 Kabul / Accepted: 29.09.2023 Yayımlanma / Published: 15.10.2023
doi: 10.28948/ngumuh.1195013

algorithm (MPA) [23], orca predation algorithms (OPA) [24], and Salp swarm algorithm (SSA) [25].

One of the main motivations of this study is the "No Free Lunch" (NFL) theorem, frequently mentioned in optimization problems. Although numerous optimization techniques are available in the literature, new algorithms are constantly being introduced, and existing algorithms require further development. This is due to the NFL theorem, which states that it is impossible for a single optimization algorithm to universally solve all optimization problems [26, 27].

This paper introduced the advanced SSA algorithm using the equilibrium operator with the Ikeda chaotic map.

The continuation of the study is organized as follows.

In the second part (Materials and Methods), the original SSA and the advanced SSA algorithms are explained and detailed. The experimental setup is presented in the third part of the study. Additionally, the experimental results are analyzed in this section. In the fourth part (Conclusion), the importance of the proposed SSA is discussed. Furthermore, this part presents conclusions and recommendations for future work.

2 Material and methods

This section will present the original SSA method and our proposed approach, the advanced SSA. Additionally, we will provide details about the experimental setup employed in our study.

2.1 *Original Salp Swarm Algorithm*

Salps are jelly-like sea creatures that belong to the Salpidae family. Although their bodies resemble jellyfish, their movement patterns are quite similar. Salps exhibit swarm behavior, and one of their most fascinating behaviors is forming long chains of interconnected individuals, in the depths of the ocean, as illustrated in Figure 1.

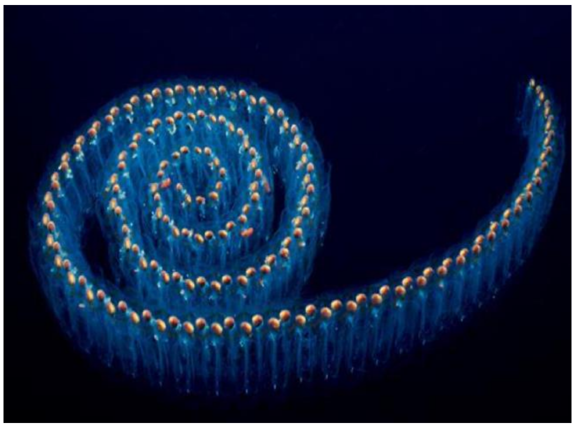

Figure 1. Figures of salp chain [28]

The Salp Algorithm comprises two types of salps: leader and follower salps. The leader salp takes the lead while the follower salps, and trails behind. Salps exhibit a specific behavior called salp chain, which is used for foraging [25]. Figure 2 provides a representative illustration of this behavior.

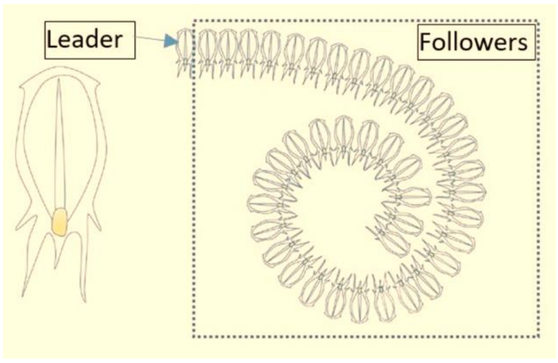

Figure 2. Illustration of salp behavior

The position update equation of the salps is shown in Equations (1).

$$x_j^i = \begin{cases} F_j + c_1((ub_j - lb_j) * c_2 + lb_j), & c_3 \geq 0.5 \\ F_j - c_1((ub_j - lb_j) * c_2 + lb_j), & c_3 < 0.5 \end{cases} \quad (1)$$

In the equation provided:

- x_j^i represents the position of the leader salp in the jth dimension.
- F_j represents the position of the food source in the jth dimension.
- c_1, c_2, and c_3 are randomly generated variables evenly distributed in the range [0,1].
- ub_j and lb_j represent the upper and lower limits in the jth dimension.

The algorithm described is considered a metaheuristic algorithm that exhibits early convergence. Various hybrid versions have been developed in the literature to address this issue. These hybrid versions have been applied to solve various real-world problems across different domains [29].

