

Article

UAV Swarm Dynamic Task Allocation via Merged Coordination-Optimized Pigeon-Inspired Optimization

Yingran Zhao ^{1,2,*}  and Wenju Hu ¹

¹ School of Mechanical-Electronic and Vehicle Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China; huwenju@bucea.edu.cn

² School of Beijing, Beihang University, Beijing 100191, China

* Correspondence: zhaoyingran06@126.com

Highlights

What are the main findings?

- The simple, randomly generated optimal pigeon in the first phase of the basic Pigeon-inspired Optimization (PIO) is replaced by identifying the optimal pigeon within the flock through a strategy of group merging and group learning.
- The two phases of the basic PIO are combined through a dynamic factor, replacing the mode in the basic PIO where the phases are only controlled by the number of iterations.

What are the implications of the main findings?

- For the dynamic task allocation problem of heterogeneous UAV swarms, striking a balance between optimization performance and solution efficiency of the algorithm is crucial. This paper proposes Merged Coordination-Optimized Pigeon-Inspired Optimization (MCOPIO) algorithm by improving the basic PIO algorithm. The proposed algorithm not only enhances the performance when tackling task allocation problems, but also avoids a sharp rise in computational complexity, resulting in a running time comparable to that of the basic PIO.

Abstract

To tackle the dynamic assignment problem of unmanned aerial vehicle (UAV) swarms, a merged coordination-optimized pigeon-inspired optimization (MCOPIO) algorithm based on the pigeon-inspired optimization (PIO) algorithm is proposed in this paper. The algorithm disrupts the original pigeon distribution via random grouping and performs mutual learning and optimization within the new groups. After dynamic optimization, the underperforming pigeons are discarded, and the flock is reorganized. Subsequently, the two stages of the basic PIO are integrated through a dynamic factor. These improvements overcome the limitations of the basic PIO algorithm, such as insufficient global search capability, poor stability, and disconnection between the two algorithm stages. Comparative experiments are conducted with the state-of-the-art intelligent computing algorithms, such as the basic PIO, particle swarm optimization (PSO), genetic algorithm (GA), and improved consensus-based bundle algorithm (ICBBA), the comparative results verify the feasibility and effectiveness of our improved PIO for UAV swarm dynamic task allocation.

Keywords: unmanned aerial vehicle(UAV); dynamic task assignment; pigeon-inspired optimization (PIO); population merging; joint optimization



Academic Editor: Pablo Rodríguez-Gonzálvez

Received: 22 May 2026

Revised: 23 June 2026

Accepted: 24 June 2026

Published: 30 June 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons](https://creativecommons.org/licenses/by/4.0/)

[Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

With the continuous advancement of science and technology, the demand for unmanned aerial vehicles (UAVs) has been steadily increasing, accompanied by the continuous expansion of their application scenarios and the growing complexity of their assigned tasks. UAVs have been widely deployed in agriculture, industry, and even the service sector, playing an indispensable role in various applications such as food delivery, agricultural pesticide spraying [1], fire rescue [2], and preventive inspection.

However, as the types and quantities of tasks continue to grow, UAV swarms are confronted with significant challenges in realizing rapid and effective task allocation in dynamic environments, especially when multiple mutually conflicting tasks need to be balanced simultaneously. Figure 1 shows the schematic diagram of UAV task allocation.

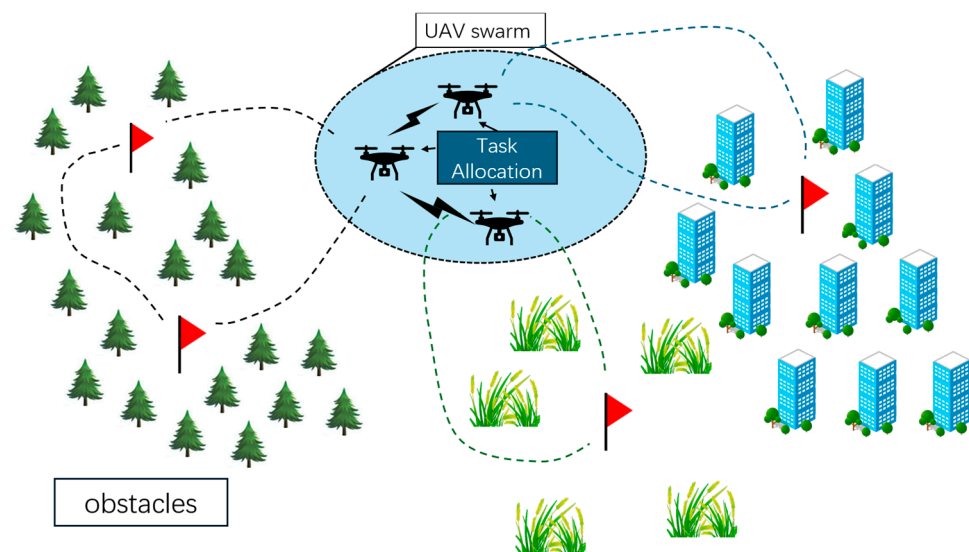


Figure 1. Schematic Diagram of UAV Task Allocation.

To address the task allocation problem in dynamic environments, scholars have proposed various algorithms, such as Particle Swarm Optimization (PSO) [3], Genetic Algorithm (GA) [4], Consensus-Based Bundle Algorithm (CBBA) [5], and Non-dominated Sorting Genetic Algorithm II (NSGA-II) [6]. These four algorithms represent classic benchmarks of different technical routes, architectural paradigms, and optimization objectives in the field of task allocation. PSO and GA are typical representatives of centralized single-objective metaheuristic algorithms and are the most widely used basic benchmarks for task allocation problems; CBBA is a classic benchmark algorithm in distributed task allocation; and NSGA-II is the gold standard for multi-objective evolutionary algorithms.

Inspired by the foraging behavior of birds, PSO guides the population to converge toward the optimal solution using two types of information: the historical best position of each individual and the global best position of the entire swarm. It features simple implementation and fast convergence, making it suitable for quickly solving medium-scale problems. Scholars have further optimized PSO by introducing chaotic mapping, clustering mechanisms, reinforcement learning, and other methods, thereby enhancing the algorithm's global search capability and search accuracy [7,8]. In contrast, GA is inspired by biological evolution and achieves convergence through selection, crossover, and mutation operations. It boasts strong global search ability, good robustness, and can effectively handle discrete and nonlinear complex optimization objectives. Scholars have improved GA using methods such as adaptive parameter adjustment and hybrid algorithms [9,10]. These algorithms have exhibited desirable performance in addressing complex optimization challenges in UAV control, including task allocation and path planning.

As a classic benchmark algorithm, the Consensus-Based Bundle Algorithm mainly solves task allocation problems through distributed auction and consensus negotiation. It features low communication overhead, strong robustness, and natural adaptability to dynamic environments. By introducing strategies such as the clustering-first task allocation strategy, pheromone-guided task selection mechanism, energy consumption-optimal dynamic allocation strategy, and improving the algorithm itself, scholars have enhanced the computational efficiency of CBBA and proposed the Improved Consensus-Based Bundle Algorithm (ICBBA) [11]. For heterogeneous UAV dynamic mission reallocation scenarios under fault conditions, recent research has further extended the CBBA framework: Bi et al. [12] proposed a targeted improved CBBA with three core optimizations, including a heartbeat-hold-based distributed UAV damage detection mechanism for real-time fault response, a timeliness-weighted bidding strategy to dynamically adjust task priority, and a time-order priority insertion rule to optimize bundle construction efficiency. The improved algorithm achieves faster solving speed and better reallocation effectiveness when dealing with UAV damage events in dynamic environments.

With its three core mechanisms of fast non-dominated sorting, crowding distance diversity maintenance, and elite preservation, NSGA-II has become a benchmark algorithm in the field of multi-objective optimization and is widely used in UAV swarm task allocation. It can simultaneously optimize conflicting objectives such as task completion time, energy consumption, and load balancing. To address the high computational complexity and insufficient real-time performance of serial NSGA-II in large-scale UAV swarm scenarios, scholars have proposed various parallel improved versions, including master-slave parallelism [13], island model parallelism [14], and parallelized a posteriori optimization [15], which have improved the algorithm's performance.

In addition to traditional evolutionary and swarm intelligence methods, emerging artificial intelligence techniques have been gradually integrated into UAV cooperative decision-making. Aiming at complex swarm operation requirements, Yu et al. explored a large language model empowered multi-UAV task planning paradigm, providing a novel technical reference for intelligent scheduling and dynamic task allocation of UAV clusters [16].

Pigeon-inspired Optimization (PIO) is also an intelligent optimization algorithm [17,18]. By learning from the homing behavior of pigeons, it consists of two phases: the map and compass operator and the landmark operator, which guide convergence using the global optimum and the population center, respectively. Scholars have improved the PIO algorithm by means of mutation operators or the introduction of new factors [19,20].

