International Journal of Engineering Research And Management (IJERM)
ISSN: 2349- 2058, Volume-12, Issue-05, May 2025

Research on Application of Improved Nutcracker Optimization Algorithm in Cloud Computing Task Scheduling

Pengliang Zhao, Wenhan Luo, Changxing Li

***Abstract*—The rise of cloud computing has completely transformed the architecture and operation mode of traditional IT infrastructure, and task scheduling algorithms play a crucial role in this process. An excellent task scheduling algorithm can ensure the efficient and stable operation of cloud computing systems, enhance user experience, and maximize the economic benefits of service providers. This paper addresses the issues of uneven heterogeneous resource load and insufficient resource utilization in cloud task scheduling. By taking task completion time, load balance degree, and resource waste as optimization objectives, this paper improves the recently proposed Nutcracker Optimization Algorithm (NOA) and creatively applies it to the cloud task scheduling scenario to improve the scheduling efficiency of cloud computing systems.**

This paper proposes an Improved Nutcracker Optimization Algorithm with Adaptive Chaotic Strategy (INOACS). By introducing random reverse learning, elite pool selection, and non-exclusive local search strategies, the quality of solutions is improved, and the local exploitation and global search capabilities are enhanced, avoiding the algorithm falling into local optima. To address the problems of uneven initial population distribution and the difficulty in balancing global search and local exploitation, chaotic mapping initialization and an adaptive α parameter based on population diversity are introduced, further improving the convergence speed and optimization ability of the algorithm.

A scheduling model with task completion time, load balance degree, and resource waste as optimization objectives is constructed based on INOACS. Simulation experiments verify the good performance of the INOACS algorithm under different task scales and numbers of virtual machines.

***Index Terms*—Nutcracker Optimization Algorithm; Swarm Intelligence Algorithm; Cloud Computing; Task Scheduling**

I. INTRODUCTION

Cloud computing leverages network-based resource sharing to enhance IT operational efficiency, reduce costs, and foster innovation, without requiring users to purchase and maintain expensive servers. It enables customers to deploy applications more rapidly, scale their businesses, adopt a pay-as-you-go model, and enhance their competitiveness.

As a vital pillar of next-generation information technology, cloud computing is profoundly transforming the architectural model and operational methods of traditional IT infrastructure. In a cloud computing environment, an excellent cloud task scheduling algorithm is of great significance in improving system performance, reducing operational costs, and ensuring service quality. With the continuous development of cloud computing technology and the expansion of application scenarios, the research and optimization of cloud task scheduling algorithms will remain a long-term and important topic, with significant theoretical and practical value for advancing the progress and industrial development of cloud computing.

Traditional scheduling algorithms often have large time overheads and low reliability, and they do not consider practical issues such as load balancing and multi-tenant environments. These limitations make them unsuitable for complex cloud environments and not conducive to the long-term use of cloud systems. In recent years, swarm intelligence algorithms have been increasingly applied in cloud computing task scheduling.

The Nutcracker Optimization Algorithm (NOA), proposed recently, is an excellent metaheuristic optimization algorithm that has been proven to perform exceptionally well in various optimization problems and has attracted widespread attention. In this study, we explore the application of NOA in cloud computing to address scheduling challenges in complex cloud environments and provide an efficient and feasible solution.

An Improved Nutcracker Optimization Algorithm with Adaptive Chaotic Strategy (INOACS) has been developed, establishing a mathematical model aimed at reducing task completion time, balancing virtual machine load, and minimizing resource waste in cloud task scheduling algorithms. The specific improvements include:

- Elite Pool Selection Strategy: This strategy fully utilizes high-quality individuals to enhance local exploitation capabilities.
- Non-Exclusive Local Search and Random Reverse Learning: These strategies are combined to expand the solution space and improve solution quality.
- Chaotic Mapping Initialization: This strategy increases the diversity and uniformity of the initial population, avoiding clustering issues that may arise from random initialization and improving the algorithm's convergence speed.
- Adaptive α Parameter: This parameter balances the search and exploitation processes of the algorithm, enhancing its stability and solution quality.