2.2 *Advanced Salp Swarm Algorithm*

Salps are transparent fish species measuring 1-10 cm in length. These organisms feed on plankton, earning them the nickname "ghost fish." The Salp Swarm Algorithm is a metaheuristic optimization algorithm inspired by the foraging and feeding behaviors of these creatures in the ocean. Three critical parameters affect the SSA's performance: c1, c2, and c3. The c1 and c2 parameters significantly impact the position update, while the c3 parameter is responsible for the strategy used to update the next position of the Salp. [25].

The randomly generated c3 parameter decides how to update the Salp's location. In this study, instead of allowing the c3 parameter to decrease randomly, it is regulated using Ikeda chaotic mapping. The value produced by the chaotic map determines which location update mechanisms will be activated. The Ikeda chaotic map is depicted in Figure 3.

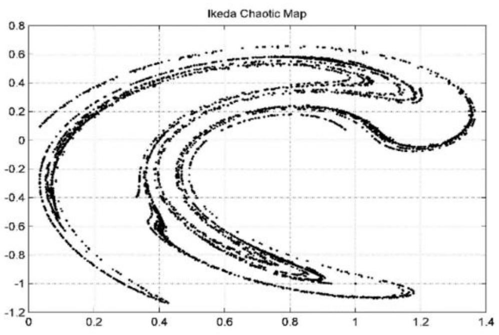

Figure 3. Image of Ikeda map [30]

The formulation of the Ikeda chaotic map is shown in Equations (2).

$$x_{k+1} = 1 + U(x_k cost_k - y_k sint_k), U = 0.8$$
$$y_{k+1} = U(x_k cost_k + y_k sint_k) \qquad (2)$$
$$t_k = 0.4 - 6/(1 + x_k^2 + y_k^2)y = mx + n$$

The new position update equation of the salps, incorporating the values generated by the Ikeda chaotic map (***Ikeda maps***), is shown in Equations (3).

$$if \;\; Ikeda\, map(chaotic\ value) < \; 0.5 \;\; then$$
$$F_j - c_1\left(\left(ub_j - lb_j\right) * c_2 + lb_j\right)$$
else $\qquad (3)$
$$F_j + c_1\left(\left(ub_j - lb_j\right) * c_2 + lb_j\right)$$
$$end\ if$$

3 Result and discussion

This section will first explain the benchmark test functions and real-world engineering design optimization problems. Then, the experimental study environment and experimental results will be detailed.

3.1 *Experimental setup*

To demonstrate the reliability of the algorithms developed in the literature, testing them on optimization problems with diverse characteristics is essential. Classical benchmark function sets and real-world engineering optimization problems are widely accepted in the literature for evaluating global optimization problems [21, 23, 25, 31-34]. This section is divided into two subsections. The first subsection provides a detailed explanation of the classic benchmark test functions. The second subsection focuses on real-world engineering problems solved using the developed method.

3.1.1 *Benchmark functions*

Ten popular benchmark function sets of different difficulty levels were used to test and validate the developed approach. These functions possess a fixed dimension and exhibit multimodal characteristics, meaning they contain multiple local minimums and a single global minimum. Including multimodal functions allows for evaluating the algorithm's capability to escape local minimums and assess its convergence rate. A representation of these benchmark test functions can be found in Table 1.

Table 1. Description of benchmark functions

No	Equation
F1	$F_1(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$
F2	$F_2(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_i)}{b_i^2 + b_i x_3 x_4}\right]^2$
F3	$F_3(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$
F4	$F_4(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$
F5	$F_5(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $* [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$
F6	$F_6(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$
F7	$F_7(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$
F8	$F_8(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$
F9	$F_9(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$
F10	$F_{10}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$

These function properties are shown in Table 2.

