Dynamic Optimization Algorithm Based on Multi-modal Niche Lion Swarm by Sensitive Individuals

Qi Zhang and Mingyan Jiang*

School of Information Science and Engineering, Shandong University, Qingdao, 266237, China
Email: jiangmingyan@sdu.edu.cn

Abstract. Aiming at the problem that the model of complex dynamic system which is difficult to track its global optimal solution, this paper proposes a multi-modal niche lion swarm optimization algorithm based on sensitive individuals to solve the optimization problem of complex dynamic system. Sensitive individuals are set in the lion swarm algorithm to perceive changes in the dynamic environment, and to track the global optimal value in time. It combines with niche technology to increase the diversity of the population and avoids falling into local extremes. The dynamic environment is simulated by the dynamic function generator and compared with other optimization algorithm to verify the superiority of the algorithm proposed in this paper.

Keywords. Dynamic optimization; multi-modal; lion swarm; niche technology.

1. Introduction

In recent years, in the fields of production scheduling, artificial intelligence, combination optimization, engineering design, large-scale data processing, network communications, data mining, and capital budgeting, many complex dynamics that are closer to real life are often encountered relative to static optimization problems. Dynamic optimization problem (DOP) [1-4] means that its objective function is not only related to decision variables, but also changes with time (environment), so its optimal solution will also change with time (environment). Static optimization problem (Static optimization problem, SOP) means that its objective function is only related to decision variables, and its optimal solution does not change with time (environment). In recent years, DOP has attracted more and more researchers’ interest, and becomes a research hotspot in computational intelligence.

In order to track the best solution after environmental changes, Blackwell and Bentley [5] studied the use of charged particles in a dynamic environment. The use of charged particles could prevent the clusters of particles from converging into a single point. However, the consequence of this was that the particles would increase the repulsive force and increase the computational cost; Hu and Eberhart proposed a method to monitor the global optimal solution and the sub-global optimal solution to detect changes in the optimal solution [6]; Ursem [7] demonstrated a multi-population method in which the grouping of sub-populations was determined by the “peak-valley detection process”, but the peak-valley detection method only worked on points between known optimal values. The particles in the remaining space still could not be sampled.

The intelligent optimization algorithm provides a general framework for solving optimization problems of complex systems. It does not depend on the specific field of the problem, and does not require a clear analytical expression of the objective function, it has strong robustness to the type of problem, so intelligent Optimization algorithms have received more and more attention, in addition to...
the traditional classic intelligent optimization algorithms, such as particle swarm optimization (PSO) [8], Artificial Bee Colony Algorithm (ABC) [9], Ant Colony Optimization Algorithm (ACO) [10] and many more. The lion swarm optimization algorithm [11] is a new intelligent optimization algorithm proposed in 2018. Compared with the traditional optimization algorithm, this algorithm has many advantages. It uses the lion swarm to be divided into the lion king, the lioness and the cub and simulates three intelligent behaviors, the lion king to guard the territory, lioness cooperative hunting, cubs follow the lion king and lioness. It can be seen that the algorithm has relatively good co-evolution capabilities, and the communication mechanism is more complex and diversified, which provides us with a new idea for solving complex optimization problems. The LSO itself is a static optimization algorithm. At present, the research and improvement of this algorithm are mostly carried out in the field of static optimization, while the field of dynamic environment optimization is rarely involved. Therefore, solving dynamic optimization problems by using the parallelism of the lions optimization algorithm, the effectiveness and practicability of the global optimization problem is a new method of solving complex optimization problems.

2. Dynamic Optimization Problems and Environment Creation

2.1. Dynamic Optimization Problem

Compared with static environment problems, dynamic environment optimization problems [12] not only need to be able to track the optimal solution, but also to track the change trajectory of the optimal solution as much as possible. The dynamic optimization problem is generally described as:

\[ f(\bar{x}, t) \rightarrow \min, \bar{x} \in R^N, \ t \in T \]  \hspace{1cm} (1)

2.2. Dynamic Environment Creation

Angeline [13] proposed a method to create a series of dynamic environments based on the three-dimensional parabolic function of the following formula:

\[ f(\bar{x}) = \sum_{i=1}^{3} x_i \]  \hspace{1cm} (2)

The Angeline function can generate three different dynamic environments: linear, circular and random trajectory dynamic environments. The use of this parabolic function method is limited to simple dynamic environments.