II. RELATED WORKS

In the field of cloud computing task scheduling, several researchers have proposed improvements to optimization algorithms: Said Nabi et al.[1] proposed an Adaptive Particle Swarm Optimization (AdPSO) algorithm, introducing a Linearly Decreasing Adaptive Inertia Weight (LDAIW) strategy. This strategy combines the benefits of Linearly Decreasing Inertia Weight (LDIW) and Adaptive Inertia Weight (AIW) to achieve a better balance between global and local search capabilities. Mohamed Abd Elaziz[2] proposed an improved Henry Gas Solubility Optimization (HGSO) algorithm. By integrating the Whale Optimization Algorithm (WOA) to enhance search capability and using Combined

Opposition-Based Learning (COBL) to update the worst solutions, the algorithm's convergence speed and solution quality are improved. Chirag Chandrashekar et al. [3]proposed an improved Ant Colony Optimization algorithm, HM-SMACA, to address the large search space, slow convergence, and difficulty in finding optimal solutions in traditional ACO algorithms. The algorithm introduces a weighting concept to optimize pheromone updating and path selection. Xueliang Fu et al.[4] proposed a hybrid Particle Swarm Optimization and Genetic Algorithm (PSO_PGA). By introducing a predation mechanism and crossover-mutation operations, and segmenting the particle swarm to process subpopulations, the algorithm expands its search range in the solution space. Su Hongbin[5] proposed an improved Grey Wolf Optimizer (PARGWO). Inspired by Particle Swarm Optimization (PSO), the algorithm modifies the grey wolf hierarchy, allowing wolves of different ranks to adopt different update methods to enhance global optimal search capability.

Due to its excellent performance, NOA has been widely applied to various optimization problems.Wang Bo et al. [6] proposed an improved NOA. The algorithm initializes the population using a generating set to enhance population uniformity and diversity, improving global search capability. A random inertia weight is introduced to balance exploration and exploitation, while a lens imaging reverse learning strategy is used to expand the search interval for the optimal solution, avoiding local optima and improving convergence precision and speed.Daqing Wu et al. [7] proposed an Adaptive Hybrid Nutcracker Optimization Algorithm (AH-NOA) to address the original algorithm's tendency to fall into local optima and slow convergence. The algorithm uses the Clarke-Wright (CW) algorithm to initialize part of the solution, making the initial population distribution more uniform. Genetic operators and local search operators are introduced to expand the solution space and optimize solution quality. An adaptive parameter adjustment mechanism balances exploration and exploitation, enhancing convergence speed.Mohamed Abdel-Baset et al. [8] proposed an improved NOA to enhance global search capability and prevent local optima. A Convergence Improvement Strategy (CIS) is introduced, along with dynamic parameter adjustment to optimize exploration and exploitation, improving convergence speed. For discrete optimization problems, discretization and reference point mechanisms further enhance solution precision and search efficiency.Jiangrong Zhao et al. [9] proposed a Multi-Strategy Adaptive Nutcracker Optimization Algorithm (MANOA) to address sensitivity to initial conditions and slow convergence. The algorithm introduces a population initialization strategy to generate a more diverse initial population and a simplified path node strategy to reduce path complexity. A dynamic inertia weight factor combined with chaotic perturbation balances global exploration and local optimization, improving performance at different stages.Yu Li et al. [10] proposed a Reinforcement Learning-based Dual-Population Nutcracker Optimization Algorithm (RLNOA). The algorithm divides the original population into exploration and exploitation subpopulations based on fitness values. The exploration subpopulation uses Random Opposition-Based Learning (ROBL) to enhance diversity, while the exploitation subpopulation employs Q-Learning to adaptively select between caching and recovery strategies, accelerating convergence and improving generalization ability.Ahmed F. Mohamed et al. [11] proposed an improved NOA (NOCG) for feature selection and global optimization. The algorithm uses the Chaotic Game Optimization (CGO) algorithm to update part of the solution, enhancing exploration and making the initial population distribution more uniform. A CrossViT (Cross-Attention Multi-Scale Vision Transformer) is introduced for feature extraction, improving feature diversity and quality. An adaptive parameter adjustment mechanism balances exploration and exploitation, while a feature selection strategy reduces dimensionality, improving classification efficiency and accuracy.Chang Xiao et al. [12] proposed an improved NOA for multi-UAV path planning. The algorithm integrates an improved sine-cosine search strategy to balance global exploration and local exploitation, focusing on global exploration in early stages and local exploitation in later stages. A lens imaging reverse learning strategy further enhances the diversity of the initial solution, improving convergence speed and precision.