To address the problems of insufficient global search ability, proneness to falling into local optima, and optimization inefficiency arising from the transition between the two stages being controlled solely by the number of iterations in the PIO algorithm, this paper proposes a Merged Coordination-Optimized Pigeon-Inspired Optimization (MCOPIO) algorithm based on the pigeon-inspired optimization algorithm.

This algorithm also comprises two steps. First, random grouping learning is adopted to fully utilize global information and search for the optimal solution [21]. Second, the optimization mode is dynamically selected according to the gap between different solutions, and coordinated optimization is realized using two types of information: the global optimum and the swarm center.

The main contributions of this paper are summarized as follows:

(1) An improved pigeon-inspired optimization algorithm integrated with a dedicated dynamic factor mechanism is proposed, which can effectively adapt to the time-varying characteristics of dynamic task allocation for UAV swarms.

(2) Multiple groups of comparative experiments and extended stability tests are carried out to fully verify the optimization performance, environmental adaptability and engineering practical value of the proposed MCOPIO algorithm.

The overall structure of this paper is arranged as follows: Section 2 and Section 3 respectively present the problem description of task allocation and the basic constraint models. Section 4 introduces the evaluation function of the algorithm. Section 5 elaborates on the principle of the original pigeon-inspired optimization algorithm and the improved mechanism of dynamic factor fusion proposed in this paper. Section 6 designs benchmark test functions and comparative experimental schemes, presents simulation results, and completes multi-algorithm comparison and stability verification. Finally, Section 7 concludes the whole paper.

2. Problem Formulation

In real-world environments, various dynamic changes may arise during UAV mission execution, such as mission addition or UAV failure. To address these dynamic changes, this paper designs a dynamic task allocation triggering mechanism that initiates the dynamic task allocation process according to variations in the UAV system status and mission requirements. The main triggering conditions include mission addition and UAV failure.

When the number of tasks increases, the multi-UAV task allocation system needs to incorporate newly generated tasks based on the existing allocation scheme and re-plan each UAV's task execution sequence. In dynamic environments, new tasks such as emerging target surveillance and emergency rescue missions emerge continuously. Under such circumstances, the multi-UAV system must assign these new tasks to appropriate UAVs in real time to ensure all missions are addressed promptly. The upper-right section of Figure 2 depicts the scenario of task expansion: while four UAVs are executing eight predefined tasks, five additional tasks are suddenly generated. The UAV swarm then redistributes these five new tasks and integrates them into their respective task sequences. For UAV failure scenarios: if a single UAV malfunctions unexpectedly and cannot complete its subsequent tasks, this paper develops a reasonable dynamic task allocation strategy to reassign the unfinished missions of the faulty UAV to other fully operational UAVs. The lower-right section of Figure 2 provides a schematic diagram of UAV failure. Four UAVs collaboratively perform eight tasks, but one vehicle suffers an abrupt breakdown. The remaining homogeneous UAVs take over all tasks originally assigned to the faulty UAV and carry out redistribution.

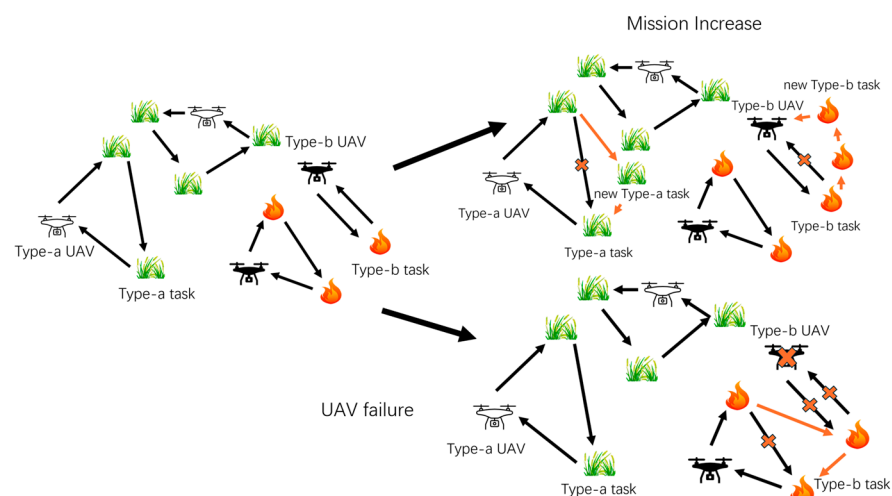


Figure 2. Schematic of Mission Increase Scenario.

Task allocation in dynamic environments mainly includes two aspects: pre-allocation and dynamic allocation. Pre-allocation refers to the allocation of existing tasks by UAVs in an initial static environment, aiming to optimize the task allocation results as much as possible and ensure efficient task execution under static conditions. Dynamic allocation means that when the trigger conditions are met, UAVs assign newly added tasks to appropriate UAVs based on the originally allocated tasks. This strategy is suitable for scenarios where the number of tasks is small, or the impact of newly added tasks is limited.

3. Method for UAV Swarm Control

The dynamic task assignment problem for multiple unmanned aerial vehicles (UAVs) adopts the Cooperative Multiple Task Assignment Problem (CMTAP) model [22,23]. This model was first formally proposed by Shima et al. based on multi-UAV systems, specifically designed to solve the problem of multi-task scheduling and assignment with complex constraints in multi-agent cooperative scenarios.

The core of the model is to optimally assign a series of heterogeneous tasks to heterogeneous agents with different capabilities under the premise of satisfying multi-dimensional constraints, to achieve the optimization of global objectives.

In this paper, assume that the set of UAVs is $I = \{1, 2, \dots, N_u\}$ and the set of tasks is $j = \{1, 2, \dots, N_T\}$. The goal is to find an optimal task assignment scheme that satisfies constraints such as UAV and task execution, to maximize the system utility, which is expressed as

$$\max \sum_{i=1}^{N_u} \left(\sum_{j=1}^{N_T} S_{ij}(p_i) x_{ij} \right) \quad (1)$$

$$\text{subject to } \sum_{j=1}^{N_T} x_{ij} \leq L_i \quad (2)$$

$$\sum_{i=1}^{N_u} \sum_{j=1}^{N_T} x_{ij} \leq \min \left\{ N_T, \sum_{i=1}^{N_u} L_i \right\} \quad (3)$$

$$x_{ij} \in \{0, 1\}, \forall i \in I, \forall j \in J \quad (4)$$

$$t_j^{\text{start}} \leq t_j \leq t_j^{\text{end}}, \forall j = 1, 2, \dots, N_T \quad (5)$$

$$d(U_i, G) \leq R_{\text{com}}, \forall i \in I \quad (6)$$

$$\sum_{j \in \mathcal{T}_i} b_{ij} \leq B_i^{\text{max}}, \forall i \in I \quad (7)$$

where (1) presents the marginal score function for task allocation, $S_{ij}(p_i)$ denotes the reward obtained by UAV i when performing task j along path p_i . The binary variable $x_{ij} \in \{0, 1\}$ is the decision variable: $x_{ij} = 1$ indicates that task j is assigned to UAV i , and $x_{ij} = 0$ means that UAV i fails to win the bid for task j . (2) enforces the constraint that the number of tasks assigned to UAV i cannot exceed its operational capacity, L_i denotes the maximum number of tasks that UAV i can execute. (3) further restricts the total number of assigned tasks to be no greater than the maximum task capacity of all UAVs. (4) defines the binary decision variables for the task allocation problem, while (5) represents the time window constraint associated with task t_j . (6) is the communication connectivity range constraint, which ensures that the Euclidean distance $d(U_i, G)$ between UAV i and the command center G does not exceed the maximum effective communication radius R_{com} , preventing communication outages that would disrupt task coordination. (7) represents the communication bandwidth constraint, where $\mathcal{T}_i = \{j \in J \mid x_{ij} = 1\}$ denotes the set of tasks assigned to UAV i , b_{ij} is the bandwidth required for UAV i to transmit data related to

task j , and $B_i^{\max}$ is the maximum available bandwidth of UAV i ; this constraint ensures that the total data transmission volume of each UAV does not exceed its communication link capacity, avoiding congestion and ensuring real-time task execution.

Considering the differences in UAV capabilities and task types, this paper introduces an additional constraint on task types to ensure the matching between task types and UAV types, which is expressed as

$$j \in M_i, \quad \forall j \in \{j | x_{ij} = 1\} \quad (8)$$

where M_i denotes the set of task types that can be executed by UAV i .