However, practical problems are extremely complex, highly non-linear and multi-peak dynamic environments. Morrison and De Jong proposed the dynamic environment test function generator DF1 [14] function. It can specify a certain number of cones or peaks to generate a complex environment within a specified number. For two-dimensional problems, the static evaluation function in the DF1 function is defined as:

\[ f(X, Y) = \max_{x=1,m} [H_i - R_i \cdot \left( (X - X_i)^2 - (Y - Y_i)^2 \right)^\frac{1}{2}] \]  \hspace{1cm} (3)

where \( f(X, Y) \) is the fitness value at the position \( (X, Y) \), \( m \) is the number of vertebral bodies in the environment, \( (X_i, Y_i) \) is the vertex position of the first vertebral body, and \( H_i \) and \( R_i \) is the height and slope parameters of the first vertebral body, expressed as:

\[ H_i \in [H_{base}, H_{base} + H_{range}] \]  \hspace{1cm} (4)

\[ R_i \in [R_{base}, R_{base} + R_{range}] \]  \hspace{1cm} (5)
Equation (3) shows that the value of any point on the fitness surface of the search space can be determined by a maximization function. Each vertebral body is independent, determined by, \( (X_i, Y_i) \), \( H_i \), and \( R_i \) together. In order to produce varying degrees of dynamics, one-dimensional non-linear logic functions can be used:

\[
Y_t = A \cdot Y_{t-1} \cdot (1 - Y_{t-1})
\]

In equation (6), \( A \) is a constant and \( Y_t \) is the value at the \( t \) iteration. A series of different \( Y \) values can be generated by changing the \( A \) value.

3. Improved Lion Swarm Optimization

3.1. Original Lion Swarm Optimization Algorithm (LSO)

LSO is an abstract organization form based on the cooperative hunting behavior of lion swarm. The location of king is updated according to:

\[
x_i^{k+1} = g^k (1 + \gamma \| p_i^k + g^k \|)
\]

where \( g^k \) is the optimal position of the \( k^{th} \) generation population, \( \gamma \) is a random number subject to distribution. The position of the hunter is updated according to:

\[
x_i^{k+1} = \frac{p_i^k + p_h^k}{2} (1 + \alpha_c \gamma)
\]

\[
\alpha_f = \text{step} \cdot \exp \left( -\frac{10t}{T} \right)^{10}
\]

\[
\text{step} = 0.1 \cdot (\text{high} - \text{low})
\]

where \( \text{high} \) and \( \text{low} \) are the upper and lower limit mean values of each dimension of the search space, respectively, \( t \) is the current iteration number and \( T \) is the maximum iteration number. The position update function is:

\[
x_i^{k+1} = \begin{cases} 
\frac{g^k + p_h^k}{2} (1 + \alpha_c \gamma), & q \leq \frac{1}{3} \\
\frac{p_h^k + p_i^k}{2} (1 + \alpha_c \gamma), & \frac{1}{3} < q \leq \frac{2}{3} \\
\frac{g^k + p_i^k}{2} (1 + \alpha_c \gamma), & \frac{2}{3} \leq q < 1 
\end{cases}
\]

\[
\alpha_c = \text{step} \cdot \left( \frac{T-t}{T} \right)
\]

\[
g^k = \text{high} + \text{low} - g^k
\]

where \( p_i^k \) is the historical optimal position of the \( i^{th} \) lion and the \( k^{th} \) generation, \( p_h^k \) is the historical optimal position for cubs to follow the hunter \( k^{th} \) in the generation, \( g^k \) is the position where the \( i^{th} \) cub is driven in the hunting range.

3.2. Improved Lion Swarm Optimization

3.2.1. Identification of Niche Technology. For a multimodal function, if a sufficiently large number of samples are effectively sampled, the number of niches can be calculated by observing the function topographic information obtained from the samples as shown in figure 1.
Figure 1. Individual sequence sorted by distance.

The fitness fluctuation function $B_j$ of the $j^{th}$ individual in the sequence is defined as follows:

$$B_j = \frac{F(d_{j+1}) - F(d_j)}{F_{\text{max}} - F_{\text{min}}} = \frac{\Delta F}{F_{\text{max}} - F_{\text{min}}}$$  \hspace{1cm} (14)

$F(d_j)$ indicates the fitness of the $j^{th}$ individual in the sequence, $F_{\text{max}}$ and $F_{\text{min}}$ are the maximum and minimum values of the fitness function in the population, respectively. When the fluctuation is less than a preset threshold, the fitness increment between two adjacent individuals can be ignored.

3.2.2. Lion Swarm Optimization Algorithm Based on Sensitive Particles. The dynamic particle swarm algorithm based on sensitive particles [15] is a typical dynamic particle swarm algorithm. It randomly selects one or more positions when the algorithm is initialized, called sensitive particles, and calculates the fitness value of sensitive particles in each iteration. When the fitness value changes, it is considered that the environment has changed. The way to respond is to reinitialize the particle position and particle velocity according to a certain proportion. Similar to the introduction of sensitive particles into the PSO algorithm to perceive changes in the dynamic environment, and the introduction of sensitive lions into the lion optimization algorithm, the sensitive lions are re-initialized after they perceive environmental changes, so that the lions can follow the changes in the target position as a whole.

3.3. Dynamic Optimization Algorithm Based on Multi-modal Niche Lion Swarm by Sensitive Individuals (SDLSO)

Multi-modal optimization in a dynamic environment using lion optimization algorithm. It is divided into two steps. The first step is to detect changes in the environment and evaluate the changed environment, the second step is to apply the lion optimization algorithm to the detected current environment and combine the niche technology to perform multi-modal optimization.