III. MATH

The random reverse learning strategy is a further optimization of the traditional reverse learning strategy. Its core idea is to break the limitations of traditional reverse learning through randomness by combining random factors with the upper and lower bounds of the search space. This results in a richer set of reverse solutions that can cover a larger search space. The specific formula is as follows:

$$x_i^{ro} = lb + ub - rand \times x_i^t$$

The Elite Pool Selection Strategy is a method that maintains a pool containing multiple elite individuals to increase population diversity and guide the search direction of the algorithm. The elite individuals include those with higher fitness values in the current population and their crossover combinations.

$$x_{ep4} = \left(x_{ep1} + x_{ep2} \right) / 2$$

$$x_{ep5} = \left(x_{ep1} + x_{ep2} + x_{ep3} \right) / 3$$

$$x_{ep6} = rand \times x_{ep1} + rand \times x_{ep2} + rand \times x_{ep3}$$

$$x_{ep} = \left\{ x_{ep1}, x_{ep2}, x_{ep3}, x_{ep4}, x_{ep5}, x_{ep6} \right\}$$

The quality of the optimal solution has a significant impact on the overall performance of the algorithm. If the optimal solution falls into a local optimum, it may cause other follower individuals to also fall into local optima. The core of the Non-Exclusive Local Search (NELS) strategy is to expand the search range and improve the quality of solutions by not only focusing on the neighborhood of the current optimal solution during the local search process, but also searching for multiple potential solutions. Unlike other local search strategies, NELS modifies each dimension of the current solution space along the search space. Due to the incorporation of random operations, this strategy has the

ability to escape from suboptimal solutions.

$$x_{new}(j) = rand \times x^b(RS)$$

$$x_{new}(j) = x^b(j) - \left(x^b(RS) \times rand\right) \times eps - \left(x^b(j) = NO\right)$$

Chaotic maps possess properties of ergodicity and unpredictability. Commonly used chaotic maps include the Logistic map, Tent map, and Chebyshev map, each with its unique mathematical characteristics and application scenarios. The Tent map is characterized by its uniformly distributed outputs. In this paper, the Tent map is employed to initialize the population to enhance the diversity and uniformity of the initial population. The Tent map is a simple yet effective chaotic system, and its iterative formula is as follows:

$$x_{n+1} = \begin{cases} a \cdot x_n, & 0 \le x_n \le 0.5 \\ a \cdot (1 - x_n), & 0.5 < x_n \le 1 \end{cases}$$

To better balance the exploration and exploitation capabilities of the algorithm, this paper introduces an adaptive α parameter adjustment strategy based on population diversity. The α parameter is used to control the probability of avoiding local optima, and dynamically adjusting this parameter can effectively improve the performance of the algorithm. The specific implementation is as follows:

$$\alpha = min\left(\alpha_{max}, \alpha_{min} + (\alpha_{max} - \alpha_{min})\left(1 - \frac{t}{T}\right)e^{-\overline{\sigma_t}}\right)$$

$$\alpha = max\left(\alpha_{min}, \alpha_{max}\left(1 - \frac{t}{T}\right)^2\right)$$

The execution process of INOACS is as follows:

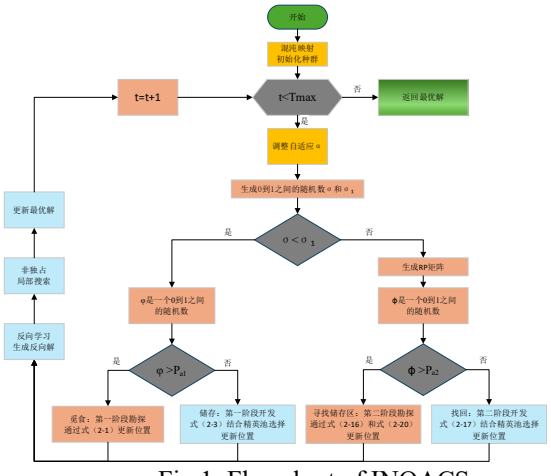

Fig.1. Flowchart of INOACS

This section compares the improved Nutcracker Optimization Algorithm with Adaptive Chaotic Strategy (INOACS) with the original Nutcracker Optimization Algorithm (NOA) and Grey Wolf Optimizer (GWO) on eight benchmark functions. The maximum number of iterations is set to 500, and each algorithm is run independently 20 times. The performance of each algorithm is tested with a population size D of 100.