Due to the randomness of task occurrence, the paths of UAVs to task points are also random. Therefore, there must be conflicts in the task paths between different UAVs. To avoid collisions between UAVs caused by path conflicts, this paper adopts the artificial potential field method proposed by Khatib for obstacle avoidance to solve the path conflict problem [24–26]. When two UAVs are relatively close to each other, the repulsive force is increased to make the UAVs move away from each other to complete avoidance. The relational expression between the received repulsive force and the distance is as follows:

$$F_{\text{rep},ij} = -\nabla U_{\text{rep}}(d_{ij}) = \begin{cases} k_{\text{rep}} \left(\frac{1}{d_{ij}} - \frac{1}{d_{\text{safe}}} \right) \frac{1}{d_{ij}^2} \cdot \frac{X_i - X_j}{d_{ij}}, & 0 < d_{ij} < d_{\text{safe}} \\ 0, & d_{ij} > d_{\text{safe}} \end{cases} \quad (9)$$

where d_{ij} denotes the Euclidean distance between UAV i and UAV j , X_i and X_j are the spatial position coordinates of UAV i and UAV j , respectively, d_{safe} is the preset safe distance threshold between UAVs, and k_{rep} is the repulsion gain coefficient.

4. Task Allocation Effectiveness Evaluation Model

The evaluation model is crucial for intelligent optimization algorithms. It reflects the optimization degree of the algorithm and guides its optimization direction. Since the task environment considered in this paper is a dynamic and stochastic task allocation problem within a certain spatial scope, the weighted sum method is adopted in the evaluation model to balance benefits and costs.

Due to the limited battery capacity of UAVs, the effective working time of each UAV is constrained. Consequently, the flight time of a UAV between different task points in the air directly affects the total effective time available for performing tasks [27]. The shorter the flight time, the longer the UAV can spend on task execution, thereby enabling it to complete more tasks. The flight cost of the UAV when executing tasks in the task list is expressed as

$$c_{ik}(a_i) = \begin{cases} d_{0,1}/u_i, & \text{if } k = 1, \\ d_{k-1,k}/u_i, & \text{otherwise,} \\ d_{k,0}/u_i, & \text{if } k = |a_i|, \end{cases} \quad (10)$$

where c_{ik} represents the flight time consumed by UAV i during task execution, a_i refers to the task list of UAV i , $d_{i,j}$ represents the distance between the i -th and j -th task points ($d_{0,1}$ stands for the distance from the initial position of the UAV to the first task point), and u_i is the flight speed of UAV i .

Since all tasks are required to be executed before their deadlines, the issue of whether the UAV swarm can complete the tasks in a timely manner should also be considered. Therefore, this paper also introduces a revenue evaluation regarding the completion rate of tasks. The revenue evaluation function is expressed as

$$r_i(a_i) = \sum_{k=1}^{|a_i|} \varepsilon(a_{ik}) \quad (11)$$

where $\varepsilon(a_{ik})$ denotes the complexity of the k -th task in the task list a_i .

Combined with the above analysis, flight cost and task completion quantity need to be considered. Hence, the evaluation function is defined as

$$F = \begin{cases} a \cdot r_i(a_i) - b \cdot \sum_{k=1}^{|a_i|} c_{ik}(a_i), & \text{if } a_i \text{ is feasible,} \\ 0, & \text{otherwise,} \end{cases} \quad (12)$$

where a and b denote the weight coefficients of the system's revenue and cost, respectively. By adjusting the weights of revenue and cost, the model can better adapt to the task environment faced by UAVs. For example, when the number of UAVs is large while the number of tasks is small, the risk of task failure due to timeout is significantly reduced. In this case, reducing the revenue weight and increasing the cost weight can more effectively highlight the performance of task allocation, and better guide the convergence of the optimization function.

5. Improved Pigeon-Inspired Optimization Algorithm

5.1. Basic Pigeon Inspired Optimization

The principle of basic PIO is divided into two steps. The first step is the map and compass operator, and the second step is the landmark operator. Figure 3 is the schematic diagram of the basic PIO algorithm.

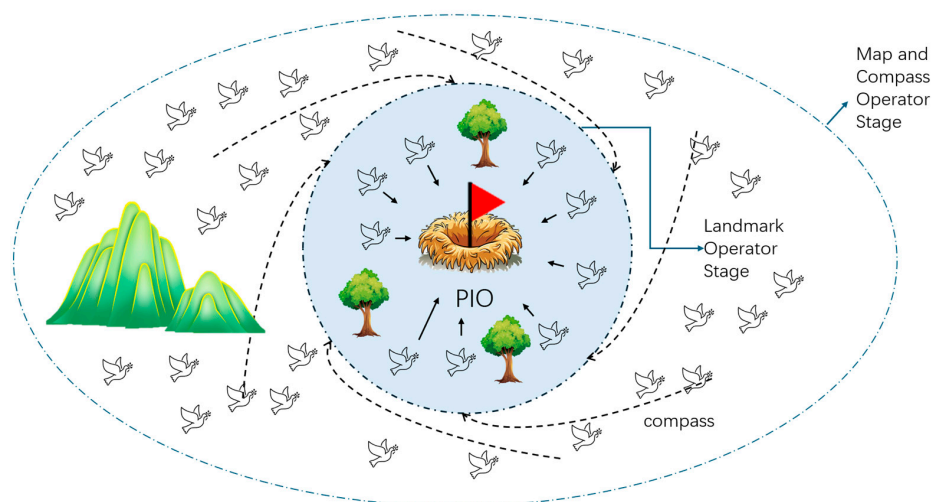


Figure 3. Schematic diagram of basic PIO.

In the first step, the pigeon flock gradually flies toward the destination by perceiving the geomagnetic field and the solar elevation angle. The iterative formulas for the position and velocity of the pigeons are as follows:

$$V_i^{t+1} = -R^{(t+1)} \cdot V_i^t + \text{rand}(x_g^t - x_i^t) \quad (13)$$

$$x_i^{t+1} = x_i^t + V_i^{t+1} \quad (14)$$

where N_{c1} is the maximum number of iterations in this step; t is the current iteration count, satisfying $1 \leq t \leq N_{c1}$; x_g^t denotes the position of the optimal pigeon in the t -th generation of the pigeon flock; x_i^t represents the position of the i -th pigeon in the same generation; R is

an operator with $R \in [0,1]$, and ‘rand’ is a random number uniformly distributed in the interval $[0,1]$.

As the pigeon flock gets closer to the target, its dependence on the sun and the magnetic compass gradually decreases, and the flock instead relies on pigeons that are more familiar with the landmarks. The iterative formula for the position of pigeons in the landmark operator is as follows:

$$NP^t = \text{ceil}\left(\frac{NP^t}{2}\right) \quad (15)$$

$$x_C^t = \frac{\sum_{i=1}^{NP^t} x_i^t \cdot f(x_i^t)}{\sum_{i=1}^{NP^t} f(x_i^t)} \quad (16)$$

$$x_i^{t+1} = x_i^t + \text{rand}(x_C^t - x_i^t) \quad (17)$$

where ‘ceil’ represents the ceiling function (rounding up); $t \leq Nc_2$, Nc_2 is the maximum number of iterations in this stage; the weighting coefficient $f(x_i^t)$ is the fitness value (position utility function value) of the i -th pigeon at the t -th iteration, and ‘rand’ is a random number uniformly distributed in the interval $[0,1]$.

5.2. Improved Pigeon Inspired Optimization

There are still some issues to be adjusted in the basic PIO algorithm. First, in the first part of the basic PIO algorithm, the pigeons adjust their positions mainly based on the position of the optimal pigeon in the flock. Such one-way learning within the population leads to insufficient utilization of global information, making it easy to miss information and fall into local optima. Moreover, the optimization performance is related to the position of the initial optimal pigeon in the flock, resulting in a lack of robustness and stability of the algorithm. In addition, whether the basic PIO algorithm switches from the map and compass operator phase to the landmark operator phase is determined only by the number of iterations Nc_1 . This prevents the algorithm from accurately judging the optimization degree of the pigeon flock: entering the second stage too early tends to cause premature convergence and trap the algorithm in local optima, while entering too late increases the algorithm complexity and computational time.

To address these deficiencies in the basic PIO algorithm, this paper proposes an improved algorithm based on the basic PIO. Firstly, the algorithm adopts the idea of group merging to find the pigeon with the optimal position in the flock through optimized learning and takes this pigeon as the leader of the flock. The remaining pigeons optimize their own positions by referring to the leader’s position. The merging formula of the pigeon flock is expressed as

$$N_i = \{pop_i, pop_{n+1-i}\} \quad (18)$$

$$x_w' = x_w + \text{fix}\left[\omega \cdot \left(1 - \frac{n}{N}\right) \cdot (x_b - x_w)\right] \quad (19)$$

where pop_i represents the information of the i -th pigeon in the original pigeon population. Formula (18) groups every two pigeons in the original population to form new subpopulations. When the total number of pigeons is odd, the remaining one pigeon is randomly added to any subpopulation. x_b is the pigeon with a better position in the new subpopulation, and x_w is the pigeon with a worse position. N is the preset termination condition for learning times, and n is the current learning time, $n \leq N$. ω is a random reversal factor, which takes a random value of 1 or -1 . ‘fix’ is the rounding function.