Table 2. Properties of fixed-dimension benchmark functions

No	Name	Min	Range	Dim
F1	Foxholes	1	[-65,65]	2
F2	Kowalik	0.00030	[-5,5]	4
F3	Six Hump Camel	−1.0316	[-5,5]	2
F4	Branin	0.398	[-5,5]	2
F5	GoldStein-Price	3	[-2,2]	2
F6	Hartman 3	−3.86	[1,3]	3
F7	Hartman 6	−3.32	[0,1]	6
F8	Shekel 5	−10.1532	[0,10]	4
F9	Shekel 7	−10.4028	[0,10]	4
F10	Shekel 10	−10.5363	[0,10]	4

3.1.2 Engineering design problems

To verify the success of the developed method, besides the classical benchmark functions, three engineering design problems are used.

The most commonly used engineering design problems in the literature are the pressure vessel design problem, the welded beam design problem, and the tension/compression spring design. These problems were used in this study.

3.1.3 Pressure vessel design optimization problem

The main purpose of this problem is to determine the most suitable design parameters that will keep the total cost of a compressed cylindrical air tank to a minimum to create a given volume and a constant working pressure and meet the design constraints. As shown in Figure 4, the hood is hemispherical when both sides of the tank are closed [35].

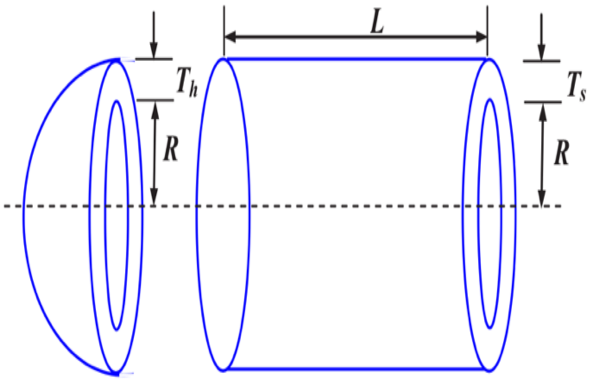

Figure 4. Schema of the pressure vessel design [23]

As shown in the figure, four design variables need to be optimized. The mathematical constraints of this problem are as in Equation (4).

$$
\begin{aligned}
&\textit{Variables} \quad \vec{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L]\\
&\quad \textit{the inner radius } (\boldsymbol{R}),\\
&\quad \textit{the thickness of the cover } (\boldsymbol{T_h}),\\
&\quad \textit{the shell thickness of the body } (\boldsymbol{T_s}),\\
&\quad \textit{the length of the cylindrical part excluding the cover } (\boldsymbol{L})\\
&\textit{Objective Functions} \quad f(\vec{x})\\
&\quad = 0.6224x_1x_3x_4 + 0.7781x_2x_3^2\\
&\quad + 3.1661x_1^2x_4 + 19.84x_1^2x_3\\
&\textit{Constraints} \quad g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0, \qquad (4)\\
&g_2(\vec{x}) = -x_3 + 0.00954x_3 \le 0,\\
&g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0,\\
&g_4(\vec{x}) = x_4 - 240 \le 0,\\
&\textit{Variable Ranges} \quad 0 \le x_1 \le 99, 0 \le x_2 \le 99,\\
&\quad 10 \le x_3 \le 200, 10 \le x_4 \le 200
\end{aligned}
$$

The objective function represents the manufacturing cost of the pressure vessel, and the smaller it is, the more efficient it is.

3.1.4 Welded beam design optimization problem

The main purpose of this problem is to find the optimum design so that the production of the welded beam is the least costly under the given constraints. As shown in Figure 5, the minimum production cost can be calculated by determining the optimum value of the four parameters [36].

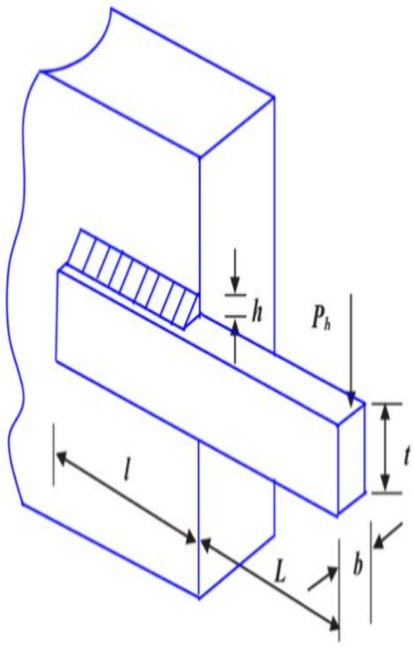

Figure 5. Schema of the welded beam design [23]

