drones

MDPI

Article

Distributed Task Allocation Algorithm for Heterogeneous UAVs Based on Reinforcement Learning

Peng Sun [1], Guangwei Yang [1,2], Xin Xu [1,*], Jieyong Zhang [1,*], Xida Deng [1], Yongzhuang Zhang [1] and Jie Cui [1]

[1] Information and Navigation College, Air Force Engineering University, Xi'an 710077, China; sunfly2182022@163.com (P.S.)
[2] The 942nd Hospital of Joint Logistics Support Force, Yinchuan 750001, China
* Correspondence: xuxin119go@163.com (X.X.); dumu3110728@126.com (J.Z.)

Highlights

What are the main findings?

- The proposed decentralized reinforcement learning algorithm, integrated with the PPO and a LSTM-attention fusion network, eliminates reliance on a global scheduler and addresses sparse reward instability, achieving 100% task success rate across diverse scenarios.
- The Cascaded Commitment Timeout (CCT) mechanism effectively prevents training deadlocks, while the fusion network captures temporal and collaborative dependencies, enabling zero-shot generalization and superior robustness against UAV failures and dynamic tasks.

What are the implications of the main findings?

- Theoretical Implication: This study enriches the family of distributed multi-agent task allocation methods, providing a viable framework for addressing deadlock and temporal dependency challenges in reinforcement learning-based scheduling.
- Practical Implication: The algorithm's high robustness and zero-shot adaptability make it suitable for real-world applications like low-altitude logistics and emergency rescue, supporting flexible scaling of UAV fleets and task volumes.

Abstract

To address the challenges faced by heterogeneous Unmanned Aerial Vehicle (UAV) systems in complex task allocation, including over-reliance on centralized scheduling, training deadlock, inadequate capture of temporal collaboration, and unstable training under sparse reward conditions, this paper proposes a distributed task allocation algorithm based on reinforcement learning. The algorithm adopts a decentralized decision-making architecture, which enables the autonomous formation of UAV collaborative groups without the need for a global scheduling center. A cascaded submission timeout mechanism is introduced to prevent training deadlock; the combination of Long Short-Term Memory (LSTM) and attention mechanism is employed to accurately model temporal correlations and collaborative dependencies; and the Proximal Policy Optimization (PPO) algorithm is leveraged to optimize the training stability under sparse reward conditions. Experimental results demonstrate that the proposed algorithm achieves a 100% task success rate in scenarios of different scales, and its key metrics, including makespan, time cost and waiting time, are significantly superior to those of mainstream baseline methods such as the Genetic Algorithm (GA) and the Hungarian Algorithm (HA). Moreover, the algorithm still

Academic Editor: Octavio Garcia-Salazar

Received: 6 February 2026
Revised: 9 March 2026
Accepted: 18 March 2026
Published: 20 March 2026

maintains excellent robustness under the conditions of UAV failures, parameter variations, and dynamic task perturbations. This method supports zero-shot generalization for any number of UAVs and tasks and provides an efficient and reliable solution for the real-time collaborative scheduling of heterogeneous UAV systems.

Keywords: UAV; task allocation; reinforcement learning; encode; Proximal Policy Optimization

1. Introduction

With the rapid development of the low-altitude economy and intelligent logistics, heterogeneous Unmanned Aerial Vehicle (UAV) systems demonstrate irreplaceable value in complex scenarios, such as search and rescue operations [1,2], environmental monitoring [3,4], and material delivery [5,6], by virtue of their superiorities, including strong collaborative capability, high fault tolerance, and excellent capacity for processing complex tasks. For instance, in search and rescue scenarios, reconnaissance UAVs locate survivors, relay UAVs maintain communication coverage, and delivery UAVs transport first-aid supplies; the collaboration of the three can significantly improve rescue efficiency [7]. In space exploration, multi-type UAVs can divide labor to complete environmental detection, sample collection, and data transmission tasks [8]. In medical scenarios, UAV teams can realize the coordinated operation of medication delivery and patient vital sign monitoring in remote areas [9]. The absence of any type of UAV in the above scenarios makes it impossible to carry out the tasks, which imposes stringent requirements on the task assignment and scheduling of UAVs. Such task assignment problems are typically modeled as the Generalized Assignment Problem (GAP) [10]. As a classic NP-hard problem, the scale of its solution space grows exponentially with the complexity of the problem, making it difficult to achieve optimal solutions.

In current research, centralized methods such as Mixed Integer Programming (MIP) [11] can obtain theoretical optimal solutions, but their computational time consumption grows exponentially with the problem scale, which fails to meet the requirements of real-time scheduling. Although heuristic algorithms [12] and distributed auction algorithms [13] feature high computational speed, these methods need to be constructed based on prior knowledge, and their inference processes rely on search and iteration. When solving large-scale problems, they suffer from low solution quality and poor efficiency, they are difficult to model complex collaborative dependencies, and they have limited generalization ability. Existing learning-based methods [14] eliminate the need to manually formulate rules based on prior knowledge; pre-trained models can solve problems in constant time, which greatly improves the solution efficiency. However, the application of learning-based methods to the task assignment of heterogeneous UAVs still faces the following challenges:

(1) Dependence on centralized decision-making: Most existing learning-based algorithms adopt a centralized architecture and require a global scheduling center to coordinate task assignment. This is prone to single point of failure, communication bottlenecks, and high latency in distributed environments, which cannot adapt to the demand for autonomous deployment of UAVs.

(2) Training deadlock problem: In multi-agent reinforcement learning (MARL), collaborative competition among UAVs is likely to lead to decision cycles or deadlock phenomena caused by resource contention. Especially in large-scale heterogeneous scenarios, the lack of effective distributed prevention mechanisms impairs the reliability and efficiency of the algorithm.

(3) Insufficient capture of temporal correlation and collaborative dependencies: Traditional neural networks are difficult to effectively model the temporal information in dynamic task sequences and the complex collaborative relationships among UAVs, resulting in poor generalization ability of the algorithm in variable-scale scenarios and failure to achieve zero-shot adaptation.

(4) Unstable training under sparse rewards: Learning-based methods have low sample utilization in sparse reward environments; the policy optimization process is susceptible to noise interference and features slow convergence, which cannot ensure the rapid generation of near-optimal solutions in large-scale problems.

To address the above challenges, this paper proposes a distributed task assignment algorithm for heterogeneous UAVs based on improved reinforcement learning. The algorithm adopts a decentralized decision-making architecture and does not require a centralized scheduling unit; it avoids training deadlocks through a cascaded submission timeout mechanism and captures temporal correlation and collaborative dependencies by combining the Long Short-Term Memory (LSTM) network with the attention mechanism; it achieves stable and efficient policy training based on the Proximal Policy Optimization (PPO) algorithm, ensuring the rapid generation of near-optimal solutions in large-scale scenarios.

In summary, the main contributions of this paper are as follows:

1. A decentralized reinforcement learning framework is designed, which allows UAVs to independently make decisions on task selection and collaborative group formation. It adapts to the requirements of distributed deployment and enables collaborative operation without a global scheduling center.
2. A cascaded submission timeout mechanism is proposed, which realizes fully distributed deadlock prevention through local dynamic countdown. This mechanism reduces the deadlock rate to an extremely low level without increasing communication overhead.
3. An LSTM + attention fusion network is constructed to simultaneously model UAV task collaborative dependencies and temporal correlations. It supports zero-shot generalization for an arbitrary number of UAVs and tasks, and it is adaptable to scenarios of different scales.
4. A training mechanism under sparse reward conditions is designed based on the PPO algorithm. Through the clipped probability ratio and multi-round sampling advantage estimation, the sample utilization efficiency and training stability are improved, ensuring efficient policy convergence.

The remainder of this paper is organized as follows: Section 2 reviews the related work; Section 3 describes the problem model and constraints in detail; Section 4 elaborates on the specific design of the proposed algorithm; Section 5 verifies the algorithm performance through experiments; and Section 6 concludes the full paper and prospects future work.

2. Related Work

This section reviews the research status of UAV task assignment problems and introduces the research on task assignment based on reinforcement learning.

2.1. UAV Task Assignment Problems

UAV task assignment is a core research direction of multi-agent cooperative systems, whose primary objective is to realize the optimal allocation and efficient scheduling of resources according to task requirements and the inherent capability characteristics of UAVs, thus ensuring the high-quality completion of tasks under given constraints. This research is widely applied to practical scenarios such as search and rescue [15], environmental monitoring [16], material delivery [17], agricultural production [18], and disaster response [19].

It needs to address complex challenges brought by task diversity, environmental dynamics, UAV heterogeneity, and collaborative dependence, and serves as a key link to improve the operational efficiency and application value of UAV swarms.

According to the number of UAVs required for task execution, UAV task assignment can be divided into single-UAV tasks and multi-UAV tasks. Optimization objectives are the core guide for task assignment, with common objectives including time optimization, resource optimization, efficiency optimization, and cost optimization. In practical scenarios, optimization objectives are often multi-objective coordination, which requires a trade-off among multiple objectives.

Based on the above core elements and classification system, the existing UAV task assignment methods have formed three mainstream technical routes, each with its own applicable scenarios and technical characteristics.

2.1.1. Mathematical Programming Methods

Mathematical programming methods aim at achieving the global optimum. By establishing a unified optimization framework to integrate all UAV and task information, they solve the resource allocation scheme and can achieve high solution quality in small and medium-sized scenarios. Mixed Integer Programming (MIP) [20] is a typical representative of mathematical programming methods, which can accurately model task decomposition, scheduling sequence, collaboration constraints, and various complex constraint conditions, as well as solve the global optimal solution through mathematical programming methods. Reference [21] studies the use of two Mixed Integer Linear Programming (MILP) models, where the first is used for the clustering problem and the second for the distribution route problem. Reference [22] adopts the Mixed Integer Nonlinear Programming (MINP) problem to reduce the energy consumption of multiple UAVs. As another centralized allocation method, Reference [23] proposes a fast energy-saving strategy based on the Hungarian Algorithm to minimize unnecessary idle time between phases.

2.1.2. Heuristic Methods

Heuristic and evolutionary algorithms [24] quickly solve the approximate optimal solution by simulating biological group behavior or iterative search mechanisms and balance efficiency and solution quality in moderately-sized scenarios. Such algorithms do not require the establishment of complex mathematical models; instead, they efficiently search for feasible solutions in the solution space by designing coding methods, fitness functions, and evolutionary operators suitable for the task allocation problem, thus avoiding the high computational complexity of traditional mathematical programming methods. Reference [25] explores the task allocation problem of heterogeneous UAV alliances in coordinated attacks on dynamic targets and proposes a Multi-Genotype Genetic Algorithm (MGGA) with customized crossover and mutation operators. While maintaining the overall alliance performance, MGGA effectively optimizes the task allocation of heterogeneous UAV alliances. Reference [26] proposes a Multi-Objective Multi-Swarm Adaptive Ant Lion Optimizer (MMSALO), which increases the randomness and diversity of ant activities around ant lions. The results show that MMSALO exhibits superior performance in solving multi-task, multi-constraint task allocation problems of UAV swarms. Reference [27] proposes a Multi-Discrete Wolf Pack Algorithm (MDWPA), which solves the UAV task allocation problem in complex environments by discretizing wandering, calling, besieging behaviors, and new individual supplement. The results show that the MDWPA performs excellently in terms of accuracy, robustness, and convergence rate.