After the original pigeon population is formed into new subpopulations, the pigeons with worse positions in the subpopulations learn from the pigeons with better positions

according to Formula (19). Then the superiority of their positions is re-judged, and the learning continues until the learning termination condition is satisfied. At this time, the pigeons with worse positions are abandoned. All pigeons with better positions in the new subpopulations are merged into a brand-new pigeon population, and the operations in Formulas (18) and (19) are repeated until only one pigeon is finally generated as the leader pigeon.

By adopting this method to search for the leader pigeon, this paper not only makes full use of the information of all pigeons in the flock, improves the learning efficiency of the pigeon flock, and enhances the global search ability, but also avoids the problems of local optima and instability that may be caused when all pigeons learn from the randomly generated optimal pigeon.

After solving the above problems, we further address the issue of the two-stage separation in the basic PIO, which prevents an accurate transition from the first stage to the second stage. In this paper, dynamic factors c_1 and c_2 are adopted to allocate the weights of individual learning and swarm optimization. The two stages of the basic PIO are integrated into a single formula. This formula simultaneously incorporates the two optimization strategies from the two stages of the basic PIO. According to the positional relationship between the current pigeon and the leader pigeon, the dynamic operators c_1 and c_2 take different values, thereby controlling the dominant optimization strategy. Figure 4 illustrates the differences between the basic PIO algorithm and the MCOPIO algorithm in this part.

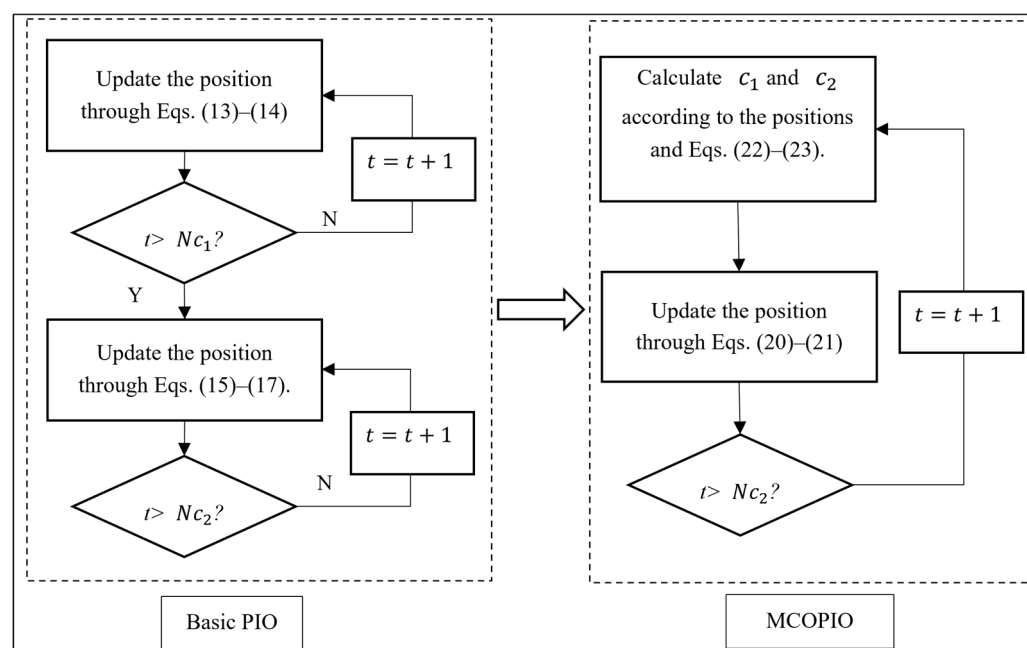


Figure 4. Schematic Diagram of Comparison Between Basic PIO and MCOPIO.

When the distance between the pigeon and the leader pigeon is large, the formula from the first stage of the basic PIO is mainly used to optimize the pigeon's position, so that the pigeon can keep up with the flock. When the distance is small, the optimization method from the second stage of the basic PIO is mainly adopted to optimize based on the overall information. The improved optimization formula is expressed as

$$x_i^{t+1} = x_i^t + c_1 \cdot [e^{-R(t+1)} + \text{rand}(x_g^t - x_i^t)] + c_2 \cdot [\text{rand}(x_c^t - x_i^t)] \quad (20)$$

$$x_C^t = \frac{\sum_{i=1}^{NP} x_i^t \cdot F(x_i^t)}{\sum_{i=1}^{NP} F(x_i^t)} \quad (21)$$

$$c_1 = \frac{F(x_i) - F(x_g)}{W - F(x_g)} \quad (22)$$

$$c_2 = 1 - c_1 \quad (23)$$

where the definitions and calculation methods of the elements in (20) are consistent with those in Formulas (13)–(17). To control the algorithm solely via the iteration parameter, we modify the basic PIO by removing its population halving operation. This modification involves no adjustments to convergence-related mechanisms and thus has no adverse impact on the algorithm's convergence performance. In Formula (22), $F(x_i)$ is the fitness function value of the i -th pigeon in the flock, $F(x_g)$ is the fitness function value of the leader pigeon, and W is the fitness function difference between the worst-positioned pigeon and the leader pigeon in the flock. After each round of position update, the pigeon with the optimal position is re-selected as the leader pigeon, and the center of gravity of the pigeon flock is recalculated. The optimization terminates when the iteration number t meets the termination condition Nc_2 . Figure 5 shows the complete flow chart of the MCOPIO algorithm.

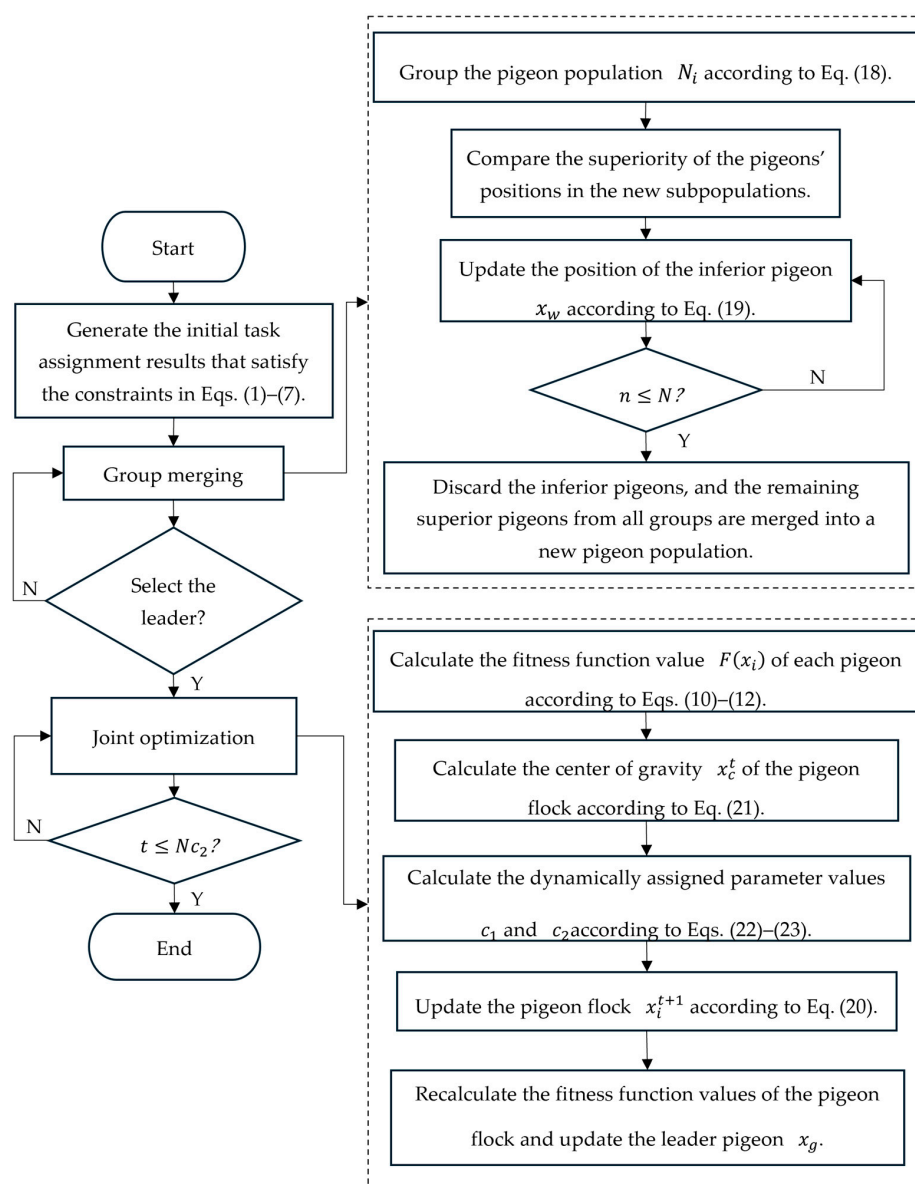


Figure 5. Flowchart of the MCOPIO algorithm.