Table I
Test results on benchmark functions

函数	值	GWO	NOA	INOACS
F_1	Best	1.35708E-35	0	**0**
	Mean	8.87741E-33	6.08E-218	**0**
	Std	2.02239E-32	0	**0**
F_2	Best	1.59957E-20	3.62E-194	**0**
	Mean	1.3174E-19	8.227E-84	**2.98E-152**
	Std	1.98952E-19	2.602E-83	**8.83E-152**
F_3	Best	1.09305E-18	3.11E-307	**0**
	Mean	4.96064E-16	6.57E-189	**5.44E-252**
	Std	9.76012E-16	0	**0**
F_4	Best	3.55457E-12	6.88E-167	5.44E-149
	Mean	1.82608E-11	5.62E-107	**2.22E-129**
	Std	1.48749E-11	1.78E-106	**7.01E-129**
F_5	Best	5.263631892	5.70034	**0.0221883**
	Mean	6.037421663	6.1373097	**1.9469794**
	Std	0.544787096	0.7264894	**0.3534995**
F_6	Best	1.91313E-06	2.349E-06	3.269E-06
	Mean	3.69443E-06	1.536E-05	8.992E-06
	Std	1.11128E-06	1.537E-05	4.244E-06
F_7	Best	0.000204376	0.000779	**4.512E-06**
	Mean	0.001035056	0.0015516	**5.44E-05**
	Std	0.001014728	0.0005679	**4.478E-05**
F_8	Best	3376.030981	4189.8284	**-4189.767**
	Mean	2921.285829	4189.8192	4188.0807
	Std	290.086666	**0.0112751**	3.685028

The primary objectives of the cloud task scheduling algorithm in this paper are to minimize the makespan, achieve load balancing, and reduce resource wastage. This requires the scheduling algorithm to ensure that tasks submitted by users are completed as quickly as possible, while also ensuring the rational distribution of tasks among response nodes and maximizing the utilization of computational power on each node.

Task execution time:

$$ML = \max\left\{\sum_{i=1}^{n} x_{ij} t_{ij}\right\}, j = 1, 2, \ldots, m$$

Define L_{min} as the minimum completion time under ideal conditions, where all tasks have the same length, and all virtual machines have the same processing performance, which is also the maximum value.

Normalize ML to obtain the objective function f1 for minimizing task completion time:

$$f_1 = \frac{ML - L_{min}}{L_{max} - L_{min}}$$

Load balancing degree:

$$L_{avg} = \frac{\sum_{j=1}^{m} L_j}{m}$$

Define σ as the standard deviation of the load conditions of the virtual machines under the current scheduling scheme:

$$\sigma = \sqrt{\sum_{j=1}^{m} \left(L_j - L_{avg}\right)^2 / m}$$

Normalize σ to obtain the objective function f2 for minimizing task completion time:

$$f_2 = \frac{\sigma - \sigma_{min}}{\sigma_{max} - \sigma_{min}}$$

Resource waste:

$$W_j = C_j \sum_{i=1}^{n} x_{ij} t_{ij} - \sum_{i=1}^{n} x_{ij} r_i t_{ij}$$

Normalize W to obtain the objective function f3 for minimizing task completion time:

$$f_3 = \frac{W - W_{min}}{W_{max} - W_{min}}$$

In summary, the final fitness function F is a weighted sum of the three sub-objectives.

$$F = w_1 f_1 + w_2 f_2 + w_3 f_3$$

$$w_1 = w_2 = w_3 = 1/3$$

IV. EXPERIMENTATION

To fully validate the effectiveness of the algorithm improvements, this chapter conducts simulations of the cloud environment using MATLAB to test the performance of the algorithm under various conditions.

Firstly, the improved INOACS is compared with the standard Nutcracker Optimization Algorithm (NOA), Grey Wolf Optimizer (GWO), and Whale Optimization Algorithm (WOA) under the same conditions. The population size is set to 100, the maximum number of iterations is 200, the number of virtual machines is 8, and the number of tasks is 800.

The specific attributes of the tasks are randomly generated using a normal distribution. The minimum task length is 300, the maximum task length is 1000, and the average task length generated is 644.37. The minimum computational power requirement for tasks is 1, the maximum is 10, and the average computational power requirement generated is 5.57.

