Slap swarm algorithm with memory mechanism and boundary collision processing

Kaixuan Yu¹, Yachao Li² and Dongsheng Zhang¹*

¹School of Software, Henan University, Kaifeng, Henan, 475000, China
²School of Software, Henan University, Kaifeng, Henan, 475000, China
*zds@vip.henu.edu.cn

Abstract: Aiming at the problem of that the standard salp swarm algorithm has low result precision and slow convergence velocity in the evolutionary process, an improved salp swarm algorithm optimization based on memory mechanism and boundary collision processing is proposed. Firstly, a Chebyshev chaotic map was used to initialize the salps to make them distribute more evenly in search space. Secondly, adding the memory mechanism introduces the individual history optimum of salps into the optimizing strategy of individual salps, which accelerates the convergence speed of the algorithm. Finally, the boundary collision rebound mechanism is introduced to ensure the effectiveness and diversity of the population. The improved algorithm is simulated on 12 types of benchmark test functions in this paper, and compared with other intelligent optimization algorithms under the same conditions. The results show that the improved algorithm has not only a obvious improvement in convergence speed and convergence accuracy, but also has better optimization performance.

1. Introduction
As a class of optimization algorithms, swarm intelligence algorithms are becoming more and more familiar in recent years. Inspired by various behaviors, phenomena or processes in nature, swarm intelligence optimization algorithms have created a series of new computing methods. In a series of algorithms, swarm intelligence algorithms play a special role in solving some problems that ordinary algorithms can not solve. There are some swarm intelligence algorithms widely used in engineering optimization, scientific computing, automatic control and other fields, such as SFLA[1], BOA[2], PSO[3] and GSO[4].

We intend to improve the salp swarm algorithm in this paper. Salp swarm algorithm, proposed by Australian Griffith University scholar Mirjalili in 2017, is a new heuristic algorithm based on swarm intelligence research. It has better characteristics than traditional optimization algorithms. Because of its advantages of simple model, fast convergence speed and few parameters, it has been well applied in the fields of engineering practice and multi-goal optimization, and SSA has gained wide attention from scholars at home and abroad, and has become a new research hotspot in the field of intelligent optimization algorithm.

Like other swarm intelligence algorithms, the algorithm has the disadvantages of local optimality, low search accuracy and slow convergence speed in solving the problem of high-dimensional complex optimization, so it needs to be further improved. The algorithm has been improved by relevant scholars to improve the algorithm's optimization accuracy and convergence speed. An improved algorithm based on attenuation factor and dynamic learning is proposed. Firstly, adding the attenuation factor in the...
update stage of leader salps can improve the local exploitation ability. Secondly, aiming to improve the global search ability of the algorithm, the dynamic learning strategy is introduced in the update stage of follower salps[5]. Aiming at the problems of local optimal solution and slow convergence speed, the method of reverse learning and chaotic local optimization is proposed, which improves the convergence speed and expands the search space[6]. A chaotic ascidian swarm algorithm is proposed by combining the ascidian swarm algorithm with chaos theory. When solving the problem of feature extraction, it can find the optimal feature subset, maximize the classification accuracy and minimize the number of selected features[7]. A method based on TARG algorithm is proposed to solve the problem of time difference of TDOA by using SSA to verify the validity and superiority of the algorithm[8].

Under the No-Free-Lunch theorem, no one algorithm applies for all optimization problems[9]. Therefore, it is very important to improve the application of existing algorithms. To solve the problem that the original SSA algorithm has slow convergence speed and easy to fall into local extremum, a new improved SSA algorithm is proposed in this paper, which is called the salp swarm algorithm with memory mechanism and boundary collision processing. Firstly, a Chebyshev chaotic map was used to initialize the salps to make them distribute more evenly in search space. Secondly, adding memory mechanism makes full use of population history information to improve the convergence speed of the algorithm. Finally, the boundary collision rebound mechanism effectively ensures the population effectiveness, and then improves the solving accuracy of the algorithm. The performance of MBSSA algorithm is tested in detail by several sets of benchmark test functions with different kinds and dimensions, while comparing with other algorithms to verify the feasibility and effectiveness of the improved strategy.

2. Salp swarm algorithm

As a random population optimization algorithm, SSA is inspired by the population mechanism of foraging and navigating in the ocean. Salp is a kind of far-sea glial chordates that feed on phytoplankton in water and perform movement in water by inhaling and spewing out sea water. Their bodies are barrel-shaped and almost completely transparent, and they are between 1 cm and 10 cm long. Because of the intelligent behavior of the group, Mirjalili and other scholars put forward the salp swarm algorithm[10].