5.3. Complexity and Convergence Analysis of the Proposed Algorithm

For computational complexity, let NP be the population size, D represent the number of tasks (dimension of decision variables), T be the maximum number of main iterations, and T' denote the iteration times of intra-group local optimization learning in the population merging stage. The population merging operation includes random pairwise grouping traversal, iterative intra-group local optimization learning and fitness calculation, which is only executed once in the initialization stage with a complexity of $O(NPDT'\log NP)$ since the population is iteratively reduced in $O(\log NP)$ rounds, and the elimination step only retains the better individual in each pair without introducing extra complexity. In the main iteration stage, the map-and-compass operator and landmark operator are executed T times, and the time cost of each iteration is $O(NPD)$, so the total complexity of the iterative optimization process is $O(TNPD)$. Accordingly, the overall time complexity of the MCOPIO algorithm is $O(NPDT'\log NP + TNPD)$. By omitting lower-order terms and constant coefficients in line with standard asymptotic complexity analysis, the asymptotic time complexity class of the algorithm is $O(T \cdot NP \cdot D)$, which is at the same level as the basic PIO, PSO and GA without introducing excessive additional computational overhead. The space complexity for storing the population and fitness values is $O(NPD)$, which fully meets the real-time requirements of UAV swarm dynamic task allocation.

The convergence of evolutionary algorithms is the core theoretical foundation to ensure their optimization effectiveness. This paper adopts the widely recognized classical convergence analysis framework in the field of evolutionary algorithms [28] to theoretically prove the proposed MCOPIO algorithm. Rudolph [28] proved the general convergence theorem for single-objective evolutionary algorithms through homogeneous finite Markov chain analysis: An evolutionary algorithm converges to the global optimal solution with probability 1 if and only if it satisfies the following two conditions: (1) It has an elitism mechanism that preserves the best solution found in history; (2) Its search operators satisfy ergodicity, i.e., they can generate any feasible solution in the search space with positive probability. This theorem is the basis for the convergence proof of all single-objective evolutionary algorithms, and the proposed MCOPIO algorithm fully satisfies the above two conditions.

First, MCOPIO has a strict elitism mechanism that permanently preserves high-quality solutions in both core stages. In the population merging stage, the algorithm randomly divides the population into pairs (x_m, x_n) , compares the fitness of the two individuals in each pair, retains only the better individual, and eliminates the worse one.

$$x_{\text{retain}} = \begin{cases} x_m, & c(x_m) \leq c(x_n), \\ x_n, & \text{otherwise,} \end{cases} \quad (24)$$

This process ensures that the overall fitness of the population is non-decreasing after each round of merging, and no solution better than the current individuals is lost. In the main iterative optimization stage, the fitness values of all individuals are recalculated in each iteration, and the global best individual (leader pigeon) x_g is updated.

$$x_g^{(t+1)} = \begin{cases} x_i^{(t+1)}, & c(x_i^{(t+1)}) \leq c(x_g^{(t)}), \\ x_g^{(t)}, & \text{otherwise,} \end{cases} \quad (25)$$

All subsequent iterations always use the current global best as the guiding information, and the global best individual is only replaced by a better one and never discarded. Therefore, MCOPIO can completely preserve all historical best solutions and satisfy the elitism condition.

Second, the search operators of MCOPIO satisfy ergodicity and can reach any position in the search space with positive probability. In the population merging stage, a random grouping strategy is adopted, and a random reversal factor ω (taking values 1 or -1 with equal probability) is introduced in the individual learning process, which can generate search steps in different directions with positive probability and avoid a single search direction.

$$P(\omega = 1) = P(\omega = -1) = \frac{1}{2} \quad (26)$$

In the main iteration stage, the position update formula (Equation (20)) contains two independent uniform random numbers $\text{rand}(\cdot) \in [0, 1]$, enabling individuals to move in any direction with any step size with positive probability. Therefore, the search operators of MCOPIO can generate any feasible solution with positive probability and satisfy the ergodicity condition.

In summary, MCOPIO satisfies both core conditions of elitism and search ergodicity. According to Rudolph's general convergence theorem [28], MCOPIO converges to the global optimal solution of the UAV swarm dynamic task allocation problem with probability 1.

6. Experimental Results

For all optimization algorithms involved in comparative experiments, including the proposed MCOPIO, basic PIO, PSO, and GA, a unified integer encoding scheme is adopted to represent the task allocation solution. Each individual is a vector with a length equal to the number of tasks, where each element represents the index of the UAV responsible for the corresponding task. All algorithms are implemented based on the same problem formulation and encoding rule to ensure the fairness and standardization of the comparative experiments.

To verify the feasibility of the MCOPIO algorithm, a model scenario is constructed as follows: The test task focuses on the problem of increasing the number of tasks in dynamic task allocation. In this paper, tasks are classified into rescue-type and monitoring-type based on their categories, where different types of tasks correspond to different levels of task complexity, and specific unmanned aerial vehicles (UAVs) can only execute specific types of tasks. The UAV parameter design is shown in Table 1, and the mission parameter design is shown in Table 2. Table 3 shows the parameter settings of different algorithms. All control parameters adopted for GA, PSO, ICBBA, PIO and the proposed MCOPIO are the classic and widely used default values reported in existing mainstream literature. Under such standard parameter configurations, the baseline optimization performance of all comparative algorithms is maintained at a similar level, which ensures a fair and credible comparison of algorithm performance. The calculation formulas for the inertia weights of PSO, PIO and MCOPIO are expressed as follows:

$$\omega = \omega_{start} - (\omega_{start} - \omega_{end}) \cdot \frac{iter}{max_iter} \quad (27)$$

$$R = R_{initial} \left(1 - \frac{t}{Nc_2}\right)^2 \quad (28)$$

Table 1. UAV Type and Parameter Settings.

UAV Type	Cruise Speed	Fuel Consumption
Rescue UAV	100 m/s	0.001/km
Surveillance UAV	70 m/s	0.01/km

Table 2. Mission Type and Parameter Settings.

Mission Type	Mission Completion Time	Task Deadline
Rescue mission	5 s	200 s
Surveillance mission	10 s	200 s

Table 3. Basic Parameter Settings of Different Algorithms.

Algorithm Name	Parameter Settings
GA	$NP = 30, max_gen = 50, pc = 0.8, pm = 0.05$
PSO	$NP = 30, max_iter = 50, c1 = 2, c2 = 2, \omega_start = 1, \omega_end = 0.4, \omega = \omega_start - (\omega_start - \omega_end) \cdot (iter / max_iter)$
ICBBA	$max_iter = 50, alpha = 1.0, beta = 1.0, \mu = 0.1, H = 300$
PIO	$NP = 30, max_iter = 50, Nc1 = 30, R_initial = 0.8$ $R = R_initial \times (1 - t / Nc2)^2$
MCOPIO	$NP = 30, max_iter = 50, Nc1 = 30, R_initial = 0.8$ $R = R_initial \times (1 - t / Nc2)^2$

Equation (27) is the linear decreasing inertia weight calculation formula of PSO, where ω_{start} and ω_{end} are the initial and terminal values of the inertia weight, and $iter$ and max_iter correspond to the current iteration number and the maximum iteration number respectively. Equation (28) is the dynamic search radius update formula for PIO and MCOPIO, in which $R_{initial}$ denotes the initial search radius, and t and Nc_2 represent the current iteration count and total iteration count, respectively. Such weight configuration enables fast convergence in the initial stage and more precise exploitation of the optimal solution in the subsequent stage.

First, the performance of MCOPIO is compared with PIO, PSO, and GA, on the standard Ackley test function. The iteration curves are shown in Figure 6 below. It can be seen that MCOPIO maintains a continuous downward trend throughout the iteration process and achieves satisfactory final optimization performance. This fundamentally verifies the feasibility of MCOPIO.

Second, the dynamic task allocation capability of the algorithm is simulated in a small-scale scenario with a small number of tasks. Since the scenario scope is small and the number of tasks does not exceed the total number of tasks that can be completed by UAVs of the corresponding types, all UAVs are able to finish the tasks before the task deadline. Therefore, the task deadline and task reward terms in the evaluation function can be temporarily ignored, and the task allocation performance of the algorithm can be directly evaluated by the energy consumption term in the evaluation function. The settings are as follows: the maximum task load of a single UAV is 4; the number of monitoring UAVs and rescue UAVs is three each; the initial number of tasks is 6, and the number of newly added tasks is 6, with an equal number of monitoring tasks and rescue tasks. The positions of UAVs and task locations are randomly generated.