2.1.3. Learning-Based Methods

Learning-based task assignment methods have gradually become a research hotspot in UAV task assignment by virtue of their data-driven decision-making advantages [28]. By training learning decision-making policies, such methods can quickly generate scenario-adaptive allocation schemes during the deployment phase, which greatly improves the real-time response capability and is particularly suitable for large-scale and dynamically changing complex scenarios. Their core advantages lie in strong generalization ability, which can adapt to unseen task and environmental scenarios, and fast decision-making speed that meets the requirements of real-time scheduling. Liu et al. [29] introduced the Specific Multi-Agent Deep Deterministic Policy Gradient (SMADDPG) algorithm for UAV task planning in 3D dynamic environments. In simulation experiments with six and 12 UAVs as the agent scale, the SMADDPG outperforms the Deep Deterministic Policy Gradient (DDPG) and Multi-Agent Deep Deterministic Policy Gradient (MADDPG) in terms of task completion efficiency, convergence speed, and success rate. It can effectively cope with dynamic situations and maintain a high task success rate in both static and dynamic environments. Dai et al. [30] explored the use of three methods—auction-based method, vacancy chain method, and Deep Q-Network (DQN)—for task assignment in multi-UAV exploration and destruction missions, aiming to minimize travel costs and improve task assignment performance. Ref. [31] developed an improved reinforcement learning algorithm to enhance the convergence accuracy of the algorithm, which introduces the transfer learning theory. After finding a similar UAV task assignment model in the policy library, the algorithm transfers the training parameter results of the previous source tasks to the new model through transfer learning.

In general, mathematical programming methods have advantages in solution quality but are insufficient in scalability; heuristic methods can quickly obtain approximate optimal solutions but tend to fall into local optima; learning-based methods show potential in large-scale scenarios but still need to solve the problems of collaborative modeling and generalization. Although existing studies have proposed various solutions for different scenarios, there is still room for improvement in aspects such as the formation of dynamic collaboration groups of heterogeneous UAVs, real-time scheduling in large-scale scenarios, and global optimization under complex constraints, which are also the core entry points of related research.

2.2. Task Allocation Based on Deep Reinforcement Learning

In recent years, Deep Reinforcement Learning (DRL) [32] has demonstrated remarkable advantages in complex task assignment problems, especially in scenarios with dynamic changes, uncertainties, and high-dimensional state spaces. By enabling agents to learn optimal policies through interaction with the environment, DRL performs exceptionally well in UAV task assignment and resource management problems, achieving real-time policy adjustment, optimized resource utilization, and collaborative decision-making without relying on handcrafted rules or prior models. Compared with traditional optimization methods, DRL can more effectively handle complex factors such as fluctuations in task priorities and environmental interferences in dynamic heterogeneous systems, thereby improving task completion efficiency and system performance.

Liu et al. proposed a dynamic task planning method with a two-layer optimization mechanism, DAWN [33]. This method adopts DRL to solve the problem of minimizing the global flight path and energy consumption and introduces a trust network into local path planning to balance regional coverage and energy consumption. Eser et al. proposed a gossip consensus-based auction algorithm, Harmony DTA [34]. This algorithm integrates a two-stage auction mechanism with DRL and realizes consensus among agents

through the gossip protocol, minimizing the total system cost and task execution delay in communication-constrained environments. Li et al. proposed a DRL framework based on an encoder–decoder architecture [35]. This framework reconstructs the traditional Pickup and Delivery Problem (PDP) into a Capacitated Multi-depot Open-loop Pickup and Delivery Problem (CMOPDP) with heterogeneous starting points, explores the constraint relationships among nodes via a heterogeneous attention mechanism, maximizes the matching degree between UAVs and task nodes using a dual-decoder structure, and introduces an entropy reward to enhance the exploration capability and prevent the model from falling into local optima. Abou Houran et al. modeled task assignment as a Capacitated Vehicle Routing Problem (CVRP) [36]. This method employs an attention model as the core architecture of DRL, generates optimized paths through an encoder–decoder structure, and trains the model using a policy gradient algorithm, with the objective of minimizing the maximum travel distance of all UAVs. Yan et al. proposed a method combining hierarchical coalition formation game with DRL, MCC-RMCF [37]. This method first decomposes large-scale UAV swarms into sub-swarms through Multidimensional Contribution Clustering (MCC) to reduce decision-making complexity and then adopts an Overlapping Coalition Formation (OCF) game model combined with the marginal utility preference criterion and random exit mechanism to realize the distributed task assignment. Wan et al. unified the modeling of task assignment and path planning as a Markov Decision Process (MDP) problem [38]. Based on multi-agent reinforcement learning, this method designs individual and collaborative reward mechanisms to balance the exploration and exploitation behaviors of agents and optimizes the jamming effect of directional antennas in scenarios with uncertain target positions and existing no-fly zones.

3. Problem Formulation

For the researched heterogeneous multi-UAV task allocation problem, this paper constructs a decentralized sequential decision-making model supporting global communication. Heterogeneous UAVs of different types are deployed from their respective initial bases of belonging categories and select subsequent task targets via an autonomous decision-making mechanism after the termination of current task execution. This model operates continuously with the decision–execution–feedback cycle as the basic unit until the completion of the global task cycle and the return of all agents to their corresponding initial bases; ultimately, each agent generates a unique task path.

To ensure the collaborative efficiency among UAVs, a sequential decision-making mechanism based on historical interaction information is designed in this paper: when a UAV conducts the next round of task selection, it can retrieve in full the historical action trajectory data of other UAVs. The specific implementation logic is as follows: after a UAV determines its subsequent task, it broadcasts two core pieces of information to all other agents through the global communication link—its own target coordinates and the remaining execution requirements of the task. The termination of the execution of a certain task will trigger an update of the decision phase, at which point the UAV coalition participating in the task collaboration must synchronously initiate the next round of the task screening process.

3.1. Heterogeneous UAV System Model

This paper introduces the type-feature model proposed in Reference [39] as the core characterization framework for the heterogeneous UAV system. The core logic of this model is that the heterogeneous UAV system (HUS) consists of multiple types of heterogeneous UAVs, and each type of UAV is equipped with an exclusive skill feature vector

to accurately characterize its inherent skill endowments and capability boundaries. The specific definitions are as follows:

Definition 1. *UAV type set: Let the UAV type set be* $K = \{k_1, k_2, \ldots, k_m\}$, *where* m *denotes the number of UAV types.*

Definition 2. *UAV set: Let the UAV set be* $U = \{u_1, u_2, \ldots, u_n\}$, *where* n *denotes the total number of UAVs. For the* i*-th type of UAV* k_i, *it contains* n_i *UAVs, satisfying* $n = \sum_{i=1}^{m} n_i$.

Definition 3. *Single UAV skill characterization: UAVs of the same type,* k_i, *have identical feature vectors. For any UAV,* u_j, *in type* k_i, *its feature vector is* $v_{u_j} = v_{t_i} = \left[v_{1j}, v_{2j}, \ldots, v_{dj}\right]$, *where* $d \in \mathbb{Z}^+$ *represents the number of unique skills (corresponding to specific functions such as reconnaissance, relay, and delivery), and* $v_j \in \mathbb{N}$ *denotes the capability value of the UAV for the corresponding skill—1 indicates possession of the skill, and 0 indicates non-possession.*

When multiple UAVs form a cooperative group, the comprehensive skill feature vector, v_c, of the swarm is obtained by performing an element-wise summation operation on the skill feature vectors of all UAVs in the group. The core function of this comprehensive vector is to quantify the overall skill supply capacity of the collaborative swarm, providing a quantitative basis for the matching verification of task skill requirements.

3.2. Task and Environment Definition

3.2.1. Spatial and Position Model

Base station: the initial and final return positions of all UAVs are the base stations corresponding to their affiliated types. Let the base station set be $B = \{b_1, b_2, \ldots, b_m\}$, where b_i denotes the position of the base station for the i-th type of UAV.

Task set: the task set is denoted as $T = \{t_1, t_2, \ldots, t_p\}$, where p represents the total number of tasks. All tasks are spatially distributed in a normalized 2D region $[0,1] \times [0,1] \subset \mathbb{R}^2$.

The positions of base stations and tasks are all represented by coordinates (x_i, y_i), where $i \in B \cup T$.

3.2.2. Task Requirement Model

Each task, t_j, is associated with two core parameters:

Feature requirement vector: $r_{t_j} = \left[r_{1j}, r_{2j}, \ldots, r_{dj}\right]$, where $r_j \in \mathbb{Z}^+$, denotes the minimum skill capability required to complete the task.

Execution duration: $q_{t_j} \in \mathbb{R}^+$ represents the continuous time required to complete the task after all UAVs of the cooperative group arrive at the task location.

The necessary and sufficient conditions for task initiation are:

The feature vector of the cooperative group, C, meets the task requirements, i.e., $v_c \geq r_{t_j}$.

All UAVs in the cooperative group have arrived at the task location and remain on site for the entire task execution duration.

3.2.3. Graph Structure Modeling

To simplify the spatial correlation and state description of tasks and UAVs, two types of graph structures are defined:

Task-Base Station Graph, $G_s = (V_s, E_s)$: the vertex set, $V_s = B \cup T$, covers all base stations and tasks; the edge set, E_s, contains connections (v_i, v_j) between all vertices $(\forall v_i \neq v_j, v_i, v_j \in V)$, and the edge weight is the Euclidean distance between two vertices, which is used to calculate the UAV flight time.

UAV State Graph, $G_u = (V_u, E_u)$: the vertex set, V_u, consists of all UAVs; the edge set, E_u, contains connections between all UAVs, and the edge attributes are state information such as relative positions, skill complementarity, and historical interaction records among UAVs, which are applied to decision-making for cooperative group formation and temporal dependency modeling.

3.3. UAV State and Objective Function

3.3.1. State Definition

This paper designs that each UAV is always in one of the following three states:

Waiting state: arrived at the task location and waiting for other UAVs of the cooperative group to arrive for task initiation.

Execution state: participating in task execution and satisfying the task initiation conditions.

Flight state: flying from the current task location/base station to another task location/base station.

The full task cycle of each UAV is defined as the cumulative duration from departing from its affiliated base station to completing all assigned tasks and returning to the base station. In view of the temporal dependency characteristics of UAV decision-making (e.g., historical flight trajectories and past collaboration records affect the current task selection strategy), the model input adopts a complete state sequence instead of an instantaneous state at a single moment.

3.3.2. Objective Function

The optimization objective of this paper is to minimize the makespan, i.e., the maximum value among the full task cycles of all UAVs. Let the flight route of each UAV, u_i, be $\phi_{u_i} = (b_t, t_1, t_2, \ldots, b_t)$, where b_t is the base station of the type to which the UAV belongs, and $t_1, t_2, \ldots$ are the tasks it participates in executing. The objective function is shown in Equation (1):

$$\min_{\Phi} \max_{u_i \in U} \text{TotalTime}(\phi_{u_i}) \tag{1}$$

In Equation (1), $\Phi = \{\phi_{u_1}, \phi_{u_2}, \ldots, \phi_{u_n}\}$ denotes the set of flight routes for all UAVs, and $\text{TotalTime}(\phi_{u_i})$ represents the total operating time of UAV, u_i, along the flight route ϕ_{u_i}.