The given variables must be determined to satisfy the seven constraints. The mathematical expression of the welded beam problem is shown in Equation (5).

$$\begin{aligned}
&\textit{Variables} \quad \vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b] \\
&\qquad \textit{the length of the connected part of the bar } (\boldsymbol{l}) \\
&\qquad \textit{the thickness of the weld } (\boldsymbol{h}) \\
&\qquad \textit{the height of the bar } (\boldsymbol{t}) \\
&\qquad \textit{the thickness of the bar } (\boldsymbol{b}) \\
&\textit{Objective Functions} \quad f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \\
&\textit{Constraints} \quad g_1(\vec{x}) = \tau(\vec{x}) - \tau_{max} \le 0, \\
&g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{max} \le 0 \\
&g_3(\vec{x}) = \delta(\vec{x}) - \delta_{max} \le 0 \\
&g_4(\vec{x}) = x_1 - x_4 \le 0, \\
&g_5(\vec{x}) = P - P_c(\vec{x}) \le 0, \\
&g_6(\vec{x}) = 0.125 - x_1 \le 0, \\
&g_7(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \le 0, \\
&\textit{Variable Ranges} \quad 0.1 \le x_1 \le 2, \quad 0.1 \le x_2 \le 10, \quad 0.1 \le x_3 \le 10, \; 0.1 \le x_4 \le 2
\end{aligned} \tag{5}$$

3.1.5 Tension & compression spring design optimization problem

The main objective of this problem is to determine the three design parameters to establish the minimum weight of the tension spring to meet the specified design constraints [37]. An exemplary tension spring design is shown in Figure 6.

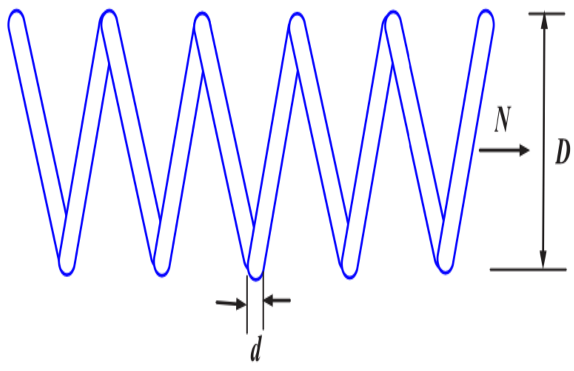

Figure 6. Schema of the compression spring design [23]

The mathematical expression of this problem is shown in Equation (6).

3.2 Experimental results

In this section, the performance of the advanced SSA algorithm is assessed by solving various optimization problems and comparing the results with popular algorithms from the literature under the same conditions. This section is divided into two parts. Firstly, the effectiveness of the proposed method is evaluated through ten fixed-dimensional optimization benchmark tests. In the second part, the performance of the developed algorithm and the compared methods are analyzed in real-world design problems.

$$\begin{aligned}
&\textbf{Variables} \quad \vec{x} = [x_1, x_2, x_3] = [d, D, N] \\
&\qquad \textit{the wire diameter } (\boldsymbol{d}), \\
&\qquad \textit{the average turn diameter } (\boldsymbol{D}) \\
&\qquad \textit{the number of active turns } (\boldsymbol{N}). \\
&\textit{Objective Functions} \quad f(\vec{x}) = (x_3 + 2)x_2x_1^2 \\
&\textit{Constraints} \quad g_1(\vec{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0, \\
&g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} \le 0, \\
&g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\
&g_4(\vec{x}) = 1 - \frac{x_1 + x_2}{1.5} - 1 \le 0, \\
&\textit{Variable Ranges} \quad 0.05 \le x_1 \le 2.00, \quad 0.25 \le x_2 \le 1.30, \quad 2.00 \le x_3 \le 15.0,
\end{aligned} \tag{6}$$

To verify the performance of the developed algorithm, five well-known algorithms, namely Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA), Moth-Flame Optimization (MFO), Moth-Flame Optimization (MFO), and Salp Swarm Algorithm (SSA), were used.
The parameter values of the compared algorithms used in the experimental study are presented in Table 3. These parameter values correspond to the recommendations the respective algorithm authors provided in their own publications.