Ten independent simulations were conducted for different algorithms under the same task environment. Figure 7 presents a bar chart of the penalty function values for each algorithm in each simulation, reflecting the performance of each run. Figure 8 shows a statistical chart of the ten simulation results for each algorithm, illustrating the mean value

and dispersion degree of the algorithm's performance. The simulation results indicate that, in small-scale scenarios with a limited number of tasks, although the optimal penalty value of MCOPIO across the ten independent simulations is slightly lower than that of GA and the original PIO, the overall results exhibit significant fluctuations due to the randomness in task generation and UAV positioning. This randomness does not negate the effectiveness of MCOPIO. MCOPIO achieves the minimum average penalty value with a more concentrated distribution of results, demonstrating its superior performance and stronger stability.

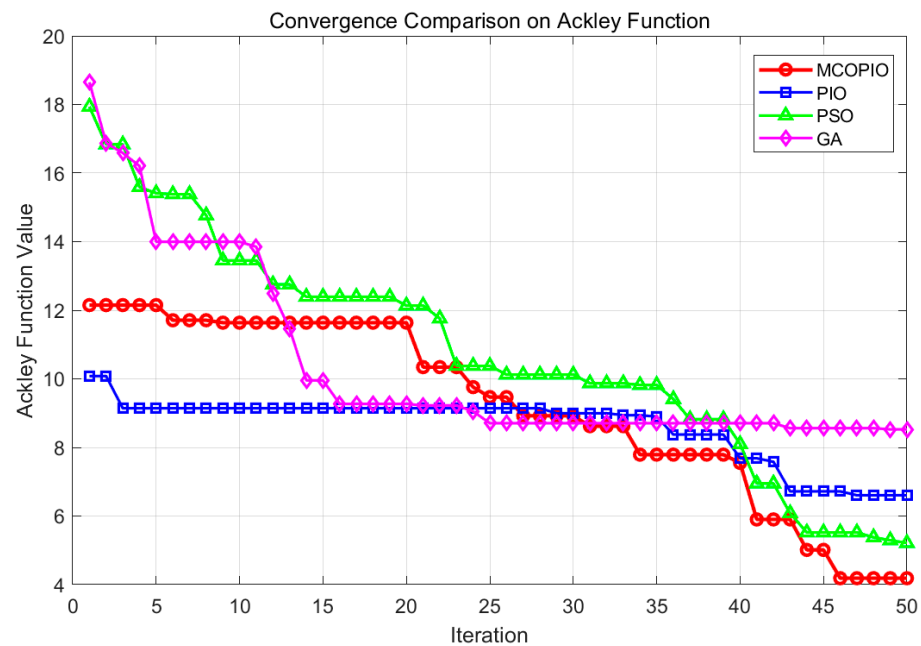


Figure 6. Iteration curves on the Ackley test function.

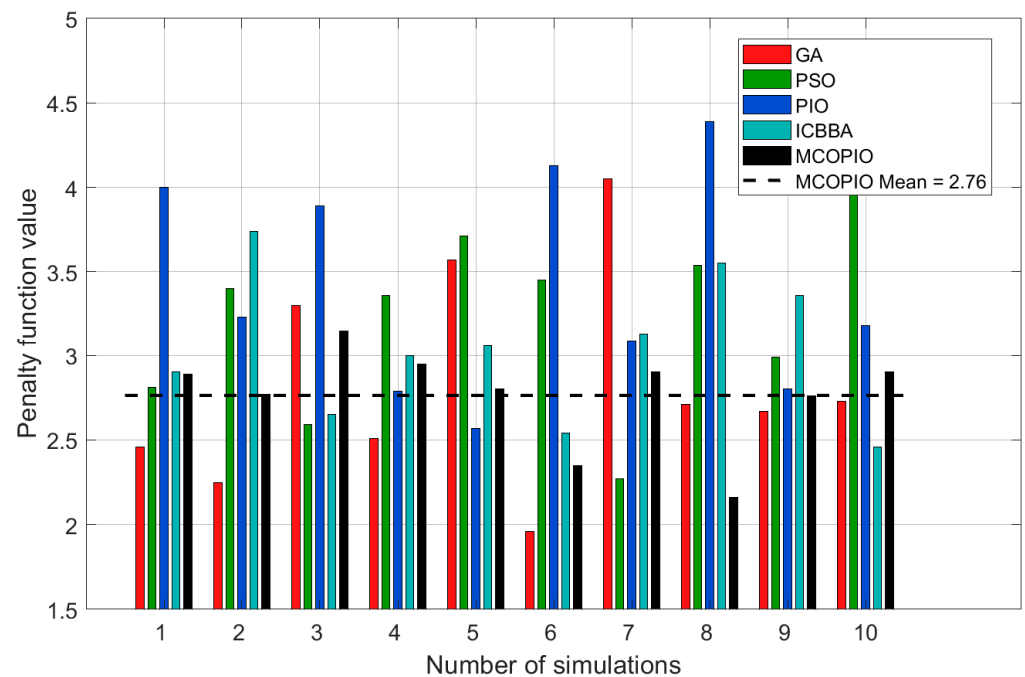


Figure 7. Simulation Result Bar Chart.

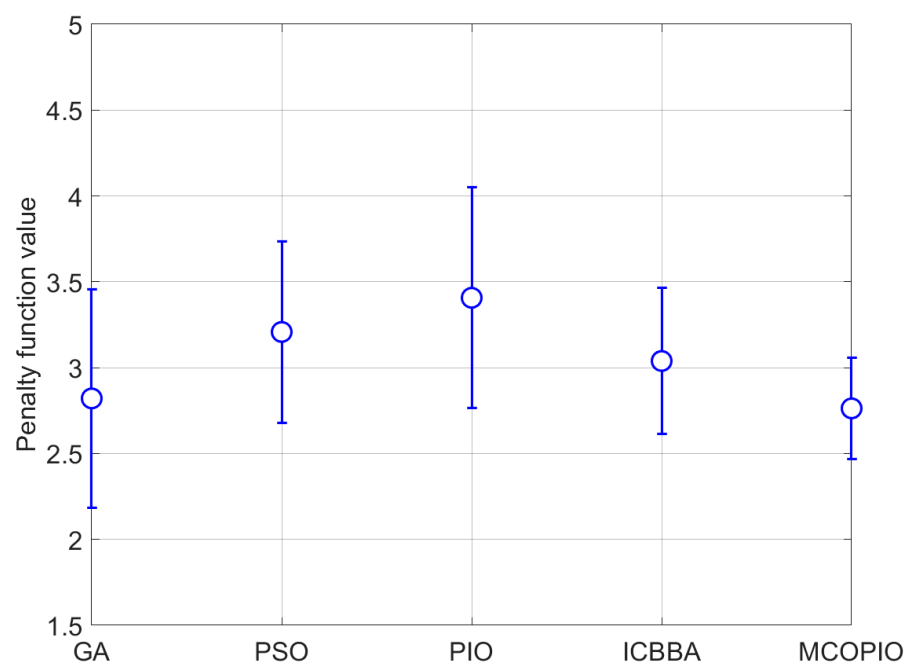


Figure 8. Statistical chart of algorithm task results.

Thirdly, we simulate the scenario where the number of tasks reaches the load upper limit of available UAVs. Assume the initial number of tasks is 12, the number of newly added tasks is 12, and the number of different types of tasks is equal. The final total scores are presented below, and the performance of each algorithm across ten independent simulations is consistent with that observed in the first set of simulations. The simulation data are shown in Table 4. The red values represent the optimal results, while the blue values denote the worst results.

Table 4. Table of total scores after ten independent simulations.

Number of Simulations	GA	PSO	ICBBA	PIO	MCOPIO	PSO Running Time	PIO Running Time	MCOPIO Running Time
1	2393.81	2393.63	2394.14	2394.67	2394.21	0.0235	0.0157	0.0161
2	2394.80	2393.93	2392.36	2394.98	2395.51	0.0194	0.0154	0.0149
3	2394.57	2394.24	2394.47	2394.28	2394.91	0.0194	0.0152	0.0153
4	2395.13	2394.80	2392.81	2393.66	2393.76	0.0180	0.0159	0.0163
5	2393.85	2394.78	2393.52	2394.74	2394.04	0.0195	0.0145	0.0153
6	2394.43	2395.07	2394.66	2394.16	2394.46	0.0161	0.0178	0.0159
7	2394.51	2394.15	2393.88	2393.64	2395.34	0.0185	0.0157	0.0163
8	2394.89	2394.13	2392.87	2393.07	2394.59	0.0204	0.0151	0.0186
9	2394.03	2395.08	2395.54	2393.25	2394.37	0.0179	0.0153	0.0161
10	2393.37	2393.94	2394.42	2393.51	2394.87	0.0174	0.0143	0.0150
Mean	2394.339	2394.375	2393.867	2393.996	2394.606	0.01901	0.01549	0.01598
Standard Deviation	0.5564	0.5160	0.9818	0.5794	0.5566			
p-Values (MCOPIO)	0.2833	0.3358	0.0384	0.0164				

The MCOPIO algorithm achieves the highest average total score of 2394.606 across ten independent simulation runs, ranking first among all comparative algorithms. It delivers a notable performance improvement over the original PIO algorithm (2393.996) and outperforms mainstream intelligent optimization algorithms including GA, PSO and ICBBA.