4. Proposed Method

4.1. Reinforcement Learning Task Allocation Framework Based on Decentralized Decision-Making

This paper proposes a reinforcement learning architecture based on decentralized decision-making to tackle the heterogeneous UAV task assignment problem. With the PPO algorithm as its core training framework, this architecture consists of two core network modules: the policy network is responsible for outputting the probability distribution of action selection, and the value network undertakes the fitting and estimation of the value function. To improve training stability and convergence efficiency, both networks incorporate the clipped importance sampling mechanism and entropy regularization technique.

The architecture realizes autonomous collaboration and task assignment of UAVs through a collaborative mechanism of asynchronous decision-making and global state sharing: no centralized scheduling center is required, and each UAV acts as an independent decision-making unit, sharing key state information via the global communication link to independently complete task selection and the formation of collaborative clusters. In the training phase, the training convergence problem in sparse reward environments is addressed by virtue of the clipped objective function and multi-round sampling advantage

estimation method of the PPO algorithm. In the execution phase, UAVs asynchronously trigger the decision-making process based on the progress of task execution and adapt to the differentiated skill requirements of tasks through the formation and disbandment of dynamic collaborative clusters, ultimately achieving the optimization objective of minimizing the maximum task cycle.

The core workflow of the architecture is as follows: in the initialization phase, all UAVs are deployed at the dedicated base stations corresponding to their respective types; after a UAV completes its current task or takes off from the base station, it independently triggers the next round of decision-making for task selection; upon the completion of decision-making, the UAV immediately broadcasts its target position, the remaining requirements of the selected task and its historical state sequence to the entire domain; the asynchronous decision-making mode is adopted, and an asynchronous decision-making pattern is naturally formed due to differences in task execution duration, with a random permutation strategy applied to the decision-making order of UAVs within the same collaborative cluster; finally, the decision-making process is terminated when all tasks are completed and all UAVs return to their respective positions.

4.1.1. Observation Space Design

Considering the temporal dependency characteristics of UAV decision-making, the observation of UAV at time, t, is defined as a temporal state sequence, $s_t^u = \{T_{t-L+1}^u, U_{t-L+1}^u, M_{t-L+1}^u\}, \ldots, \{T_t^u, U_t^u, M_t^u\}$, where L is the temporal window length (set to 6 in this paper), covering the state information of the most recent six steps. The observation at each step consists of three parts: task features, UAV interaction features, and decision mask:

(1) Task features, $T_t^u \in \mathbb{R}^{(p+1)\times d_1}$ have a dimension of $d_1 = 5 + 2d$ (d is the number of skills). Each row corresponds to the state of a task or a base station, specifically expressed as $\left[r_{tm}, \Delta x_{tm}, \Delta y_{tm}, t_{execm}, t_{flightm}, done_m\right]$, where $r_{tm} = r_m - v_{Ct}$ denotes the remaining skill requirement of task m_i, with r_m being the initial requirement and v_{Ct} the skill supply of the current collaborative swarm; $\Delta x_{tm} = x_m - x_u$, $\Delta y_{tm} = y_m - y_u$ are the relative coordinates between the task and the UAV; $t_{flightm}$ is the estimated flight time for the UAV to reach the task; and $done_m \in \{0, 1\}$ indicates whether the task is completed (1 for completed). All parameters, such as remaining requirements and execution duration corresponding to base stations, are set to 0, with only spatial coordinate information retained.

(2) UAV features, $U_t^u \in \mathbb{R}^{n\times d_2}$ have a dimension of $d_2 = d + 6$. Each row corresponds to the state of a UAV u_j relative to the observing UAV u_i, specifically expressed as $\left[v_j, t_{remaining}, \Delta x_{j,u}, \Delta y_{j,u}, task\_status_j\right]$. Among them, v_j denotes the skill feature vector of UAV u_j with a dimension of d, corresponding to the quantitative capability values of skills such as reconnaissance and relay; $t_{remaining} \in \mathbb{R}^{1\times 3}$ is the remaining state time vector of UAV, u_j, which is a vector of the remaining execution time, remaining flight time, and waiting time in sequence; $\Delta x_{j,u}$ is the relative coordinate of UAV, u_j, relative to the observing UAV, u, on the *x*-axis, calculated as $\Delta x_{j,u} = x_{uj} - x_u$; $\Delta y_{j,u}$ is the relative coordinate of UAV, u_j, relative to the observing UAV, u, on the *y*-axis, calculated as $\Delta y_{j,u} = y_{uj} - y_u$; $task\_status_j \in \{0, 1\}$ indicates the current task state associated with UAV, u_j, where 0 means open and 1 means that the task is in execution.

(3) Decision mask, M_t^u, is a binary mask indicating whether a task is open to the agent, subject to decision constraints.

4.1.2. Action Space

Let the parameterized decentralized neural network be denoted as θ, which outputs the action probability distribution, $\pi_\theta(a|s_t^u) = \pi_\theta(\tau_t = p|s_t^u)$, based on the temporal observation sequence, s_t^u, of the UAV, where $j \in \{0, 1, \ldots, P\}$ represents action options, with 0 corresponding to returning to the base station and 1 to P corresponding to each task node respectively; completed tasks, ongoing tasks, and filtered tasks are all excluded from the action space through the decision mask, M_t^u; a random sampling strategy is adopted to extract actions from the probability distribution in the training phase; in the inference phase, a greedy strategy can be used to select the action with the maximum probability or a Boltzmann weighted random strategy to balance exploration and exploitation.

4.1.3. Reward Function

To optimize both the makespan and skill waste simultaneously, a sparse reward function is designed in this paper, which is only calculated at the end of the training episode and expressed as Equation (2):

$$R(\Phi) = -T - W \tag{2}$$

In Equation (2), T denotes the makespan; W is the Average Ability Wastage Rate, calculated as shown in Equations (3) and (4):

$$W = \frac{1}{P}\sum_{i=1}^{P} w_i \tag{3}$$

$$w_i = \begin{cases} \frac{\sum_{j=1}^{d} \left| v_c^{(j)} - r_{t_i}^{(j)} \right|}{\sum_{j=1}^{d} r_{t_i}^{(j)}}, & \text{if } v_c \geq r_{t_i} \\ \eta, & \text{otherwise} \end{cases} \tag{4}$$

In Equation (4), v_c is the feature vector of the collaborative group executing task, t_i, and $\eta = 10$ is set to balance the magnitude scale of T and W. In addition, a training timeout threshold, $T_{limit} = 100$, is set: if the episode is not completed within the timeout, it is judged as a training failure, and the current episode is terminated.

4.2. Cascaded Commit Timeout Mechanism

In the research scenario of this paper, tasks dynamically switch among four states: the idle state (no UAVs are at the task location or en route to it), the incomplete state (the UAV cluster at the task location fails to meet the skill requirement threshold), the execution state (the skill supply of the UAV cluster meets the standard and the task proceeds normally), and the closed-loop state (the task is fully completed). The core requirement of minimizing the maximum task cycle dictates that UAVs process multiple tasks in parallel as much as possible. However, it is found in the actual training that UAVs are prone to decision bias, excessively pursuing the maximization of the number of parallel tasks and blindly initiating a large number of tasks without sufficient UAVs with corresponding skills to form effective collaborative clusters. Such extremely decentralized behaviors often trigger system deadlocks, resulting in the abnormal termination of training episodes, the failure of the team to obtain valid training experience, and severe damage to the stability and convergence of the training process.

To completely eradicate training deadlocks caused by excessive task decentralization and meanwhile avoid the centralized dependence on the number of globally open tasks, this paper proposes a fully distributed deadlock prevention scheme—the cascaded submission timeout mechanism. The core idea of this mechanism is as follows: after a

UAV selects and announces the execution of a certain collaborative task, it immediately enters the pre-submission phase and initiates a local dynamic countdown, τ_i, for itself. If all the required heterogeneous UAVs for the task are successfully gathered before the countdown ends, the task is officially submitted, and its execution starts; otherwise, the UAV voluntarily abandons the current task and re-enters the task selection phase. Since each UAV makes timeout judgments only based on local information, no central coordination node is required for the entire mechanism, thus achieving true zero-communication deadlock prevention.

The specific process of the mechanism is as follows:

1. When a UAV, u, selects a task, T_j, at time, t, it immediately broadcasts a statement to the environment that it has pre-occupied a position for the task and initiates a local timer:

$$\tau_i = \alpha \times d_i + \beta \times (k - 1) \times t_{hist} \quad (5)$$

In Equation (5), d_i is the estimated flight time of UAV i to the task location, t_{hist} is the historical average gathering time for this type of task, and α = 1.5, β = 1.2 are empirical coefficients.

2. Within the time window, $[t, t + \tau_i)$, if task, T_j, successfully gathers k UAVs with corresponding capabilities, all relevant UAVs commit simultaneously and the task is officially executed.

3. If any capability is still missing when the timer expires, UAV i immediately performs a local rollback: it cancels the pre-occupation of T_j, receives a small timeout penalty reward, $r_{delay} = -0.2$, and re-enters the task selection phase.

4. To further accelerate convergence, the timeout penalty is set to -0.5 (a relatively large value) in the early stage of training and decays exponentially to -0.1 with the number of training rounds.

The cascading effect of the CCT mechanism is reflected in the following: once a UAV is the first to abandon the task pre-lock due to timeout, other UAVs waiting for the same task will successively trigger timeouts in subsequent steps due to the lack of collaborative swarm members, thus quickly breaking down potential deadlock loops. Theoretically, the CCT mechanism can naturally constrain the number of tasks in the pre-lock state at the same time in the system within the $O(n)$ range (n is the total number of UAVs), and this is realized entirely through local decision-making without any global coordination. The proof is given as follows:

Let the total number of UAVs be n. Each UAV acts as an independent decision-making unit and can only initiate a pre-lock on one task at any time instant.

Let N_p denote the number of tasks in the pre-locked state simultaneously in the system. A necessary condition for a task to enter the pre-locked state is that at least one UAV sends a pre-lock request to it.

Based on the above two premises, the bound of the number of pre-locked tasks, N_p, is derived as follows:

From the condition that each UAV can pre-lock only one task, it can be concluded that n UAVs can initiate at most one pre-lock request for each of n different tasks at the same time.

From the necessary condition that a task can be pre-locked only if at least one UAV initiates a pre-lock on it, each pre-locked task occupies the pre-lock resource of at least one UAV.

Therefore, the number of simultaneously pre-locked tasks in the system must not exceed the total number of UAVs, satisfying the core constraint: $N_p \leq n$.

According to the standard definition of algorithmic complexity: if the upper bound of a variable is a linear function of the problem size, the scale constraint of the variable is $O(n)$, where n denotes the core problem size.