Table 3. Parameter values of the compared algorithms

Algorithms	Parameters	Values
GWO	alpha	2
WOA	alpha	2
MVO	wep$_{max}$	1
	wep$_{min}$	0.2
MFO	a (linearly decreases)	[-1, -2]
SSA (Original and Advanced)	Probability of crossover	0.8
	Probability of mutation	0.01

Thirty independent experiments were conducted to ensure a fair study, and the results were averaged. The number of populations used in each experiment was set to 40. A total of 50,000 fitness calculations were performed.

.

Table 4. Results of algorithms in benchmark test functions

No	Name	Evolution criteria	GWO	WOA	MVO	MFO	SSA	Advanced SSA
F1	Foxholes	mean std	4.32E+00 4.35E+00	2.34E+00 2.93E+00	9.98E-01 1.31E-11	1.85E+00 1.61E+00	**9.98E-01** 2.39E-16	**9.98E-01** 1.13E-16
F2	Kowalik	mean std	2.35E-03 6.11E-03	7.66E-04 4.21E-04	5.91E-03 1.21E-02	2.63E-03 5.01E-03	8.71E-04 2.42E-04	**4.33E-04** 1.09E-04
F3	Six Hump Camel	mean std	-1.03E+00 8.00E-09	-1.03E+00 2.15E-11	-1.03E+00 2.18E-07	-1.03E+00 6.78E-16	-1.03E+00 5.60E-15	**-1.0316** 5.68E-16
F4	Branin	mean std	**3.98E-01** 2.51E-07	**3.98E-01** 1.14E-06	**3.98E-01** 5.25E-08	3.97E-01 0.00E+00	**3.98E-01** 5.21E-15	**3.98E-01** 0.00E+00
F5	GoldStein-Price	mean std	**3.00E+00** 6.11E-06	**3.00E+00** 8.65E-06	**3.00E+00** 9.88E-07	**3.00E+00** 1.61E-15	**3.00E+00** 1.08E-13	**3.00E+00** 1.35E-15
F6	Hartman 3	mean std	-3.00E-01 2.26E-16	-3.00E-01 2.26E-16	-3.00E-01 2.26E-16	-3.00E-01 2.26E-16	-3.00E-01 2.26E-16	**-3.34E+00** 4.33E-01
F7	Hartman 6	mean std	-3.24E+00 7.87E-02	-3.26E+00 9.14E-02	-3.27E+00 6.04E-02	-3.22E+00 4.82E-02	-3.22E+00 4.22E-02	**-3.32E+00** 1.38E-08
F8	Shekel 5	mean std	-8.97E+00 2.18E+00	-9.73E+00 1.63E+00	-7.96E+00 2.80E+00	-5.98E+00 3.37E+00	-7.82E+00 3.41E+00	**-1.02E+01** 4.36E-10
F9	Shekel 7	mean std	-1.00E+01 1.34E+00	-8.72E+00 2.63E+00	-8.38E+00 2.73E+00	-7.93E+00 3.36E+00	-8.42E+00 3.15E+00	**-1.04E+01** 1.69E-10
F10	Shekel 10	mean std	-1.02E+01 1.58E+00	-7.72E+00 3.29E+00	**-9.10E+00** 2.42E+00	-8.58E+00 3.33E+00	-7.96E+00 3.51E+00	-7.05E+00 1.83E-09

3.2.1 Performance of advanced SSA in benchmark problems

The experimental study results for ten well-known benchmark functions conducted using five popular algorithms (GWO, WOA, MVO, MFO, original SSA, and advanced SSA) are presented in Table 4.

All benchmark test functions were executed in 30 dimensions

According to the results in Table 4, the advanced SSA algorithm demonstrated superior performance compared to the other algorithms, including the original SSA, in nine out of the ten benchmark test functions, indicating a success rate of 90%. These results highlight that by utilizing the Ikeda chaotic map, the proposed method achieves a favorable balance between exploration and exploitation. Following the Advanced SSA, the MFO algorithm emerged as the second most successful.