The virtual machine configurations are also randomly generated. The randomly generated virtual machine configurations are as follows: 4 low-performance virtual machines with computational capacity and processing speed of 8 and 1, respectively; 2 medium-performance virtual machines with computational capacity and processing speed of 16 and 2, respectively; and 2 high-performance virtual machines with computational capacity and processing speed of 32 and 4, respectively.

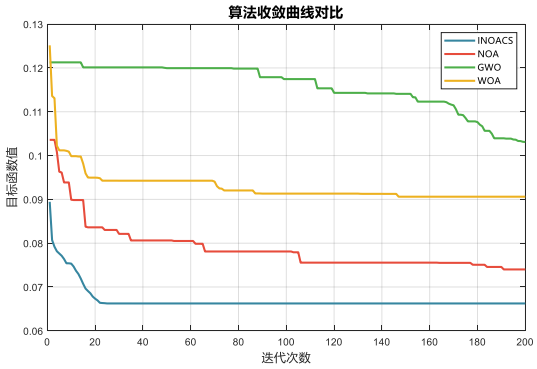

Fig.2. Comparison of Convergence Curves for INOACS, NOA, GWO, and WOA

The comparison of convergence curves is shown in Figure 2. It can be seen that INOACS has a significant advantage in convergence speed compared to NOA, GWO, and WOA, which proves the effectiveness of the chaotic mapping strategy in enhancing the algorithm's early exploration capability of the solution space. The steady decline of the convergence curve demonstrates the effectiveness of the adaptive α parameter in balancing global search and local exploitation. The fitness value of INOACS eventually stabilizes at a lower level, indicating that the improved algorithm's optimization ability has also been enhanced.

To comprehensively evaluate the performance of the INOACS algorithm in cloud task scheduling scenarios, four evaluation indicators—total objective function value, task completion time, resource wastage, and load balancing degree—are used to assess the algorithm's performance. To ensure the authenticity, fairness, and applicability of the experiments, the subsequent experimental settings are as follows: population size is set to 100, the maximum number of iterations is 200, the number of virtual machines is 4 and 8, and the number of tasks is 200, 400, 600, and 800. Under the same conditions, experiments are conducted independently to simulate different practical application scenarios and to evaluate the algorithm's performance under varying task complexities.

Table 2
Test results on benchmark functions

虚拟机数量	算法	任务数			
		200	400	600	800
4	INOACS	**0.072112**	**0.067844**	**0.067769**	**0.071452**
	NOA	0.079993	0.075579	0.075297	0.07928
	GWO	0.081887	0.115872	0.170325	0.079378
	WOA	0.08273	0.075794	0.075673	0.080421
8	INOACS	**0.060424**	**0.060765**	**0.059692**	**0.060731**
	NOA	0.070953	0.072271	0.073826	0.069267
	GWO	0.079441	0.093752	0.096524	0.090756
	WOA	0.088142	0.076081	0.078387	0.07858

Table 2 shows the fitness values of the algorithms under different task scales when the number of virtual machines is 4. INOACS outperforms the other four comparison algorithms in all cases, demonstrating that INOACS can balance multiple optimization objectives and find the optimal scheduling strategy.

Figure 3 shows the bar chart of fitness values for each algorithm under different task scales. It can be more intuitively seen that as the number of virtual machines increases from 4 to 8, the advantage of INOACS becomes more pronounced, proving that the improved algorithm can still maintain good stability in high-complexity scheduling

scenarios.