This fully verifies the effectiveness of the improvement strategies proposed in this paper for boosting the optimization capacity of the algorithm from an experimental perspective.

In addition, the standard deviation of MCOPIO's scores is 0.5566, which is comparable to that of classic optimization algorithms such as GA, PSO and PIO, and considerably lower than ICBBA's standard deviation of 0.9818. This demonstrates that MCOPIO exhibits minimal performance fluctuations across multiple independent random simulations, maintains stable optimization performance under diverse initial tasks and UAV deployment scenarios, and possesses strong robustness.

The paired t-test results reveal that the p -values for comparisons between MCOPIO and the original PIO, ICBBA, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO) are 0.0164, 0.0384, 0.2833, and 0.3358, respectively. In the evaluation function of this paper, the weight of penalty terms is far smaller than that of reward terms, with a 100-fold difference in magnitude, which narrows the numerical gaps between different algorithms. Even under such weight configurations that hinder the demonstration of algorithmic advantages, MCOPIO still exhibits statistically significant superiority over PIO and ICBBA, which further verifies the effectiveness of the improved algorithm proposed herein.

Furthermore, in terms of computational efficiency, the average running time of the proposed MCOPIO is 0.01598 s, which is very close to the original PIO (0.01549 s) and PSO (0.01901 s). The additional multi-strategy fusion modules introduced in MCOPIO do not bring a substantial rise in computational overhead, and the time complexity remains at the same level as the conventional PIO and PSO. This indicates that the performance gain of MCOPIO does not come at the cost of drastically prolonged solving time, and the proposed improved algorithm balances optimization accuracy, robustness and computational efficiency well, showing strong practical applicability for real-time dynamic UAV task allocation scenarios. Figure 9 shows the UAV path diagram and task schedule under this task environment.

Next, to further verify the performance of the MCOPIO algorithm, five independent simulations are conducted for nine groups of experiments, in which the number of UAVs is set to 6, 10, and 20, and the number of tasks is set to 1, 2, and 3 times the number of UAVs, respectively. The score of each UAV is observed, and the experimental data are presented in Table 5 below. The statistical curve graph is shown in Figure 10.

Table 5. Performance Scores Under Different Numbers of UAVs and Tasks.

Number of UAVs/Number of Tasks	1	2	3	4	5	Mean
6/12	1198.04	1196.39	1197.35	1197.17	1197.15	1197.22
6/18	1794.58	1796.83	1796.15	1796.71	1795.69	1795.99
6/24	2394.03	2394.48	2395.01	2394.11	2393.97	2394.32
10/20	1996.91	1996.09	1996.14	1995.83	1995.37	1996.07
10/30	2993.58	2993.48	2993.97	2994.25	2993.35	2993.73
10/40	3989.60	3989.63	3990.14	3989.85	3990.23	3989.89
20/40	3989.75	3990.19	3989.51	3990.46	3990.32	3990.05
20/60	5985.29	5985.91	5985.64	5984.01	5984.45	5985.06
20/80	7977.47	7977.66	7979.24	7978.49	7979.30	7978.43

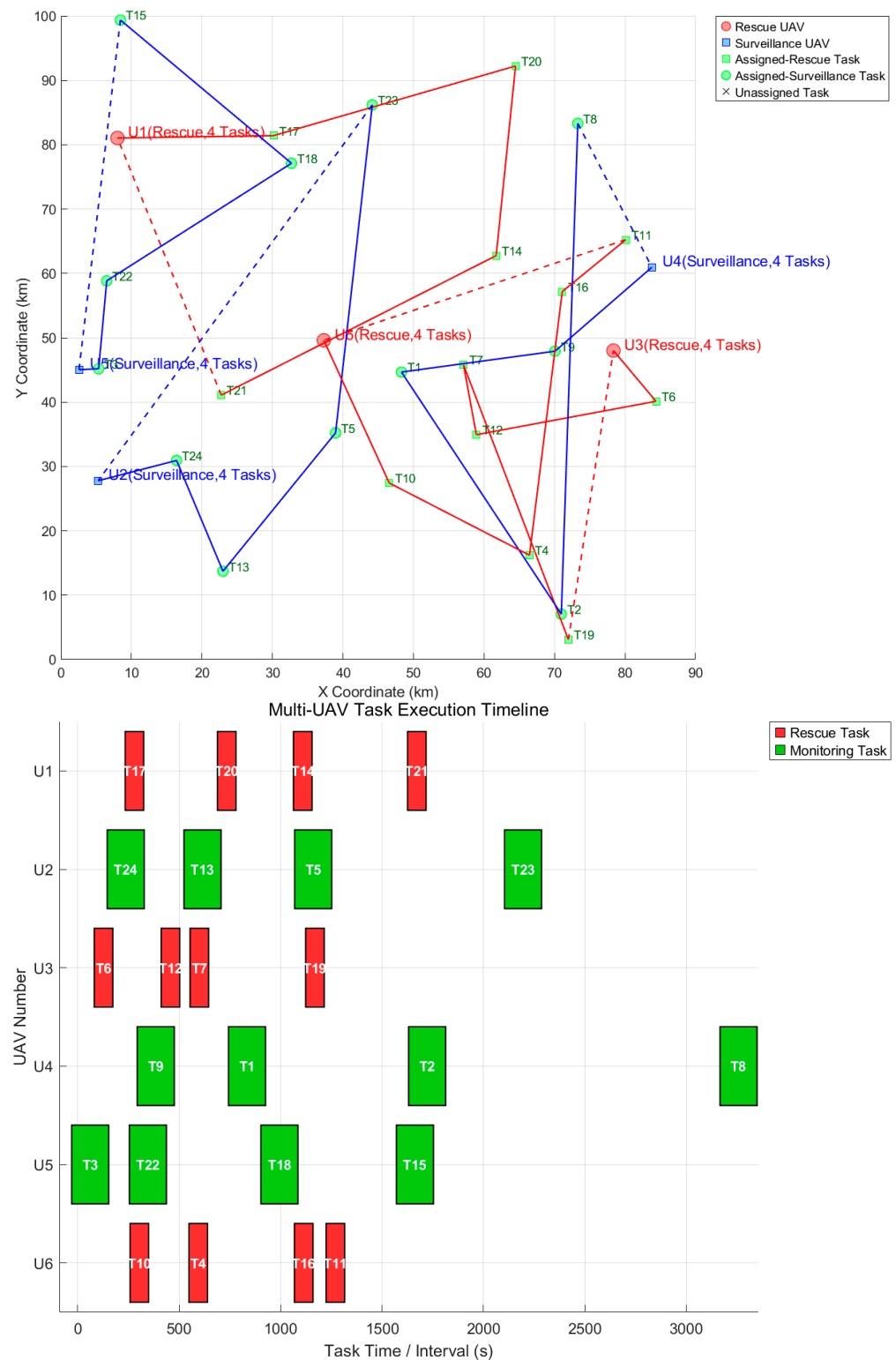


Figure 9. UAV scheduling scheme and trajectory diagram under dynamic task allocation.

As can be seen from the figure and table, under different numbers of UAVs, the score curve of the MCOPIO algorithm shows a steady growth trend as the number of tasks increases. Meanwhile, the data from multiple repeated experiments under the same scenario are close to each other, which reflects the robustness and adaptability of the algorithm, and demonstrates its stability and scalability.

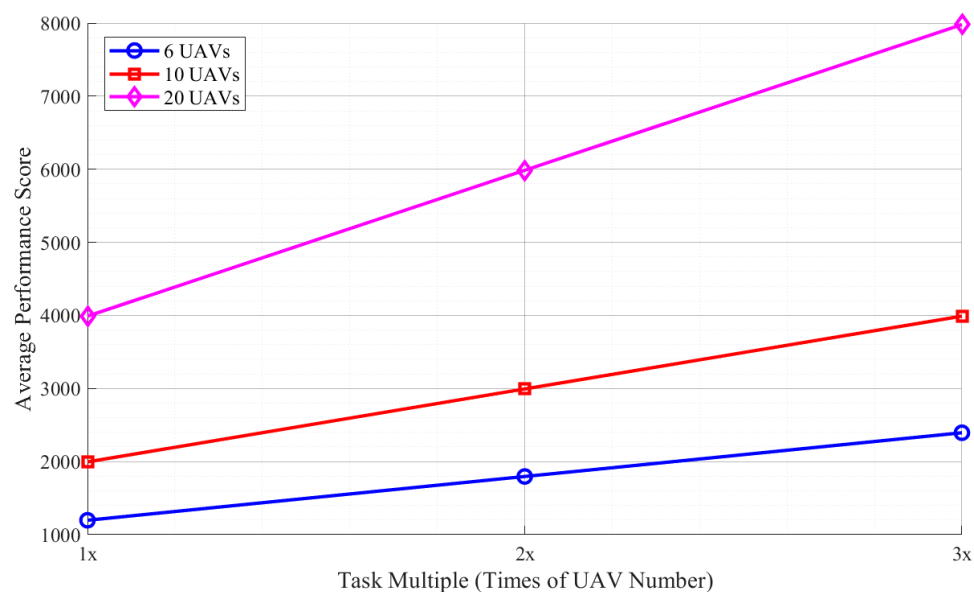


Figure 10. Distribution of Scores Under Different Numbers of UAVs and Tasks.