In this paper, the total number of UAVs, n, is the core size of the heterogeneous UAV task assignment problem. The above derivation has proved that the strict upper bound of the number of pre-locked tasks, N_p, is n, i.e., there exists a constant, $c = 1$, such that $N_p \leq c \cdot n$ holds for all UAV numbers, n.

Thus, it can be proved that the CCT mechanism constrains the number of simultaneously pre-locked tasks in the system to the $O(n)$ level.

4.3. Network

This paper designs a fusion network intended to capture the complex interactive relationships between heterogeneous UAVs and tasks, as well as the temporal dependencies in the dynamic decision-making process, enabling UAVs to generate a task assignment policy, $\pi_\theta(a|s_t)$, from a global perspective—where s_t denotes the system state at time, t, and a represents the task selection action—and simultaneously evaluate the value, $V_\phi(s_t)$, of the current state. Adhering to the core logic of unified feature processing, temporal dependency capture, and task-specific output, this network is structured as a three-stage architecture with a shared backbone and dual-branch decoding. The shared backbone network serves as the common feature extraction foundation for the Actor and Critic branches, consisting of two components: a task–UAV context encoder and an LSTM Temporal Modeling Unit. It undertakes the task of extracting fusion features from raw data and capturing temporal dependencies in decision-making. Subsequently, the dual-branch structure of the decoder is employed to realize the generation of task assignment policies and the evaluation of current state value, respectively. The consistency and reusability of the preceding features are maintained throughout the process, allowing the network to handle an arbitrary number of UAVs and task instances in the inference phase. Endowed with generalization ability for dynamic task environments, the network is suitable for the practical deployment scenarios of heterogeneous multi-UAV task assignment. The overall network structure is illustrated in Figure 1.

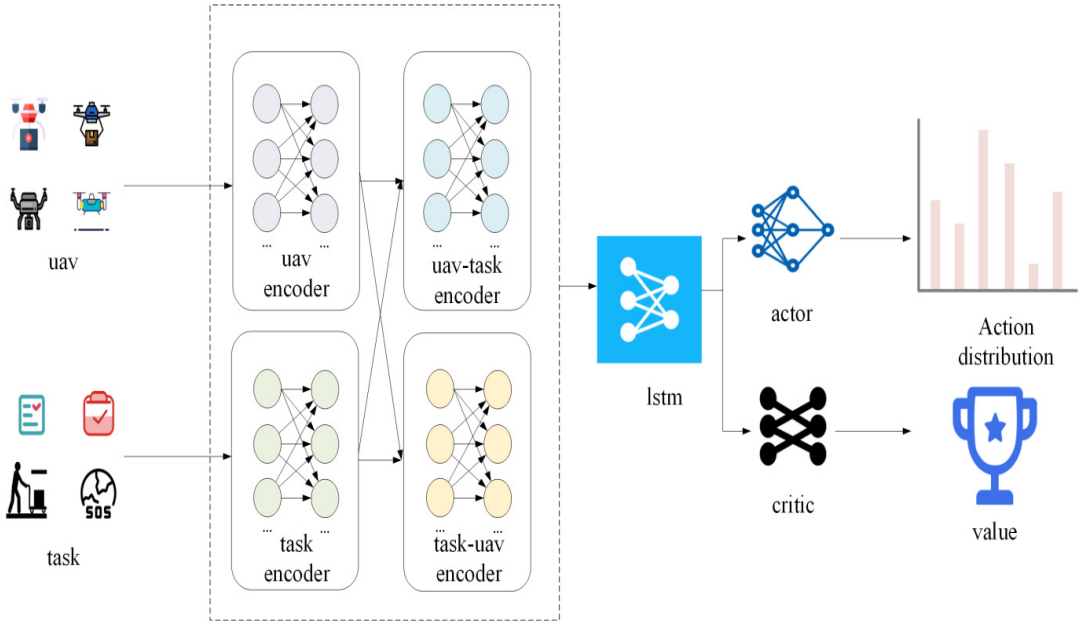

Figure 1. The overall network structure of the model.

4.3.1. Multi-Head Attention with Gated Units

The basic component of the network used in this paper is a multi-head attention layer with a gating mechanism; for the input query vector, h_q, and key-value pairs, (h_k, h_v), they are first mapped to a high-dimensional space through three learnable linear transformation matrices, W_Q, W_K, and W_V, to obtain the query matrix, Q, key matrix, K, and value matrix, V:

$$Q, K, V = W_Q h_q, W_K h_k, W_V h_v$$

In a single attention head, z, the calculation of the attention vector, α_z follows the rule of scaled dot-product attention, which multiplies the value matrix with the weights normalized by the Softmax function, as shown in Equation (6):

$$\alpha_z = \text{Attention}(Q_z, K_z, V_z) = \text{Softmax}\left(\frac{Q_z K_z^T}{\sqrt{d}}\right) V_z \quad (6)$$

In Equation (6), d denotes the feature embedding dimension, set to 128 in this paper, which is used to alleviate the problem of excessively large dot-product results in high-dimensional spaces. Multi-head attention concatenates the outputs of Z independent attention heads and completes feature fusion through another learnable matrix, W_O, to obtain the final attention representation, as shown in Equation (7):

$$\text{MHA}(h_q, h_k, h_v) = \text{Concat}(\alpha_1, \alpha_2, \ldots, \alpha_Z) W_O \quad (7)$$

In Equation (7), Z is the number of attention heads, set to 8 in this paper; different attention heads can capture feature correlations of different dimensions in the input sequence. Finally, the output vector undergoes layer normalization and is fed into a feed-forward layer with gated linear units to further enhance the nonlinear expression ability of features.

4.3.2. Encoder

This paper constructs a shared backbone network through an encoder, which consists of three parts: a task encoder, a UAV encoder, and a cross-encoder. Its core objective is to construct the local context of tasks and UAVs, as well as the cross-modal global context between them. The composition of the network is as follows:

(1) Task encoder: in the task encoder, the original state features of all tasks, T_{it}, are first mapped to d-dimensional embedding vectors, h_T, through a linear projection layer. Subsequently, a multi-head self-attention layer encodes this embedding sequence to obtain $h_T' = \text{MHA}(h_T, h_T, h_T)$, thereby capturing the internal dependencies among tasks such as spatial distribution and priority correlation. In addition, the encoded task embedding sequence is masked to ignore invalid tasks, and the average value is calculated on this basis to obtain the global task overview, $\overline{h}_T$, which serves as a global snapshot of the current state of all tasks.

(2) UAV encoder: the UAV encoder processes the original state features of UAVs (e.g., position coordinates, load capacity, and remaining endurance), U_{it}, in the same way as the task encoder: first mapping them to d-dimensional embeddings, h_U, through a linear layer, and then obtaining the UAV swarm context, $h_U' = \text{MHA}(h_U, h_U, h_U)$, via multi-head self-attention encoding, enabling UAVs to implicitly exchange state information and identify potential collaborative partners. Similarly, the average value of the UAV embedding sequence is calculated to obtain the global UAV overview, $\overline{h}_U$.

(3) Cross-encoder: to learn the cross-modal correlation between tasks and UAVs, the encoded task context, h_T', and UAV context, h_U', are input into two parallel cross-attention layers: with the UAV context as the query and the task context as the key-

value pairs, the UAV-task context, $h'_{UT} = \mathrm{MHA}\left(h'_U, h'_T, h'_T\right)$, is obtained; with the task context as the query and the UAV context as the key-value pairs, the task-UAV context, $h'_{TU} = \mathrm{MHA}\left(h'_T, h'_U, h'_U\right)$, is obtained. This process achieves the bidirectional fusion of task and UAV features, providing unified cross-modal correlation information for the policy output of the Actor and the value estimation of the Critic.

4.3.3. Temporal Modeling Unit

Since UAV task allocation is a sequential decision-making process, historical decisions have a significant impact on current selections. Therefore, a Temporal Modeling Unit LSTM is introduced between the encoder and the dual-branch output as a shared temporal feature extraction module for the Actor and the Critic. This unit takes the current UAV-task fused feature output by the encoder as input and simultaneously receives the LSTM hidden state (h_{t-1}, c_{t-1}) at the previous moment. It updates and outputs the temporal-aware feature, h_t, and the new hidden state, (h_t, c_t), at the current moment through the Long Short-Term Memory mechanism, as shown in Equation (8):

$$h_t, (h_t, c_t) = \mathrm{LSTM}(h'_{UT}(i), (h_{t-1}, c_{t-1})) \tag{8}$$

In Equation (8), $h'_{UT}(i)$ is the cross-modal feature of the i-th UAV. The dimension of the LSTM hidden state is consistent with the feature embedding dimension, d, and a batch-first structural design is adopted to adapt to the processing requirements of batch task allocation instances.The network structure of the LSTM is shown in Figure 2.

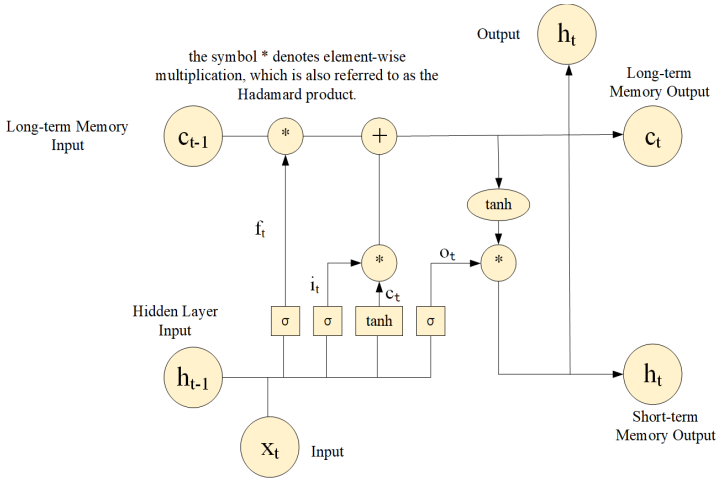

Figure 2. The network structure of the LSTM.