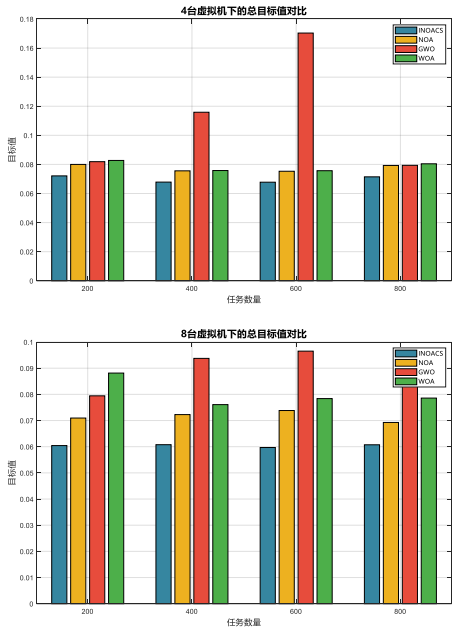

Fig.3. Comparison of Fitness Values for INOACS, NOA, GWO, and WOA

V. CONCLUSION

In summary, the INOACS algorithm can effectively balance multiple optimization objectives, including task completion time, load balancing, and resource wastage, under the same experimental conditions. In various scenarios, the standard NOA algorithm outperformed GWO and WOA, demonstrating its applicability in cloud computing task scheduling. INOACS consistently outperformed NOA across all evaluation metrics, which not only proves the effectiveness of the improvements but also highlights INOACS's excellent cloud task scheduling capability. INOACS can serve as a scheduling algorithm for cloud computing systems, meeting scheduling demands of varying complexities and enhancing the performance of cloud computing systems.

This chapter has conducted an in-depth study on cloud task scheduling based on the Improved Nutcracker Optimization Algorithm with Adaptive Chaotic Strategy (INOACS). Firstly, a scheduling model was constructed with task completion time, load balancing degree, and resource wastage as optimization objectives, and the design concept of the fitness function was elaborated in detail.

Through simulation experiments, the INOACS algorithm was compared with the standard Nutcracker Optimization Algorithm (NOA), Grey Wolf Optimizer (GWO), and Whale Optimization Algorithm (WOA). The experimental results show that under the same conditions, the INOACS algorithm significantly outperformed the other comparison algorithms in multiple evaluation metrics, including task completion time, resource wastage, and load balancing degree. INOACS maintained stable high-performance across different task scales and numbers of virtual machines, demonstrating its applicability and superiority in complex cloud environments. This provides valuable references for the design and optimization of task scheduling algorithms in practical cloud computing systems.

REFERENCES

[1] Nabi S, Ahmad M, Ibrahim M, et al. AdPSO: adaptive PSO-based task scheduling approach for cloud computing[J]. Sensors, 2022, 22(3): 920.

[2] Abd Elaziz M, Attiya I. An improved Henry gas solubility optimization algorithm for task scheduling in cloud computing[J]. Artificial Intelligence Review, 2021, 54(5): 3599-3637.

[3] Chandrashekar C, Krishnadoss P, Kedalu Poornachary V, et al. HWACOA scheduler: Hybrid weighted ant colony optimization algorithm for task scheduling in cloud computing[J]. Applied Sciences, 2023, 13(6): 3433.

[4] Fu X, Sun Y, Wang H, et al. Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm[J]. Cluster Computing, 2023, 26(5): 2479-2488.

[5] 苏鸿斌.基于改进灰狼算法的云资源调度策略研究[J].智能计算机与应用,2024,14(02):28-34.

[6] 王博,李昕涛,王珂,等.改进星鸦优化算法的无刷直流电机控制研究[J]. 微特电机,2025,53(01):53-59.DOI:10.20026/j.cnki.ssemj.2025.0008.

[7] Wu D, Yan R, ** H, et al. An adaptive nutcracker optimization approach for distribution of fresh agricultural products with dynamic demands[J]. Agriculture, 2023, 13(7): 1430.

[8] Abdel-Basset M, Mohamed R, Hezam I M, et al. An improved nutcracker optimization algorithm for discrete and continuous optimization problems: Design, comprehensive analysis, and engineering applications[J]. Heliyon, 2024, 10(17).

[9] Dinamik P T, ZHAO J, DING H, et al. Multi-Robot Path Planning Based on the Improved Nutcracker Optimization Algorithm and the Dynamic Window Approach[J]. Sains Malaysiana, 2024, 53(12): 3409-3423.

[10] Li Y, Zhang Y. A Reinforcement Learning-Based Bi-Population Nutcracker Optimizer for Global Optimization[J]. Biomimetics, 2024, 9(10): 596.

[11] Mohamed A F, Saba A, Hassan M K, et al. Boosted nutcracker optimizer and chaos game optimization with cross vision transformer for medical image classification[J]. Egyptian Informatics Journal, 2024, 26: 100457.

[12] Chang X, Yang H, Zhang B. Multi-Unmanned Aerial Vehicle Path Planning Based on Improved Nutcracker Optimization Algorithm[J]. Drones, 2025, 9(2): 116.