7. Conclusions

This paper targets the dynamic task assignment problem of UAV swarms with real-time emerging tasks. To obtain high-quality assignment schemes under time-varying task conditions, an improved PIO Algorithm named MCOPIO is proposed as the core solver. Aiming at the inherent defects of the original PIO, including weak global exploration, unstable optimization performance and disconnection between two search stages, the MCOPIO integrates population merging mechanism and dynamic adjustment operators to realize two-stage coordinated optimization, which effectively compensates the algorithm's native deficiencies and strengthens its adaptability to dynamic optimization scenarios.

On this basis, a UAV swarm dynamic task assignment model is established, where task execution efficiency and UAV energy consumption are defined as core optimization objectives. Comparative simulations covering small-scale and large-scale task environments are conducted with four mainstream metaheuristic algorithms: GA, PSO, original PIO and ICBBA. Experimental results verify that MCOPIO can generate superior task allocation results with remarkable stability. In addition, the proposed algorithm maintains consistent optimization performance when the number of tasks expands, showing outstanding scalability for variable-size UAV task scenarios. Common abbreviations used in this paper are listed in Table 6.

Table 6. List of Abbreviations.

UAV	Unmanned Aerial Vehicle
CMTAP	Cooperative Multiple Task Assignment Problem
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
PIO	Pigeon-Inspired Optimization
MCOPIO	Merged Coordinate-Optimized Pigeon-Inspired Optimization
NSGA-II	Non-dominating Sorting Genetic Algorithm II
CBBA	Consensus-Based Bundle Algorithm
ICBBA	Improved Consensus-Based Bundle Algorithm

Author Contributions: Conceptualization, Y.Z. and W.H.; methodology, Y.Z. and W.H.; software, Y.Z.; formal analysis, Y.Z. and W.H.; writing—original draft preparation, Y.Z.; writing—review and editing, W.H.; visualization, Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

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

References

1. Zhong, W. Research on Task Planning Methods for Agricultural Multi-UAV Systems. Master's Thesis, University of Electronic Science and Technology of China, Chengdu, China, 2025.
2. Yuan, H. Research on Key Technologies for Multi-UAV Swarm in Forest Fire Prevention and Control. Master's Thesis, University of Electronic Science and Technology of China, Chengdu, China, 2025.
3. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948.
4. Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning*; Addison-Wesley: Reading, MA, USA, 1989.
5. Choi, H.-L.; Brunet, L.; How, J.P. Consensus-based Decentralized Auctions for Robust Task Allocation. *IEEE Trans. Robot.* **2009**, *25*, 912–926. [[CrossRef](#)]
6. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [[CrossRef](#)]
7. Hu, W.; Ma, X. Optimization Algorithm of UAVs Task Assignment and Path Planning Based on Dynamic Cluster Particle Swarm Optimization. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 18157–18169. [[CrossRef](#)]
8. Liu, X.-F.; Fang, Y.; Zhan, Z.-H.; Zhang, J. Strength Learning Particle Swarm Optimization for Multiobjective Multirobot Task Scheduling. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 4052–4063. [[CrossRef](#)]
9. Chen, H.; Liu, Z.; Yang, X. Fuzzy Logic-based Adaptive Genetic Algorithm for UAV Path Planning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 5120–5131.
10. Gao, M.; Liu, Y.; Chen, J. Memetic Genetic Algorithm for Dynamic UAV Swarm Decision-making. *IEEE Trans. Cybern.* **2024**, *54*, 3890–3902.
11. Shen, Q.W. Research on Dynamic Task Allocation Strategy for Multi-UAV Collaboration. Master's Thesis, Guizhou University, Guiyang, China, 2025.
12. Bi, W.; Shen, J.; Zhou, J.; Zhang, A. Heterogeneous Multi-UAV Mission Reallocation Based on Improved Consensus-Based Bundle Algorithm. *Drones* **2024**, *8*, 345. [[CrossRef](#)]
13. Wang, X.; Zhao, J.; Li, S. Master-Slave Parallel NSGA-II for Real-Time Multi-UAV Task Allocation. *IEEE Access* **2023**, *11*, 123456–123467.
14. Liu, X.; Sun, J.; Zheng, L.; Wang, S.; Liu, Y.; Wei, T. Parallelization and Optimization of NSGA-II on Sunway TaihuLight System. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 975–987. [[CrossRef](#)]
15. Zhang, H.; Wang, Y.; Liu, Q. Parallelized A Posteriori Optimization with NSGA-II for Multi-UAV Cooperative Mission Planning. *IEEE Trans. Aerosp. Electron. Syst.* **2024**, *60*, 2145–2156.
16. Yu, H.; Wang, C.; Niu, Y.; Wu, L. MUTP-LLM: Empowering Multi-UAV Task Planning with Large Language Models. *Guid. Navig. Control* **2025**, *5*, 477–489. [[CrossRef](#)]
17. Duan, H.; Qiao, P. Pigeon-inspired Optimization: A New Swarm Intelligence Optimizer for Air robot Path Planning. *Int. J. Intell. Comput. Cybern.* **2014**, *7*, 24–37. [[CrossRef](#)]
18. Duan, H.B.; Huo, M.Z. *Pigeon-Inspired Optimization*; Science Press: Beijing, China, 2023.
19. Li, P.; Chen, G.F.; Gao, R.J. Research on Wireless Sensor Network Location Based on Improved Pigeon-inspired Optimization. In Proceedings of the 2019 IEEE/CIC International Conference on Communications Workshops in China (ICCC Workshops), Changchun, China, 11–13 August 2019.
20. Yuan, G.S.; Duan, H.B. Robust Control for UAV Close Formation Using LADRC Via Sine-powered Pigeon-inspired Optimization. *Drones* **2023**, *7*, 238. [[CrossRef](#)]
21. Zhu, D.L.; Shen, J.Y.; Hu, J.L.; Ouyang, Z.L.; Hu, G.Q.; Zhou, C.J.; Zhang, J.; Cheng, S. Group Merging Particle Swarm Optimization Algorithm for Rural Base Station Deployment. *IEEE Trans. Emerg. Top. Comput. Intell.* **2025**, in press.
22. Shima, T.; Rasmussen, S.J.; Sparks, A.G.; Passino, K.M. Multiple Task Assignments for Cooperating Uninhabited Aerial Vehicles Using Genetic Algorithms. *Comput. Oper. Res.* **2006**, *33*, 3252–3269. [[CrossRef](#)]

23. Xu, Z.L.; Hu, T.; Wang, S.; Kaiyue, L.; Yihui, Q.; Shenjian, Z.; Runze, H.; Wenjie, D. Heterogeneous Multi-UAV Collaborative Task Allocation Based on Modified Chimp Optimization Algorithm. *Command Control Simul.* **2024**, *46*, 13–20.
24. Khatib, O. Real-time Obstacle Avoidance for Manipulators and Mobile Robots. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 25–28 March 1986; pp. 500–505.
25. Yu, H.N.; Xi, W.Q.; Qi, F. UAV Path Planning Method Based on Ant Colony Mixed Potential Field Method. *Comput. Sci.* **2025**, *52*, 149–154.
26. Yang, W.; Sun, Z.; Qi, P.; Huang, H. Landing Path Planning for Obstacle Avoidance of Civil Aircraft. *Guid. Navig. Control* **2026**, *6*, 123–126. [[CrossRef](#)]
27. Wang, S.; Liu, Y.; Qiu, Y.; Li, S.; Zhou, J. An Efficient Distributed Task Allocation Method for Maximizing Task Allocations of Multirobot Systems. *IEEE Trans. Autom. Sci. Eng.* **2024**, *21*, 3588–3597. [[CrossRef](#)]
28. Rudolph, G. Convergence Analysis of Canonical Genetic Algorithms. *IEEE Trans. Neural Netw.* **1994**, *5*, 96–101. [[CrossRef](#)] [[PubMed](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.