4.3.4. Decoder

The decoder module adopts an Actor–Critic dual-branch structure, which shares the preceding backbone network and temporal modeling features to realize task allocation strategy output and state value evaluation respectively. It consists of the following parts:

(1) Policy Network

The policy network is the network structure of the Actor branch, whose core is to generate the probability distribution of UAVs' task selections. Its structure is shown in Figure 3. First, extract the individual feature, $h'_{UT}(i)$, of the current decision-making UAV i from the UAV-task context h'_{UT}, concatenate it with the global task overview, $\overline{h}_T$, and the global UAV overview, $\overline{h}_U$, and map it through a Multi-Layer Perceptron (MLP) to obtain the current state feature with the dimension maintained as d: $h_{ai} = \mathrm{MLP}(\mathrm{Concat}(h'_{UT}(i), \overline{h}_U, \overline{h}_T))$. Then combine this state feature with the h_t

output by the Temporal Modeling Unit, feed it into a multi-head attention layer, and introduce a binary mask, M_i, and interact with the task-UAV context, h'_{TU}, to obtain the enhanced representation: $h'_{ai} = MHA(h_t, h'_{TU}, h'_{TU}|M_i)$, where the mask, M_i, is used to constrain UAVs to select only valid tasks. Finally, take h'_{ai} as the query and h'_{TU} as the key-value pairs, calculate the attention scores through a Pointer Network, and normalize them via the Softmax function to obtain the probability distribution of the current UAV's selections for all tasks, as shown in Equation (9):

$$\pi_\theta(a|s_t) = \text{Softmax}\left(\text{Attention}(h'_{ai}, h'_{TU}, h'_{TU})\right) \tag{9}$$

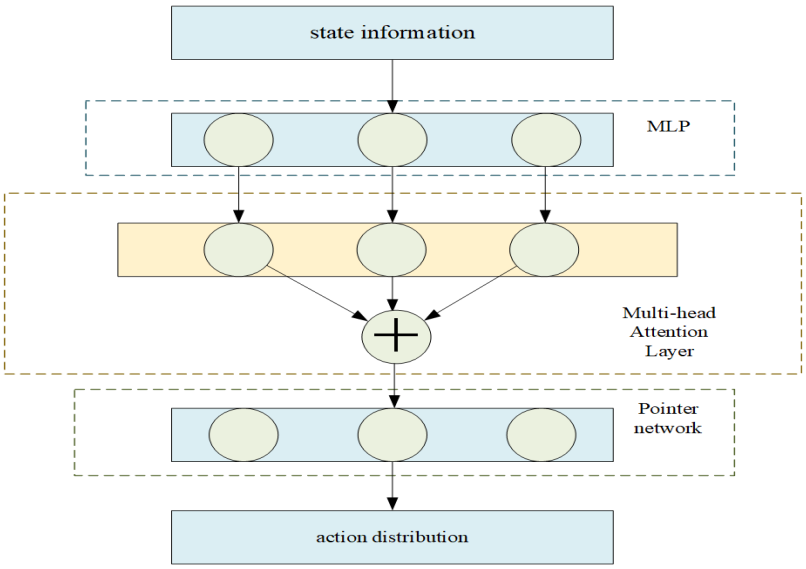

Figure 3. The network structure of the Actor.

This probability distribution supports stochastic sampling in the training phase and greedy selection in the inference phase, realizing the action decision-making for task allocation.

(2) Value Network

The value network is the network structure of the Critic branch, whose core is to evaluate the value, $V_\phi(s_t)$, of the current system state, s_t, providing a basis for the calculation of the advantage function in reinforcement learning. Its structure is shown in Figure 4. The Critic branch utilizes the enhanced feature, h'_{ai}, of the Actor branch and fuses it with the global temporal feature, h_t, output by the Temporal Modeling Unit to obtain the state feature, $h_{critic} = \text{Concat}\left(h'_{ai}, h_t\right)$, containing local decision-making information and global temporal information. Then feed the aggregated feature into a value head composed of two layers of fully connected networks, where the middle layer adopts the ReLU activation function and the output layer is a linear layer, to obtain the scalar value estimation of the current state, as shown in Equation (10):

$$V_\phi(s_t) = \text{MLP}_{critic}(h_{critic}) \tag{10}$$

In Equation (10), since h'_{ai} and h_t are each of dimension d and concatenated together, the input dimension of MLP_{critic} is 2d and the output dimension is 1, which directly reflects the quality of the current state.

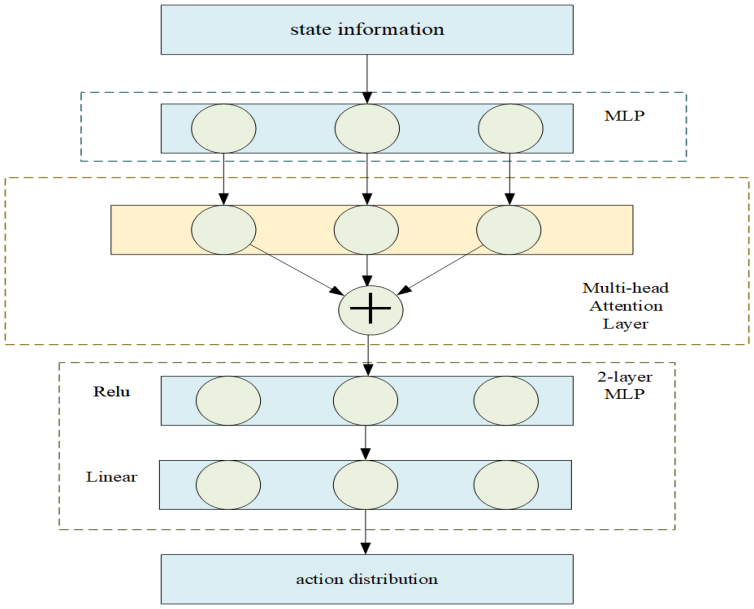

Figure 4. The network structure of the Critic.

4.4. Training Algorithm

Aiming at the sequential decision-making characteristics and multi-agent collaboration requirements of heterogeneous multi-UAV task assignment, this paper adopts the PPO algorithm for policy training. The PPO algorithm [40,41] is a classic policy-gradient reinforcement learning algorithm. It addresses the problems of unstable updates and easy divergence in traditional policy-gradient algorithms by restricting the step size of policy updates. At its core, PPO implements policy learning and state-value evaluation through an Actor–Critic dual-network architecture, making it one of the mainstream algorithms for solving sequential decision-making problems in both continuous and discrete action spaces.

In this algorithm, each UAV is modeled as an independent agent, and the training process is theoretically framed by the Partially Observable Markov Decision Process (POMDP). Policy learning and value evaluation are realized via the Actor–Critic network structure. Specifically, the Actor network acts as the policy network, which takes the temporal observation sequence of UAVs as input and outputs the action probability distribution of task selection to guide the agent to make optimal decisions. The Critic network serves as the value network, which evaluates the quality of the current system state by fitting the state-value function and computes the advantage function for policy updates of the Actor network. The two networks work cooperatively to achieve stable policy optimization.

In multi-agent scenarios, all UAVs share the parameters of the Actor–Critic network, and collaborative decision-making is realized through global state information interaction. This implementation can effectively reduce training complexity and improve the generalization ability of the policy.

To balance the bias and variance of advantage estimation, the Generalized Advantage Estimation (GAE) method is adopted to calculate the action advantage value, as shown in Equations (11) and (12):

$$A_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+1} \tag{11}$$

$$\delta_{t+1} = r_{t+1} + \gamma V_\phi(s_{t+1+1}) - V_\phi(s_{t+1}) \tag{12}$$

In Equation (12), δ_{t+1} denotes the temporal difference error, $\gamma \in [0, 1]$ is the discount factor for reward, and $\lambda \in [0, 1]$ is the GAE smoothing parameter. This method yields an unbiased advantage estimation, providing a reliable gradient signal for policy updates.

The calculation of the cumulative reward, R_t, is shown in Equation (13):

$$R_t = A_t + V_\phi(s_t) \tag{13}$$

In Equation (13), R_t reflects the cumulative discounted reward starting from time, t, which is used for the supervised training of the value network.

To avoid training instability caused by excessively large policy update magnitudes, PPO adopts a clipped objective function to restrict the trust region of policy updates, with the specific form as follows:

$$L_{clip}(\theta) = \mathbb{E}_{(s_t,a_t,A_t)\sim\mathcal{D}}[\min(r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)] \tag{14}$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t,h_t)}{\pi_{\theta_{old}}(a_t|s_t,h_t)} \tag{15}$$

In Equations (14) and (15), $r_t(\theta)$ is the action probability ratio of the new and old policies, ϵ is the clip coefficient (set to 0.2 in this paper), and $\mathcal{D}$ is the experience replay buffer that stores trajectory data $(s_t, a_t, r_t, s_{t+1}, h_t)$ generated by the interaction between agents and the environment.

The update of policy parameters, θ, is achieved by minimizing the above objective function, with the gradient calculated as follows:

$$\nabla_\theta L_{clip}(\theta) = \mathbb{E}[\nabla_\theta r_t(\theta) \cdot A_t \cdot 1\{r_t(\theta)A_t \le clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t\}] \tag{16}$$

The value network optimizes its parameters, ϕ, by minimizing the mean squared error loss, with the loss function shown in Equation (17):

$$L_V(\phi) = \mathbb{E}_{(s_t,R_t)\sim\mathcal{D}}\left[\left(V_\phi(s_t) - R_t\right)^2\right] \tag{17}$$

This loss ensures that the value network can accurately estimate the long-term cumulative reward of the state, providing a reliable benchmark for advantage calculation. To encourage agents to explore diverse policies and avoid falling into local optima, an entropy regularization term is introduced into the total loss, as shown in Equation (18):

$$L_{Ent} = -\mathbb{E}_{s_t\sim\mathcal{D}}[H(\pi_\theta(\cdot|s_t,h_t))] \tag{18}$$

In Equation (18), H is the entropy function, which measures the uncertainty of the action probability distribution. The total loss function is shown in Equation (19):

$$L_{total} = -L_{clip}(\theta) + c_V L_V(\phi) - c_H L_{Ent} \tag{19}$$

In Equation (19), c_V is the value loss weight and c_H is the entropy regularization weight, which balance policy optimization, value estimation, and exploration efficiency through hyperparameters.

To improve data utilization efficiency and training stability, a shared experience replay buffer, D, is designed to store the interaction trajectories of all agents during multiple training rounds. When the amount of data in the buffer reaches a preset threshold, multi-round updates are initiated, and batch data is randomly sampled for gradient descent optimization in each update.

To accelerate the training process, a distributed training framework is adopted: multiple parallel worker nodes collect experience synchronously, the master node aggregates the data and executes parameter updates, and the updated network parameters are synchronized to all worker nodes, realizing the parallel execution of experience collection and

policy optimization. Meanwhile, to handle the temporal dependency characteristics of task allocation, an LSTM unit is embedded in the Actor network, and each agent maintains an independent hidden state, h_t, which is dynamically updated after each step of decision-making to ensure the effective transmission of temporal information.

The specific procedures for multi-agent PPO training are presented in Algorithm 1:

Algorithm 1. Multi-Agent PPO Training Algorithm

S1: Initialization: set the parameters of the Actor network, θ, parameters of the Critic network, ϕ, experience replay buffer, D, learning rate, η, discount factor, γ, clip coefficient, ϵ, number of update rounds, K, and batch size, B.
S2: For each training episode:
a. Reset the task environment and agent states and initialize the LSTM hidden state, h_0.
b. For each decision step $t = 0, 1, \ldots, T-1$:
i. Each agent observes the current system state, s_t.
ii. The Actor network samples an action, a_t, according to $\pi_\theta(a_t|s_t, h_t)$ and executes the task allocation operation.
iii. Observe the immediate reward, r_t, fed back by the environment and transition to the next state, s_{t+1}.
iv. Update the LSTM hidden state, h_{t+1}, and store the trajectory data, $(s_t, a_t, r_t, s_{t+1}, h_t, \hat{A}_t, R_t)$, into D.
c. When the amount of data in $D \geq B$, perform K rounds of updates:
i. Randomly sample a batch of data from D.
ii. Calculate the policy loss, $L_{clip}(\theta)$, value loss, $L_V(\phi)$, and entropy regularization term, L_{Ent}.
iii. Calculate the total loss, L_{total}, update the parameters θ and ϕ via gradient descent, and perform gradient clipping.
S3: Repeat Step 2 until the number of training episodes reaches the preset maximum value or the policy converges.
S4: Output: The optimal task allocation policy, π_θ^*.