2.1 The basic principle of Salps Swarm Algorithm

The salp swarm algorithm simulates the aggregation behavior of salps that form salps chain and then prey and move. The salp chain consists of two types: leader and followers[11]. The leader is present at the front of the chain, and the followers are behind the chain.

Similarly to other swarm-based techniques, the target environment is defined as N×D. The position of salps is defined in an n-dimensional search space where n denotes number of variables for a given problem. Therefore, the locations of all salps are stored in a two-dimensional matrix called X. It can be expressed as follows:

$$X = \begin{bmatrix} x^1_1 & x^1_2 & \cdots & x^1_d \\ x^2_1 & x^2_2 & \cdots & x^2_d \\ \vdots & \vdots & \ddots & \vdots \\ x^N_1 & x^N_2 & \cdots & x^N_d \end{bmatrix}$$  

(1)

Among them, the i individual is represented as follows:

$$X_i = [x^i_1, x^i_2, \ldots, x^i_N]$$  

(2)

The leader salp is the first vector. As the leader, it will move towards food next in some extent to play its leading role. Thus, equation 3 updates the position of the leader:

$$x^j_1 = \begin{cases} F_j + c_1 \left( (u b_j - l b_j) c_2 + l b_j \right), & c_3 \geq 0.5 \\ F_j - c_1 \left( (u b_j - l b_j) c_2 + l b_j \right), & c_3 < 0.5 \end{cases}$$  

(3)

Where $x^j_1$ is the position of the leader in the jth dimension; $F_j$ is the optimal position for the current iteration; ubj and lbj are the upper and lower bounds of the search space and both are vectors; $c_2$ and $c_3$ are random variables that are subject to a uniform distribution of 0 to 1.
In Equation 4, $c_1$ maintains a balance between global search and local search:

$$C_1 = 2e^{-\left(\frac{4l}{L}\right)^2}$$  \hspace{1cm} (4)

Where $l$ is the current iteration and $L$ is the maximum number of iterations. To update the position of follows except for the leader, this equation can be expressed as follows:

$$x_j^{i} = \frac{1}{2}(x_j^{i} + x_j^{i-1})$$  \hspace{1cm} (5)

Where $x_j^{i}$ is the position of the current $i$th follower salp in $j$th dimension; $x_j^{i-1}$ is the position of the $(i-1)$th follower salp in $j$th dimension. Followers update the positions based on the positions of their own individuals and the former.

### 2.2 The Procedure of the Salp Swarm Algorithm

According to the rules of survival of the fittest, SSA compares the fitness of the current number of iterations and the previous optimal fitness by calculating the fitness values of all salp individuals and then moves towards the food source. The procedure of the SSA can be expressed as follows:

1. Determine the target function and initialize relevant parameters.
2. Initializing the salp population gets the location of each salp.
3. The fitness corresponding to each salp position is obtained through the objective function.
4. Sort all the fitness and take the best fitness as the food source.
5. The first half of the individuals were chosen as leaders and the rest as followers.
6. Update the position of the salp leader by equation 1.
7. Update the position of the salp followers by equation 2.
8. Use the global optimal position of the salp as a new food source.
9. Determine whether the maximum number of iterations is reached: if so, end the iteration and output the best food source position; otherwise return the step4 for cycle iteration.

### 3. Salp swarm algorithm with memory mechanism and boundary collision processing

#### 3.1 Using chaos to initialize the salps

In swarm intelligence algorithm, it will play a vital role in the subsequent optimization results if the particles can be evenly distributed in the solution space in the initialization stage. The traditional SSA algorithm produces the initial salp population randomly, which affects the diversity of salp population, and then affects the iterative efficiency of SSA algorithm. The chaotic sequence has the characteristics of randomness, ergodicity and regularity, and the salp population produced by it has better diversity. The basic idea is to create chaotic sequences in the interval of $[0,1]$ by mapping relationships and then transform them into individual search spaces.

Chaos is a deterministic system with unpredictable behavior in natural science. Using chaos[12] to initialize the salp population of the SSA algorithm can greatly improve the individual diversity of the initial salp population and enormously improve the computing efficiency of the SSA algorithm. The chaotic sequence has the characteristics of randomness, ergodicity and regularity, and the salp population produced by it has better diversity. The basic idea is to create chaotic sequences in the interval of $[0,1]$ by mapping relationships and then transform them into individual search spaces.