3.2.2 Performance of advanced SSA in real-world engineering design optimization problems

In solving the engineering design optimization problems, the experimental work employed the same parameters as those used for the benchmark functions. Likewise, thirty independent runs were conducted and averaged. The maximum number of fitness evaluations (FEs) was 50,000, while the population was 40 individuals [38-41]. The results were compared with the performance of GWO, WOA, MVO, MFO algorithms, and the original SSA. The results of the experimental study are presented in Table 5.

Table 5. Results of algorithms in real-world engineering design optimization problems

No	Problem	Evolution criteria	GWO	WOA	MVO	MFO	SSA	Advanced SSA
P1	Welded Beam	mean std	1.74E+00 1.53E-02	2.33E+00 4.57E-01	1.77E+00 3.16E-02	1.72E+00 8.34E-02	1.77E+00 9.88E-02	**-3.28E+05** 2.45E+04
P2	Compression Spring	mean std	3.67E+00 2.63E-03	3.69E+00 2.93E-02	3.71E+00 3.28E-02	3.67E+00 1.98E-02	3.68E+00 2.08E-02	**-9.73E+06** 3.29E+06
P3	Pressure Vessel	mean std	2.68E+03 1.14E+03	4.32E+03 1.81E+03	2.94E+03 6.12E+02	2.39E+03 3.35E+02	3.20E+03 6.28E+02	**-2.37E+11** 3.44E+10

Based on the results presented in Table 5, it can be observed that the proposed method, advanced SSA, outperformed the compared methods in all engineering optimization problems. These findings confirm the superior performance and reliability of the proposed method.

4 Conclusion

The primary objective of this study was to enhance the balance between exploration and exploitation in the SSA algorithm by incorporating the Ikeda chaotic operator. The performance of the developed method was evaluated by applying it to ten different difficulty levels of fixed-dimension multimodal benchmark functions. To validate the results, a comparison was conducted with five well-known methods, namely GWO, WOA, MVO, MFO, and the original SSA. The experimental findings indicated that the proposed method outperformed these comparison methods regarding performance.

This study demonstrated that incorporating the Ikeda chaotic map into the r3 parameter of the SSA algorithm yielded efficient and reliable results. In future studies, it would be worthwhile to investigate the impact of different operators on the vital parameters of the SSA algorithm, namely r1 and r2. Exploring the performances of alternative operators concerning these parameters could provide valuable insights and further enhance the optimization capabilities of the SSA algorithm.

The results of the experimental study are quite competitive in terms of standard deviation values. The standard deviation values of all algorithms are very close to zero. It shows that these algorithms work stably, and their values are consistent across 30 independent studies.

Conflict of interest

The authors declare that there is no conflict of interest.

Similarity rate (iThenticate): 19%

References

[1] M. A. Şahman and S. Korkmaz, Discrete Artificial Algae Algorithm for solving Job-Shop Scheduling Problems. Knowledge-Based Systems, 256, 109711, 2022. https://doi.org/10.1016/j.knosys.2022.109711.

[2] A. C. Cinar, Training feed-forward multi-layer perceptron artificial neural networks with a tree-seed algorithm. Arabian Journal for Science and Engineering, 45 (12), 10915-10938, 2020. https://doi.org/10.1007/s13369-020-04872-1.

[3] M. Gündüz, M. S. Kiran, and E. Özceylan, A hierarchic approach based on swarm intelligence to solve the traveling salesman problem. Turkish Journal of Electrical Engineering & Computer Sciences, 23 (1), 103-117, 2015. https://doi.org/10.3906/elk-1210-147.

[4] A. C. Cinar, S. Korkmaz, and M. S. Kiran, A discrete tree-seed algorithm for solving symmetric traveling salesman problem. Engineering Science and Technology, an International Journal, 23 (4), 879-890, 2020. https://doi.org/10.1016/j.jestch.2019.11.005.

[5] M. Kumar and J. S. Dhillon, Hybrid artificial algae algorithm for economic load dispatch. Applied Soft Computing, 71, 89-109, 2018. https://doi.org/10.1016/j.asoc.2018.06.035.