5. Experimental Results and Analysis

5.1. Experimental Setup

All experiments in this paper are implemented based on the PyTorch 1.18.0 deep learning framework. The training and inference processes are deployed on a single NVIDIA RTX 4090 GPU, with an Intel Core i9-13900K CPU (32 cores, 64 threads), 128 GB of memory, and the Ubuntu 22.04 operating system. The experimental code is written in Python 3.9, relying on scientific computing libraries, such as NumPy 1.24.3 and SciPy 1.10.1, and Matplotlib 3.7.1 for result visualization.

A heterogeneous UAV multi-agent simulation environment based on continuous space is constructed. Since the research focuses on the scheduling logic and collaboration mechanisms of heterogeneous UAV task assignment algorithms, a two-dimensional environment can strip out the path-planning complexity introduced by spatial dimensions. This enables accurate verification of the algorithm's performance in core modules such as task matching, coalition formation, and deadlock prevention, and avoids ambiguous experimental results caused by multi-factor coupling. Therefore, this paper adopts a two-dimensional simulation environment, in which the UAV speed is set to be uniform and the dynamic model is simplified. The task area is normalized to a 1×1 2D plane $[0, 1] \times [0, 1] \subset \mathbb{R}^2$, and the positions of base stations and tasks are generated via uniform random distribution. UAVs have multiple skill dimensions, with skill vectors encoded in binary (1 indicates possession

of the skill, and 0 indicates non-possession). The total number of UAVs is dynamically adjusted according to scenarios, and UAVs of the same type have identical skill vectors. The comprehensive skill vector of a collaborative group is obtained through the element-wise summation of the skill vectors of UAVs in the group. Task requirements are additive: when generating tasks, it is ensured that the requirement vector is non-zero and that at least one UAV collaborative group can meet its skill requirements. The experiment designs four test sets with increasing difficulty, each containing 50 randomly generated unseen instances.

To ensure that the simulation results possess engineering reference value, the simulation scenarios in this study are designed to closely align with practical applications. The UAVs are defined with skill vectors according to realistic functions such as reconnaissance, relay, and delivery, and the task requirements match multi-skill collaborative scenarios. The spatial distribution and flight time consumption conform to physical laws, and dynamic disturbances (faults, parameter variations, newly added tasks) replicate the non-ideal conditions encountered in actual deployment, thereby ensuring that the simulation results are of engineering reference value.

The training hyperparameters are set as follows: for the PPO algorithm, the learning rate, $\eta = 3 \times e^{-4}$, discount factor, $\gamma = 0.99$, GAE smoothing parameter, $\lambda = 0.95$, clip coefficient, $\epsilon = 0.2$, experience replay buffer threshold = 2048, four rounds of parameter updates are performed after each buffer is full, and batch size, $B = 64$. The value loss weight, $c_V = 0.5$, entropy regularization weight, $c_H = 0.01$, and maximum gradient clipping norm = 100. In the CCT mechanism, the empirical coefficients are $\alpha = 1.5$ and $\beta = 1.2$; the initial timeout penalty is $r_{delay} = -0.5$, which exponentially decays to -0.1 with the number of training rounds.

To comprehensively evaluate the performance of the UAV task allocation model, the experiment designs evaluation metrics from multiple dimensions, including task completion effectiveness, time efficiency, resource consumption, and scheduling rationality, as follows:

Task Success Rate: Defined as the ratio of the number of successfully completed tasks to the total number of tasks. This metric directly reflects the model's task coverage capability and execution reliability, serving as a core indicator for evaluating the effectiveness of task allocation.

Makespan: Refers to the total duration from the start of the first task to the completion of the last task, which is a key indicator for measuring the overall efficiency in scheduling problems. A shorter makespan indicates that the model can quickly coordinate UAVs to complete all tasks, reflecting the time optimization capability of task allocation.

Time Cost: Comprehensively counts the total time consumption of all UAVs during task execution, including flight time, task operation time, and intermediate waiting time. This metric reflects the overall consumption of time resources in the system and is directly related to the economic efficiency of task execution.

Waiting Time: Refers to the cumulative waiting time of UAVs caused by resource conflicts, task dependencies, or scheduling delays during task allocation. Shorter waiting time indicates smoother dynamic scheduling of UAV resources and lower resource idle rate.

Travel Distance: The cumulative flight mileage of all UAVs during task execution. This metric is directly related to UAV energy consumption; shorter travel distance means lower energy consumption and higher resource utilization efficiency.

Waiting Efficiency: Defined as the ratio of waiting time to total task execution time. This metric quantifies the effective utilization of time resources during scheduling: a higher value indicates a lower proportion of waiting time and better scheduling coordination of the system.

The above metrics construct a comprehensive evaluation system from the perspectives of task completion quality, time efficiency, resource consumption, and scheduling coordination, which can objectively reflect the actual performance of the UAV task allocation model in complex and dynamic environments.

5.2. Parameter Sensitivity Analysis

To verify the influence of core hyperparameters on model performance, this section selects four types of hyperparameters: the PPO clip coefficient, ϵ, the LSTM time window length, L, the initial CCT penalty, and the penalty coefficient, η, in the reward function. Under the experimental scenario ($n = 9$, $m = 3$, $p = 20$, $d = 3$), the hyperparameter values are varied via the control variable method, and the variations in three core metrics—task success rate, makespan, and waiting time—are evaluated. The value ranges of hyperparameters and experimental results are shown in Tables 1–4.

5.2.1. PPO Clip Coefficient, ϵ

The clip coefficient, ϵ, is used to limit the policy update step size of the PPO algorithm, with values set to $[0.1, 0.2, 0.5]$ and other hyperparameters fixed. The experimental results are shown in Table 1:

Table 1. Sensitivity analysis results of PPO clip coefficient, ϵ.

Value		Task Success Rate	Makespan	Waiting Time
0.1	mean	1	27.53	1.61
	std	0	5.67	1.16
0.2	mean	1	24.44	1.28
	std	0	5.30	1.07
0.5	mean	1	28.24	2.84
	std	0	5.09	1.47

It can be seen from Table 1 that $\epsilon = 0.2$ is the optimal value, which achieves the best balance between the policy update speed and training stability. This is consistent with the recommended value in the original PPO paper, verifying the rationality of this hyperparameter selection.

5.2.2. LSTM Time Window Length, L

The time window length, L, controls the length of temporal information captured by LSTM, with values set to $[3, 6, 9]$ and other hyperparameters fixed. The experimental results are shown in Table 2:

Table 2. Sensitivity analysis results of LSTM time window length, L.

Value		Task Success Rate	Makespan	Waiting Time
3	mean	1	27.73	2.09
	std	0	5.41	1.11
6	mean	1	24.44	1.28
	std	0	5.30	1.07
9	mean	1	28.98	1.64
	std	0	5.21	1.10

It can be seen from Table 2 that L = 6 is the optimal value, which can fully capture the temporal dependencies in UAV task assignment while avoiding redundant information and increased computational overhead caused by an excessively long window.

5.2.3. Initial CCT Penalty, r_{delay}

The initial CCT penalty is a core parameter of the CCT mechanism, with values set to $[-0.3, -0.5, -0.8]$ and other hyperparameters fixed. The experimental results are shown in Table 3:

Table 3. Sensitivity analysis results of initial CCT penalty, r_{delay}.

Value		Task Success Rate	Makespan	Waiting Time
−0.3	mean	1	27.05	2.33
	std	0	5.65	1.30
−0.5	mean	1	24.44	1.28
	std	0	5.30	1.07
−0.8	mean	1	27.84	1.39
	std	0	5.78	1.96

It can be seen from Table 3 that $r_{delay} = -0.5$ is the optimal value. This setting ensures that the countdown duration covers the necessary time for UAV flight and coalition formation, while avoiding resource idling caused by an overly long countdown.

5.2.4. Penalty Coefficient, η, in the Reward Function

The penalty coefficient, η, in the reward function is used to balance the weights between the makespan, T, and the average capacity waste rate, W, with values set to $[5, 10, 15]$ and other hyperparameters fixed. The experimental results are shown in Table 4:

Table 4. Sensitivity analysis results of Penalty Coefficient, η, in the Reward Function.

Value		Task Success Rate	Makespan	Waiting Time
5	mean	0.97	67.88	37.33
	std	0.11	39.99	13.94
10	mean	1.00	24.44	1.28
	std	0.00	5.30	1.07
15	mean	0.82	110.97	24.66
	std	0.35	26.97	12.40

It can be seen from Table 4 that η = 10 is the optimal value, which assigns comparable weights to makespan and capacity waste rate in the reward function, achieving dual optimization of time efficiency and resource utilization efficiency.

The sensitivity analysis results of the above four core hyperparameters show that the hyperparameter values selected in this paper are all within the optimal performance interval. Moreover, when slightly adjusted near these values, the core metrics of the model do not deteriorate significantly, proving that the model has certain robustness to hyperparameters. The optimal values of hyperparameters follow the principle of domain prior + experimental tuning, which not only refers to classic studies in reinforcement learning and UAV task assignment but also verifies their adaptability in the scenario of this study via the control variable method, with sufficient selection basis.

5.3. Performance Comparison Experiments

To verify the comprehensive performance advantages of the proposed model across different scenarios, we compare it with various types of mainstream baseline methods to observe improvements in makespan, success rate, and computational efficiency. Representative methods including the Genetic Algorithm (GA), Hungarian Algorithm (HA), Max–Min Fairness Algorithm (MMFA), Particle Swarm Optimization (PSO), and Consensus-Based Bundle Algorithm (CBBA) are selected as baselines, covering centralized, distributed, reinforcement learning, and heuristic strategies to ensure comprehensive comparison. The experiments are conducted across different environmental scenarios, and the performance comparison results are shown in Figures 5–8.

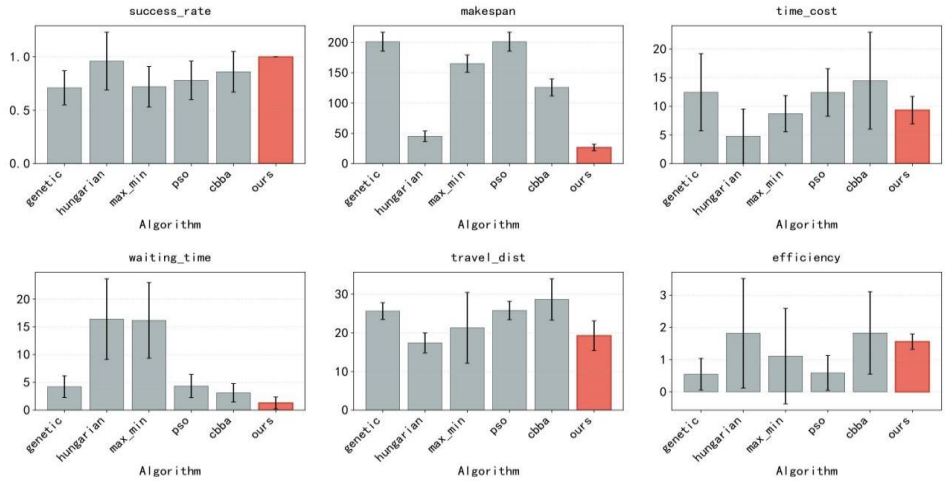

Figure 5. Results of different metrics when n = 9, m = 3, p = 20, d = 3.