Chaos is a deterministic system with unpredictable behavior in natural science. Using chaos[12] to initialize the salp population of the SSA algorithm can greatly improve the individual diversity of the initial salp population and enormously improve the computing efficiency of the SSA algorithm. In mathematics, various chaotic behaviors are often detected by iterative functions. Chaos functions produce different sequences for different initial values, but the interesting thing is that with the increasing number of iterations, the sequence limit values produced by chaotic functions are the same. Various randomness have huge impact on search in SSA swarm intelligence algorithms. At present, chaotic sequences are widely used in various swarm intelligence algorithms, and good results have been achieved. Chebyshev chaotic sequence is used to generate the initialized salp population of the SSA algorithm, which increases the individual diversity of the population and greatly improves the computing efficiency of the improved algorithm. Chebyshev's mapping equation is as follows:

$$x_{n+1} = \cos (k \times \cos^{-1}x_n), \quad x_n \in [-1,1]$$  \hspace{1cm} (6)
3.2 Global optimization using memory mechanism
When solving the optimization problem, if the salp population moves towards the best position of salp individual, it will not only make salp individual fall into local extreme, but also make the population diversity decrease sharply. When $c_1$ is less than 0.5, the salp followers will update their positions by the $X$ formula. In the population individual renewal, it not only uses the historical optimal position of the population, but also adds the information of the historical optimal position of each individual. Therefore, the early search experience of salp provides support for the later search, that is, the historical optimal position of each individual is the accumulation of experience from previous generations, and the historical optimal position information of population is to learn from peer location experience[13].

$$X_i = X_i^{Hbest} + c_1 \cdot rand$$  

Where $X_i^{Hbest}$ equals the historical optimal position of the $i$ individual. By adding memory mechanism, we can reuse the good genetic materials that have evolved in these environmental conditions. When the environment changes, the performance of the algorithm does not suddenly decrease, and it can quickly find the optimal value obtained by population evolution in the same state as in the past, thus shortening the time for re-evolution to find the optimal solution in the new environment. Therefore, the application of the memory mechanism lets the salp individuals make full use of the historical information of the population to increase the diversity of the population in the process of optimization evolution, so that the salp individuals can jump out of the local optimal solution region and improve the convergence accuracy and speed of the algorithm.

3.3 Boundary collision rebound mechanism
In the original algorithm, salps that exceed the boundary will be set as the boundary position, which greatly reduces the diversity of the population, so a collision rebound mechanism is proposed in this paper. When an individual goes beyond the boundary, the part that goes beyond the boundary bounces back from the boundary as a baseline. This equation can be expressed as follows:

$$T_p = X_i > U_b$$  

$$T_m = X_i < L_b$$  

$$X_i = T_p \cdot (U_b - (X_i \% U_b)) + T_m \cdot (T_m + (X_i \% L_b))$$

Where $X_i$ equals the position of the individual $i$. When $X_i$ is greater than $U_b$, $T_p$ equals 1, or $T_p$ equals 0. And when $X_i$ is less than $L_b$, $T_m$ equals 1, or $T_m$ equals 0.

4. Experiment
In order to verify the effectiveness and robustness of MBSSA proposed in this paper in solving optimization problems, the improved algorithm is compared with SSA, WOA, PSO and FA in 12 typical complex functions for 50 times.

4.1 Test functions
12 complex functions are used as fitness functions in Table 1, which contains the uni-modal functions, multi-modal functions, separable functions and non-separable functions. The uni-modal function only has one strict maximum (or minimum) within the defined upper and lower limit, which is usually used to detect the convergence speed of the algorithm. The multi-modal function has multiple local or global optimal solutions, which are often used to detect the ability of algorithm exploration and development. In addition, the dimension of the test function is an important factor, and the solving quality of the algorithm is different when dimensions are different. The dimensions of the test functions in Table 1 range from 2 to 200 for more comprehensive performance of the algorithm.
Table 1. Benchmark functions.

| Feature | Domain | optimal value |
|---------|--------|---------------|
| $f_1$ Matyas | UN | [-10,10] | 0 |
| $f_2$ Sphere | US | [-100,100] | 0 |
| $f_3$ Schwefel | UN | [-10,10] | 0 |
| $f_4$ Schwefel | UN | [-100,100] | 0 |
| $f_5$ Quartic | US | [-1.28,1.28] | 0 |
| $f_6$ Schwefel | US | [-100,100] | 0 |
| $f_7$ Schaffer | MN | [-100,100] | 0 |
| $f_8$ Kowalik | MN | [-5,5] | 3.075E-4 |
| $f_9$ Rastrigin | MS | [-5.12,5.12] | 0 |
| $f_{10}$ Ackley | MN | [-32,32] | 0 |
| $f_{11}$ Griewank | MN | [-600,600] | 0 |
| $f_{12}$ Penalized | MN | [-50,50] | 0 |

Experimental environment: Windows10 system, 8G memory, CPU2.5GHz, and the algorithm based on Matlab14b are written in M language. The maximum number of iterations is set at 1000. The size of the initial salps, particle swam and chromatid individuals are all 30.