[6] M. Beşkirli, İ. Koç, H. Haklı, and H. Kodaz, A new optimization algorithm for solving wind turbine placement problem: Binary artificial algae algorithm. Renewable energy, 121, 301-308, 2017. https://doi.org/10.1016/j.renene.2017.12.087.

[7] E. Kaya, BinGSO: galactic swarm optimization powered by binary artificial algae algorithm for solving uncapacitated facility location problems. Neural Computing and Applications, 1-20, 2022. https://doi.org/10.1007/s00521-022-07058-y.

[8] S. Ozsari, H. Uguz, and H. Hakli, Implementation of meta-heuristic optimization algorithms for interview problem in land consolidation: A case study in Konya/Turkey. Land Use Policy, 108, 105511, 2021, https://doi.org/10.1016/j.landusepol.2021.105511.

[9] A. C. Cinar and N. Natarajan, An artificial neural network optimized by grey wolf optimizer for prediction of hourly wind speed in Tamil Nadu, India. Intelligent Systems with Applications, 200138, 2022. https://doi.org/10.1016/j.iswa.2022.200138.

[10] B. Turkoglu and E. Kaya, Training multi-layer perceptron with artificial algae algorithm. Engineering Science and Technology, an International Journal, 2020. https://doi.org/10.1016/j.jestch.2020.07.001.

[11] B. Turkoglu, S. A. Uymaz, and E. Kaya, Clustering analysis through artificial algae algorithm. International Journal of Machine Learning and Cybernetics, 13 (4), 1179-1196, 2022. https://doi.org/10.1016/j.asoc.2022.108630.

[12] B. Turkoglu, S. A. Uymaz, and E. Kaya, Binary Artificial Algae Algorithm for feature selection. Applied Soft Computing, 120, 108630, 2022. https://doi.org/10.1007/s13042-022-01518-6.

[13] E. Kaya, S. Korkmaz, M. A. Sahman, and A. C. Cinar, DEBOHID: A differential evolution based oversampling approach for highly imbalanced datasets. Expert Systems with Applications, 169, 2021. https://doi.org/10.1016/j.eswa.2020.114482.

[14] S. A. Uymaz, G. Tezel, and E. Yel, Artificial algae algorithm (AAA) for nonlinear global optimization. Applied Soft Computing, 31, 153-171, 2015. https://doi.org/10.1016/j.asoc.2015.03.003.

[15] M. A. Akbari, M. Zare, R. Azizipanah-Abarghooee, S. Mirjalili, and M. Deriche, The cheetah optimizer: A nature-inspired metaheuristic algorithm for large-scale optimization problems. Scientific reports, 12 (1), 1-20, 2022. https://doi.org/10.1038/s41598-022-14338-z.

[16] M. Jafari, E. Salajegheh, and J. Salajegheh, Elephant clan optimization: A nature-inspired metaheuristic algorithm for the optimal design of structures. Applied Soft Computing, 113, 107892, 2021. https://doi.org/10.1016/j.asoc.2021.107892.

[17] B. Abdollahzadeh, F. Soleimanian Gharehchopogh, and S. Mirjalili, Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. International Journal of

Intelligent Systems, 36 (10), 5887-5958, 2021. https://doi.org/10.1002/int.22535.

[18] F. A. Hashim and A. G. Hussien, Snake Optimizer: A novel meta-heuristic optimization algorithm. Knowledge-Based Systems, 242, 108320, 2022. https://doi.org/10.1016/j.knosys.2022.108320.

[19] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. Computers & Industrial Engineering, 158, 107408, 2021. https://doi.org/10.1016/j.cie.2021.107408.

[20] H. Jia, X. Peng, and C. Lang, Remora optimization algorithm. Expert Systems with Applications, 185, 115665, 2021. https://doi.org/10.1016/j.eswa.2021.115665.

[21] W. Zhao, L. Wang, and S. Mirjalili, Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications. Computer Methods in Applied Mechanics and Engineering, 388, 114194, 2022. https://doi.org/10.1016/j.cma.2021.114194.

[22] M. Braik, A. Hammouri, J. Atwan, M. A. Al-Betar, and M. A. Awadallah, White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowledge-Based Systems, 243, 108457, 2022. https://doi.org/10.1016/j.knosys.2022.108457.