4. Simulation Experiment and Results

4.1. Experimental Parameter Settings

The experiment uses MATLAB R2019b to simulate. For the SDLSO, set the population size $N=20$, and the upper limit of the number of iterations is $T=800$. The function dimension is $D=2$, and a single experiment is run 30 times, each runs in order to ensure fairness, the initial position of the algorithm is the same as the global optimal solution and the initial positions of the particles are generated randomly. The initial height $A$ of all vertebral bodies is evenly distributed within the allowable range. In order to effectively verify the experimental effect of the SDLSO algorithm in a dynamic environment, this experiment uses particle swarm optimization (PSO) as a comparison. Except that the above parameter settings are consistent, the two learning factors of the PSO algorithm are both 1.49, and $w=0.7$. 

4
4.2. Experimental Results

The experimental results of the PSO algorithm and the proposed SDLSO algorithm are shown in figure 2. The optimization result shows that when the number of dynamic function changes is $i=1:120$, the optimal value is $h=450-\text{fix}(i/5)$; When $i=121:790$, $h=430$; when $i=791:1000$, $h=350+\text{fix}(i-500/10)*3$; when $i=1001:1200$, $h=500$.

![Figure 2. PSO’s performance and SDLSO’s performance.](image)

A single experiment is run 30 times, and the average value is finally obtained. From the curve, we can see that compared with PSO, the improved SDLSO has a better fitness value, and they can follow up in time at the turning points of the curve. The turning points indicate changes in the external environment. Experimental results show that the SDLSO algorithm can successfully track the optimal solution in a two-dimensional dynamic environment.

5. Conclusion

Aiming at the of a complex dynamic system changing with time and being difficult to track its global optimal solution, this paper proposes a multi-modal niche lion swarm optimization algorithm based on sensitive individuals for dynamic problem. It has ability to perceive changes in the external environment, track changes in the environment in time, and solve complex dynamic system optimization problems. And the introduction of niche technology has improved the diversity of the population and avoided falling into local extremes. It uses the dynamic generator to build the environment, and also compares the effectiveness of other dynamic optimization algorithm with that of the proposed algorithm.

Acknowledgments

This work is supported in part by the Natural Science Foundation of Shandong Province (No. ZR2020MF153), and the Key Research and Development Plan of Shandong Province (Major Project of Scientific and Technological Innovation) (No. 2019JZZY01011).
References

[1] Mavrovouniotis M, Yang S, Van M et al. 2020 Ant colony optimization algorithms for dynamic optimization: A case study of the dynamic travelling salesperson problem IEEE Computational Intelligence Magazine 15 52-63.

[2] Bonilha I S, Mavrovouniotis M, Müller F M, Ellinas G and Polycarpou M 2020 Ant colony optimization with heuristic repair for the dynamic vehicle routing problem 2020 IEEE Symposium Series on Computational Intelligence (SSCI) pp 313-320.

[3] Yazdani D, Omidvar M N, Branke J, Nguyen T T and Yao X 2020 Scaling up dynamic optimization problems: A divide-and-conquer approach IEEE Transactions on Evolutionary Computation 24 (1) 1-15.

[4] Thant Sin S T 2021 The parallel processing approach to the dynamic programming algorithm of knapsack problem 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) pp 2252-2256.

[5] Santana R and Shakya S 2020 Dynamic programming operators for the bi-objective traveling thief problem 2020 IEEE Congress on Evolutionary Computation (CEC) pp 1-8.

[6] Hu X and Eberhart R C 2002 Adaptive particle swarm optimization: Detection and response to dynamic systems Proc. IEEE Congr. Evol. Compu. pp 1666-1670.

[7] Ursem R K 2000 Multinational GAs: Multimodal optimization techniques in dynamic environment Proc. 2nd Genetic Evol. Comput. Conf. pp 1-12.

[8] Kennedy J and Eberhart R 1995 Particle swarm optimization Icnn95-international Conference on Neural Networks (IEEE).

[9] Jiang M and Yuan D 2014 Artificial Bee Colony Algorithm and Its Applications (Beijing: Science Press).

[10] Dorigo M and Gambardella L M 1997 Ant colony system: A cooperative learning approach to the traveling salesman problem IEEE Trans on Evolutionary Computation 1 (1) 53-66.

[11] Liu S J, Yang Y and Zhou Y Q 2018 A swarm intelligence algorithm - Lion swarm algorithm Pattern Recognition and Artificial Intelligence 31 (05).

[12] Xiang C, Tan X and Yang Y 2014 Improved particle swarm optimization algorithm in dynamic environment The 26th Chinese Control and Decision Conference (2014 CCDC) pp 3098-3102.

[13] Angeline J P 1997 Tracking extrema in dynamic environment Proceeding of the Evolutionary Programming VI (Indianapolis, IN) (Berlin: Springer-Verlag) pp 335-345.

[14] Morrison R W and De Jong K A 1999 A test problem generator for non-stationary environments Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 pp 2047-2053.

[15] Fernandez-Marquez J L and Arcos J L 2010 Adapting particle swarm optimization in dynamic and noisy environments IEEE Congress on Evolutionary Computation pp 1-8.