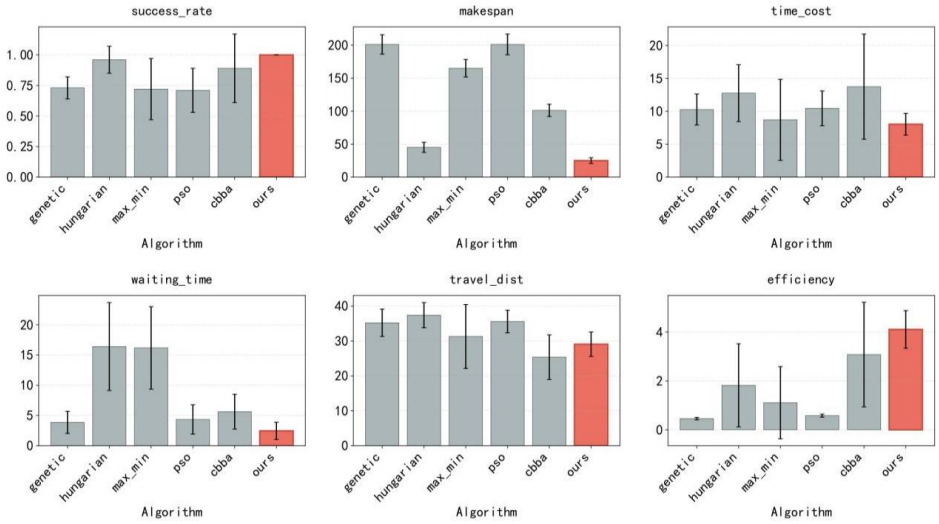

Figure 6. Results of different metrics when n = 15, m = 5, p = 20, d = 5.

Comprehensive performance verification of the proposed algorithm and five mainstream baseline algorithms (including GA and HA) is conducted across four different scenarios. The results demonstrate that the proposed algorithm exhibits significant advantages in comprehensive performance, with its core metrics outperforming all baseline methods.

Task Success Rate: The proposed algorithm achieves a 100% success rate (mean = 1.00, standard deviation = 0.00) across all scenarios. Among the baseline algorithms, the best-performing HA only reaches a 96% success rate, while GA and PSO typically achieve

success rates between 71% and 78%, and CBBA maintains a success rate between 85% and 90%. This highlights the reliability and stability of the proposed algorithm in task execution across different scenarios.

Makespan: The proposed algorithm performs particularly outstandingly, with a mean value ranging from 25.12 to 30.40, achieving an order-of-magnitude improvement compared to baseline algorithms. Specifically, the makespan of GA and PSO both exceed 200, HA has a makespan of approximately 45.21, and CBBA ranges from 94 to 125. By optimizing the scheduling strategy, the proposed algorithm significantly shortens the task cycle and improves system throughput.

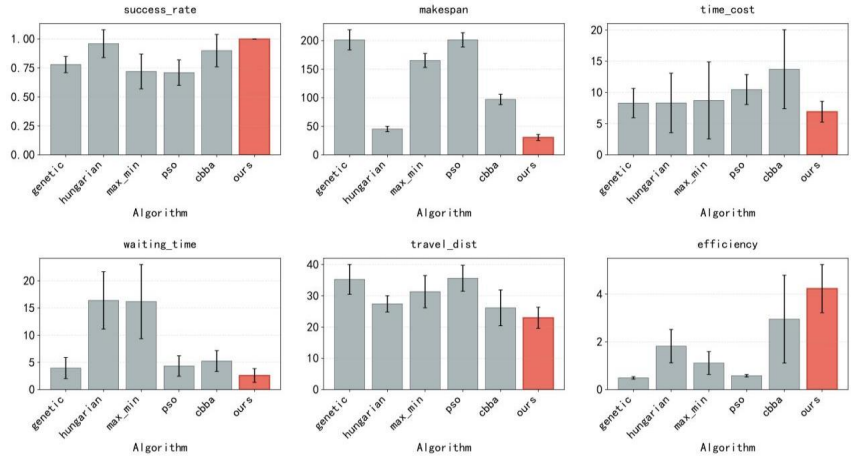

Figure 7. Results of different metrics when n = 25, m = 5, p = 20, d = 5.

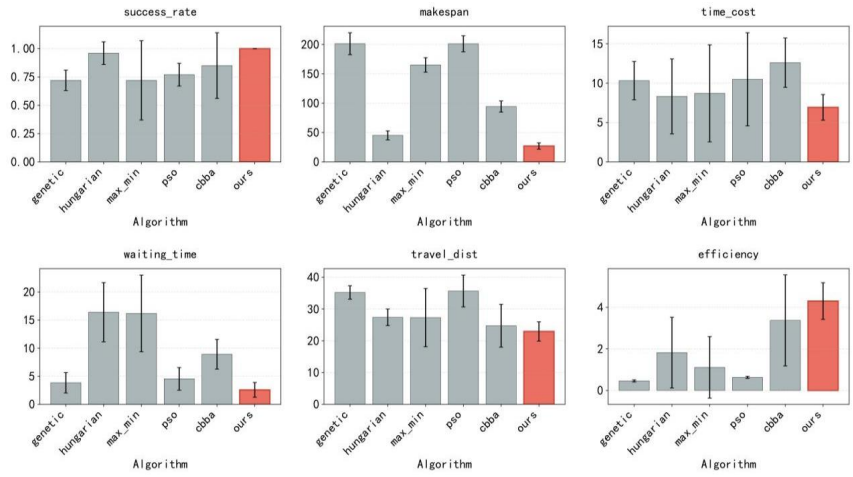

Figure 8. Results of different metrics when n = 50, m = 5, p = 50, d = 5.

Computational Efficiency and Resource Utilization: The mean time cost of the proposed algorithm is only 6.90–9.32, which is lower than all baseline algorithms except in the first scenario, and far superior to CBBA (12.6–14.46). The waiting time is as low as 1.29–2.59, which is significantly lower than the ~16 of HA and MMFA and the 3–9 range of other baselines, reflecting the high efficiency of the proposed algorithm in resource scheduling.

Comprehensive Efficiency: The proposed algorithm reaches 4.11, 4.23, and 4.30 in Scenarios 2–4, respectively, far exceeding all baseline algorithms. Even though it is slightly lower than HA and CBBA in the first scenario, its absolute advantages in core metrics such as success rate and makespan compensate for this. Additionally, the travel distance of the proposed algorithm remains at a low level of 19.26–29.08, achieving a balance between resource consumption and execution efficiency.

In summary, the proposed algorithm can balance high success rate, short task cycle, low computational overhead, and high resource utilization efficiency across different scenarios, demonstrating strong environmental adaptability. Its performance advantages verify the unique value of the strategy combining reinforcement learning and temporal modeling in scheduling optimization problems, providing an efficient solution for task scheduling in complex scenarios.

5.4. Ablation Study

To verify the necessity of the core innovative components of the proposed method (LSTM temporal module, CCT anti-deadlock mechanism, and Attention collaboration-aware module), we construct variant models by removing individual components and quantify the contribution of each component to the overall performance. Three variant models are built based on the full model, as follows:

w/o LSTM: Removes the LSTM Temporal Modeling Unit, retaining only the attention mechanism and CCT mechanism, to verify the role of temporal dependency modeling.

w/o CCT: Removes the CCT anti-deadlock mechanism, retaining the LSTM and attention mechanism, to verify the necessity of the anti-deadlock mechanism.

w/o Attention: Removes the attention mechanism, retaining the LSTM and CCT mechanism, and uses simple linear projection to fuse features to verify the role of collaboration awareness.

The experiments are conducted across three scenarios, and the performance comparison results of different models are visualized in Figures 9–11.

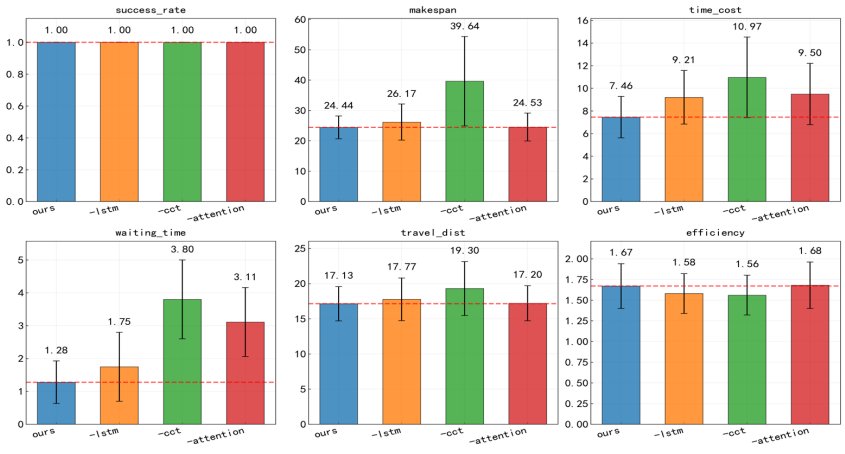

Figure 9. Results of different metrics when n = 9, m = 3, p = 20, d = 3.

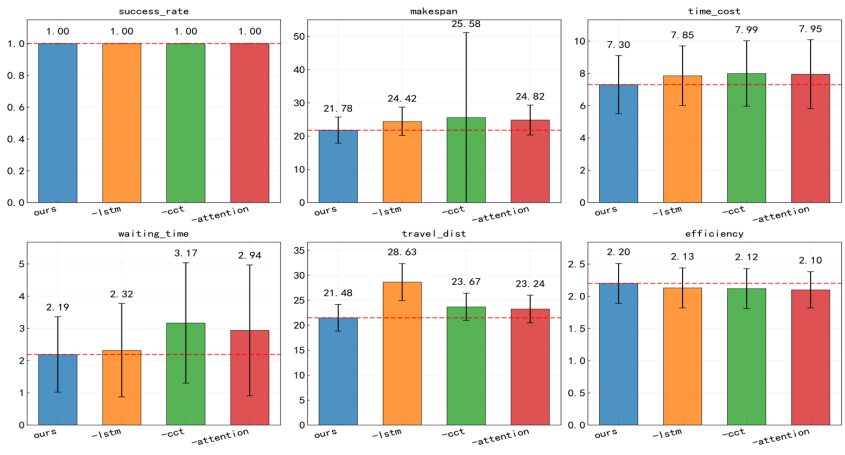

Figure 10. Results of different metrics when n = 15, m = 5, p = 20, d = 5.