4.2 Experimental details
The best fitness value, mean value and variance of each algorithm is calculated by running 50 times independently in different dimensions. The experimental results are shown in Table 2.

Table 2. Results of algorithms on the test benchmark functions.

|         | dim=10 | dim=30 | dim=100 |
|---------|--------|--------|---------|
|         | Best   | Ave    | Var     | Best   | Ave    | Var     | Best   | Ave    | Var     |
| MBSSA   |        |        |         |        |        |         |        |        |         |
| SSA     | 4.45E-10 | 1.04E-09 | 1.08E-19 | 2.14E-08 | 2.76E-07 | 2.89E-13 | 754.27265 | 1467.4481 | 164951.01 |
| WOA     | 2.47E-90 | 3.15E-75 | 2.91E-148 | 3.39E-83 | 5.44E-71 | 8.87E-140 | 2.26E-83 | 3.83E-73 | 2.94E-144 |
| PSO     | 1.67E-79 | 2.63E-71 | 9.00E-141 | 8.31E-15 | 2.22E-06 | 1.27E-10 | 25.478318 | 240.91961 | 49657.135 |
| FA      | 2.20E-09 | 5.00E-09 | 1.45E-19 | 8.52E-08 | 1.08E-07 | 1.06E-16 | 5.67E-06 | 8.19E-06 | 1.26E-12 |
| SSA     | 5.97E-06 | 0.0738812 | 0.069905 | 0.337044 | 2.1430451 | 1.590673 | 34.778026 | 46.321445 | 37.189984 |
| WOA     | 2.63E-61 | 8.62E-53 | 1.48E-103 | 1.29E-58 | 1.47E-49 | 6.43E-97 | 7.03E-56 | 3.14E-50 | 1.35E-98 |
| PSO     | 1.08E-25 | 7.28E-13 | 9.85E-24 | 0.0004588 | 0.0684360 | 0.0163143 | 6.2159378 | 13.39806 | 9.8713607 |
| FA      | 1.32E-05 | 1.78E-05 | 3.31E-12 | 0.0001135 | 0.0001420 | 1.49E-10 | 0.0016954 | 0.0019175 | 1.45E-08 |

| MBSSA   |        |        |         |        |        |         |        |        |         |
| SSA     | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| WOA     | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| PSO     | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| FA      | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| SSA  | WOA  | PSO  | FA  |
|------|------|------|-----|
| 3.05E-09 | 5.35E-06 | 4.02E-10 | 2.33E+02 |
| 8.52E-02 | 1.98E+02 | 1.45E+05 | 1.86E+04 |
| 1.58E-19 | 1.06E-15 | 1.55E-29 | 8.99E+00 |
| 3.38E-09 | 6.82E-09 | 3.96E-18 | 2.96E-05 |