[23] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, Marine Predators Algorithm: A nature-inspired metaheuristic. Expert Systems with Applications, 152, 113377, 2020. https://doi.org/10.1016/j.eswa.2020.113377.

[24] Y. Jiang, Q. Wu, S. Zhu, and L. Zhang, Orca predation algorithm: A novel bio-inspired algorithm for global optimization problems. Expert Systems with Applications, 188, 116026, 2022. https://doi.org/10.1016/j.eswa.2021.116026.

[25] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software, 114, 163-191, 2017. https://doi.org/10.1016/j.advengsoft.2017.07.002.

[26] D. H. Wolpert and W. G. Macready, No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1 (1), 67-82, 1997. https://doi.org/10.1109/ 4235.585893.

[27] Y.-C. Ho and D. L. Pepyne, Simple explanation of the no-free-lunch theorem and its implications. Journal of optimization theory and applications, 115 (3), 549-570, 2002. https://doi.org/10.1023/A:1021251113462.

[28] H. Bingol and M. Yildirim, Global Optimizasyon İçin Sürü Tabanlı Bir Yaklaşım Salp Sürü Algoritması. Fırat Üniversitesi Fen Bilimleri Dergisi, 33 (1), 51-59, 2021.

[29] M. Castelli, L. Manzoni, L. Mariot, M. S. Nobile, and A. Tangherloni, Salp Swarm Optimization: A critical review. Expert Systems with Applications, 189, 116029, 2022. https://doi.org/10.1016/j.eswa.2021.116029.

[30] Y. Şekertekin and Ö. Atan, An image encryption algorithm using Ikeda and Henon chaotic maps. 24th Telecommunications Forum (TELFOR), 1-4, 2016.

[31] S. Mirjalili, S. M. Mirjalili, and A. Lewis, Grey wolf optimizer. Advances in engineering software, 69, 46-61, 2014. https://doi.org/10.1016/j.advengsoft.2013.12.007.

[32] S. Mirjalili and A. Lewis, The whale optimization algorithm. Advances in engineering software, 95, 51-67, 2016. https://doi.org/10.1016/j.advengsoft.2016.01.008.

[33] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, The arithmetic optimization algorithm. Computer methods in applied mechanics and engineering, 376, 113609, 2021. https://doi.org/10.1016/j.cma.2020.113609.

[34] N. Panagant, N. Pholdee, S. Bureerat, K. Kaen, A. R. Yıldız, and S. M. Sait, Seagull optimization algorithm for solving real-world design optimization problems. Materials Testing, 62(6), pp. 640-644, 2020. https://doi.org/10.3139/120.111529.

[35] S. Hassan, K. Kumar, C. D. Raj, and K. Sridhar, Design and optimisation of pressure vessel using metaheuristic approach. Applied Mechanics and Materials, 465,401-406, 2014. https://doi.org/10.4028/www.scientific.net/AMM.465-466.401.

[36] A. T. Kamil, H. M. Saleh, and I. H. Abd-Alla, A Multi-Swarm Structure for Particle Swarm Optimization: Solving the Welded Beam Design Problem. Journal of Physics: Conference Series, 1804 (1), 2021.

[37] Y. Çelik and H. Kutucu, Solving the Tension/Compression Spring Design Problem by an Improved Firefly Algorithm. IDDM, 1 (2255), 1-7, 2018.

[38] G. Kaur and S. Arora, Chaotic whale optimization algorithm. Journal of Computational Design and Engineering, 5 (3), 275-284, 2018. https://doi.org/10.1016/j.jcde.2017.12.006.

[39] M. Kohli and S. Arora, Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of computational design and engineering, 5 (4), 458-472, 2018. https://doi.org/10.1016/j.jcde.2017.02.005.

[40] A. H. Gandomi, X.-S. Yang, S. Talatahari, and A. H. Alavi, Firefly algorithm with chaos. Communications in Nonlinear Science and Numerical Simulation, 18 (1), 89-98, 2013. https://doi.org/10.1016/j.cnsns.2012.06.009.

[41] G. I. Sayed, G. Khoriba, and M. H. Haggag, A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell, 48 (10), 3462-3481, 2018. https://doi.org/10.1007/s10489-018-1158-6.