| MBSSA  |
|--------|
| 0.0016937 | 7.58E-06 | 10538.001 | 16948.429 |

| F9  |
|-----|
| SSA  | WOA  | PSO  | FA  |
| 9.88E-16 | 8.88E-16 | 0.0013420 | 0.0022212 |

| F10  |
|------|
| SSA  | WOA  | PSO  | FA  |
| 8.88E-16 | 4.32E-15 | 4.09E-15 | 4.14E-15 |

| F11  |
|-----|
| SSA  |
| 0.0295178 | 0.0186068 | 0.0134200 | 0.0022212 |
The 50 experimental results in Table 2 show that MBSSA has the best experimental results out of all the test functions. Among them, the optimal value and the mean value can all reflect the convergence accuracy and optimization ability of the algorithm. For the uni-modal function \(f_1 \sim f_4\), MBSSA can find all the theoretical optimum. Meanwhile, with the increase of solving dimensions, the solving difficulty increases exponentially, and it is normal if the convergence accuracy of the algorithm decreases. However, compared with SSA, WOA, PSO and FA, MBSSA has better performance in optimization accuracy and solution stability. For multi-modal functions, the precision of the Algorithm is lower than that for single-modal functions. When the dimension increases from 10 dimensions to 100 dimensions, the accuracy of MBSSA decreases, but the best value and the average value of MBSSA are the best of the five algorithms in each dimension. The solving accuracy of MBSSA is less than other four algorithms in the function \(f_6\). For \(f_7, f_8, f_{10}\) and \(f_{12}\), the optimal value, mean value and variance of MBSSA are the best of the six algorithms. For \(f_9\) and \(f_{11}\), only the optimal value, mean value and variance of MBSSA are the theoretical optimal values. The results show that MBSSA is more accurate and stable than the other four algorithms. Therefore, MBSSA has better performance and robustness in solving uni-modal, multi-modal and Galway’s benchmark functions.

4.3 Analysis of the convergence curves
The convergence curves of each algorithm for different function optimization are shown in Fig 1. As seen in Fig 1, compared with SSA, WOA, PSO and FA, MBSSA has faster convergence speed and is closer to the optimal fitness value in the uni-modal test functions. It can be concluded that adding the memory mechanism can accelerate the convergence of the algorithm and MBSSA with boundary collision rebound mechanism has higher precision of optimization. With the increasing of spatial dimensions, the precision of all algorithms is decreasing, but compared with other algorithms, the precision of MBSSA is still the highest. In the multi-modal test functions, under the influence of several different local optimal solutions of the optimization function, the precision of each algorithm is lower than that of the uni-modal test functions. In the multi-modal low-dimensional functions, all the algorithms can reach the theoretical optimal value, but in the multi-modal high-dimensional functions, the optimization accuracy of MBSSA is higher than that of other algorithms.
5. Conclusions

In face of massive text data, the improved salp swarm algorithm has better global searching ability and the optimization accuracy is also enhanced than the standard algorithm. Based on the basis of the salp swarm algorithm, we propose an improved salp swarm algorithm with memory mechanism and boundary collision processing, which also introduces the chaotic map to initialize the salps to make them more evenly distributed in the search space. The salp swarm algorithm is applied to the optimization problem of the classical test functions. The algorithm is verified by the best fitness value, the average best fitness value, the standard deviation and the success rate of the solution. The results show that the improved salp swarm algorithm not only has faster convergence speed and higher convergence accuracy in the optimization problem, but also has more advantages in both feasibility and robustness. In the future work, we can consider applying some mechanism to our model, such as opposition-based learning, to further improve its effectiveness.

References

[1] Eusuff M M., Lansey K E. (2003) Optimization of water distribution network design using the shuffled frog leaping algorithm. Water Resources Planning and Management, 129(3):  210–225.
[2] Arora S., Singh S. (2019) Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23(3):  715–734.
[3] Poli R., Kennedy J., Blackwell T. (2007) Particle swarm optimization. Swarm Intelligence, 1(1):  33–57.
[4] Yang X S. (2010) Firefly algoriyhm: stochastic test functions and design optimization. International Journal of Bio-Inspired Computation, 2(2): 78–84.

[5] Lei Lin., Yue Lin., Zhilong Kang. (2020) Improved salp swarm algorithm based on reduction factor and dynamic learning. Control Theory and Applications, 37(8): 1766–1780.

[6] ZHAO X Q., YANG F., HAN Y Z., et al. (2020) An opposition-based chaotic salp swarm algorithm for global optimization. IEEE Access, 8: 36485–36501.

[7] Sayed G I., Khoriba G., Haggag M H. (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Applied Intelligence, 48(3): 1–20.

[8] Tao Chen., Mengxin Wang., Xiangsong Huang. (2018) Time difference of arrival passive location based on salp swarm algorithm. Journal of Electronics and Information Technology, 40(7): 1591–1597.

[9] Wolpert D H., Macready W G. (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1): 67–82.

[10] Hegazy A E., Makhlof M A., Eltawel G S., et al. (2018) Improved salp swarm algorithm for feature selection. Journal of King Saud University–Computer and Information Sciences, 6(3): 1–10.

[11] Mirjalili S., Gandomi A H., Mirjalili S Z., et al. (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Advances in Engineering Software, 114(6): 163–191.

[12] Hui Cheng., Chengzhong Liu. (2013) Mixed fruit fly optimization algorithm based on chaotic mappy. Computer Engineering, 39(5): 218–221.

[13] Lu Bai., Lifang Wang. (2017) Fruit fly optimization algorithm with memory. Journal of Taiyuan University of Science and Technology, 38(3): 172–177.