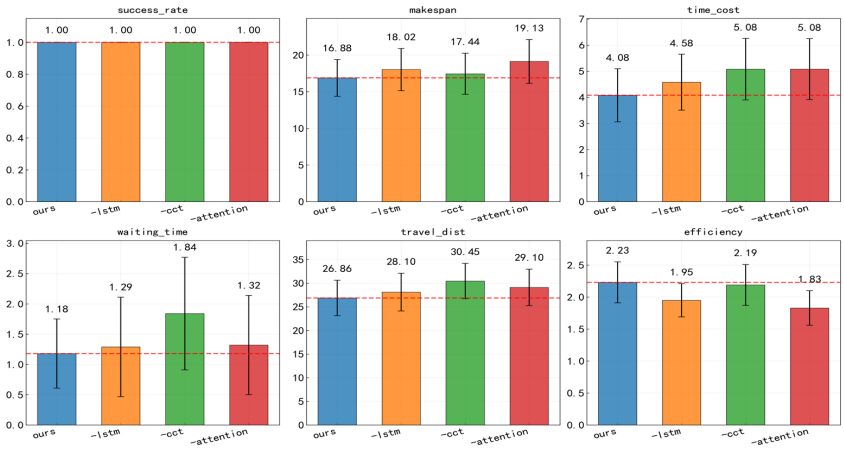

Figure 11. Results of different metrics when n = 25, m = 5, p = 20, d = 5.

The results in Figures 9–11 demonstrate that the full model significantly outperforms all variant models across all performance metrics, fully confirming the critical role of each core component and the rationality of the overall architecture design. All models maintain a 100% task success rate, with performance differences mainly reflected in efficiency metrics.

In the core makespan metric, the proposed model achieves the best performance, with a mean value of only 16.88–24.44. All variant models exhibit varying degrees of performance degradation after removing core components.

The w/o CCT variant shows the most significant degradation in Scenario 1, with the makespan surging from 24.44 to 39.64 (an increase of 62.2%) and time cost rising from 7.46 to 10.97. This verifies the irreplaceable role of the CCT mechanism in avoiding scheduling deadlocks and shortening task cycles. In Scenarios 2–3, the makespan and waiting time of the w/o CCT variant are also higher than those of the proposed model, further confirming its value in ensuring stability in resource scheduling.

The role of the LSTM temporal module is highlighted by the w/o LSTM variant: its makespan is higher than that of the proposed model in all scenarios (Scenario 1: 26.17 vs. 24.44; Scenario 3: 18.02 vs. 16.88), and its comprehensive efficiency drops from 2.23 to 1.95 in Scenario 3. This indicates that temporal modeling can effectively capture the time dependencies between tasks and optimize scheduling timing.

The necessity of the Attention collaboration-aware module is also fully verified: the w/o Attention variant exhibits significant increases in makespan (Scenario 3: 19.13 vs. 16.88) and waiting time (Scenario 1: 3.11 vs. 1.28), with comprehensive efficiency dropping to 1.83 in Scenario 3 (lower than the proposed model's 2.23). This shows that the attention mechanism can enhance collaboration awareness between components, reducing resource conflicts and waiting losses.

In summary, the proposed model achieves the shortest makespan, lowest computational overhead, and highest comprehensive efficiency through the synergistic effect of LSTM temporal modeling, CCT anti-deadlock mechanism, and Attention collaboration awareness. The core components complement each other and are indispensable, collectively supporting the high performance of the algorithm, thus verifying the scientific validity and effectiveness of the overall architecture design.

5.5. Robustness Experiments

To verify the robustness of the proposed heterogeneous UAV task allocation algorithm in complex non-ideal scenarios, the robustness experiments are conducted in this section, focusing on investigating the impacts of three types of perturbations—UAV failure probability, core parameter variation, and dynamic task generation rate—on the algorithm's

performance. A full factorial test scheme is adopted in the experiments, where the three perturbation factors are all set to gradient levels of [0.0, 0.1, 0.3, 0.5]. The experiment was conducted under the scenario (n = 15, m = 5, p = 20, and d = 5). Each combination is independently run for 20 test rounds, and a 60 s cross-platform timeout threshold is set for a single test round. The tests are implemented based on the PyTorch framework, with the pre-trained model loaded and parameters fixed and the task environment reinitialized for each test round.

The experiments are carried out by adding different combinations of perturbation factors, and gradient levels of UAV failure probability, UAV core parameter variation, and dynamic task generation rate are introduced into the test environment respectively. Among them, the UAV failure probability is directly configured in the task environment; the core parameters are randomly perturbed according to the set variation degree with the lower limit of physical rationality retained; the dynamic task generation rate is used to control the proportion of newly added tasks per unit time. During the test, six core experimental metrics are mainly observed, including task success rate, maximum completion time, time cost, task waiting time, flight distance, and execution efficiency. The average value of each metric is calculated after 20 test rounds for each combination, and the final results are shown in Figures 12–14.

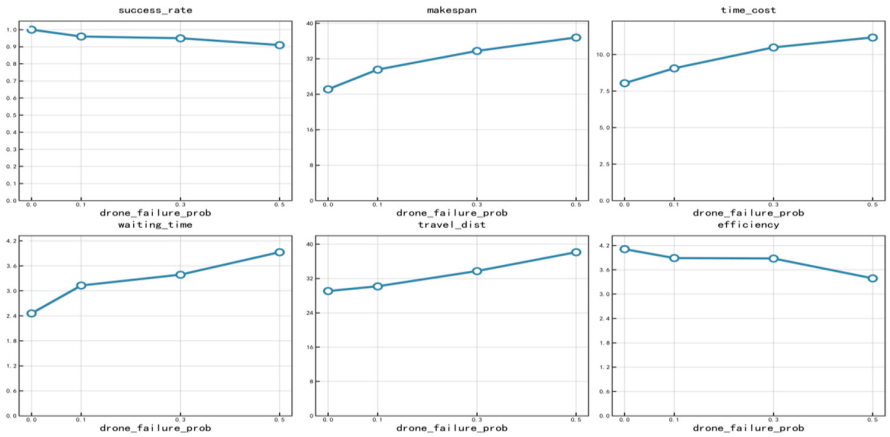

Figure 12. Comparison chart of results under different UAV failure probabilities.

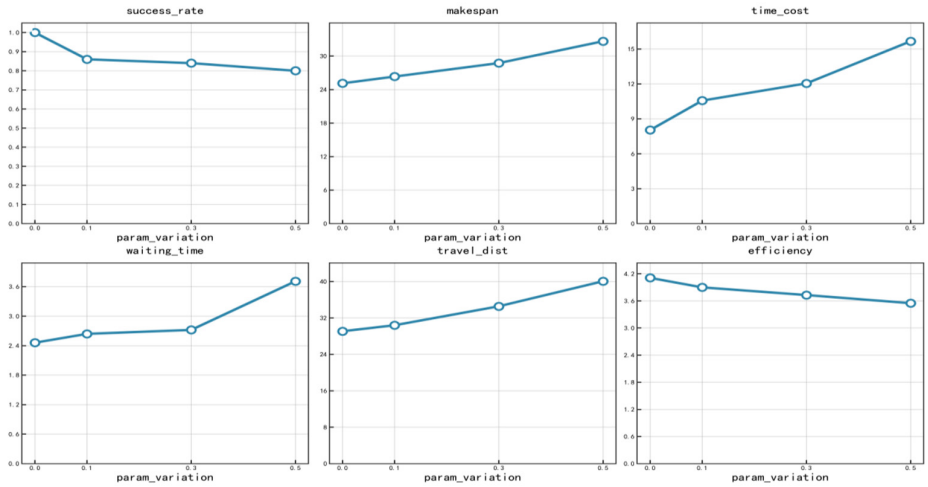

Figure 13. Comparison chart of results under different parameter variation rates.

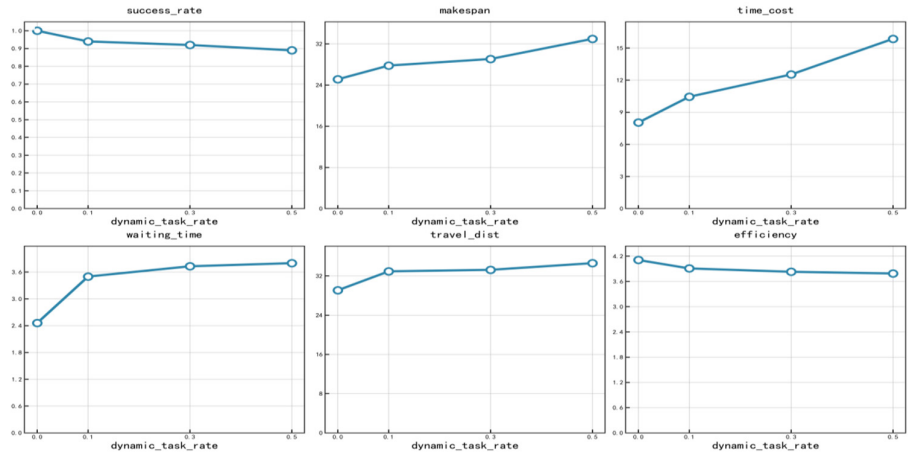

Figure 14. Comparison chart of results under different dynamic task rates.

In terms of the results, the algorithm exhibits robust fault tolerance in the UAV failure probability perturbation test. When the failure probability increases from 0.0 to 0.5, the task success rate only decreases gently from 1.00 to 0.91, a decrease of less than 10%; the core makespan increases from 25.12 to 36.81, and the comprehensive efficiency decreases from 4.11 to 3.39. The variation range of the metrics is moderate without any abrupt deterioration, which proves that the algorithm can effectively address the resource shortage problem caused by UAV failures.

Faced with core parameter variation perturbations, the algorithm still maintains high performance. Even when the parameter variation rate reaches 0.5, the task success rate remains at a high level of 0.80, the makespan is controlled at 32.65, and the comprehensive efficiency is 3.55, showing only a slight decline compared with the ideal scenario. This result verifies the algorithm's adaptability to fluctuations in core UAV parameters such as payload and speed, and it can operate stably without relying on precise parameter configuration.

Under the perturbation of dynamic task generation rate, the algorithm demonstrates excellent dynamic response capability. When the dynamic task rate increases to 0.5, the task success rate still reaches 0.89, the makespan is 32.98, and the comprehensive efficiency is 3.79, with all metrics remaining within a reasonable range. Compared with the ideal scenario, the performance decline under high dynamic task pressure is less than 8%, which highlights the scheduling flexibility of the algorithm in scenarios with dynamically changing task volumes.

In summary, the proposed model exhibits a gentle variation trend in all key metrics in the full gradient tests of the three core perturbations, without any significant performance collapse. Even in extreme scenarios with a 50% failure probability, 50% parameter variation, or 50% dynamic task rate, it can still maintain a task success rate of over 80% and a comprehensive efficiency of over 3.39, which fully proves its excellent robustness. This characteristic enables it to adapt to the interference of various non-ideal factors in practical applications, providing a reliable guarantee for the engineering implementation of heterogeneous UAV task allocation.

5.6. Impact of Inference Strategies on Decision Quality

The inference strategy experiments aim to verify the impact of different inference strategies on the decision performance of multi-agent task allocation systems. The greedy strategy and stochastic sampling strategy are selected as the core comparison objects, and evaluations are conducted from three dimensions: makespan, inference efficiency, and task success rate. The experiments load the optimal weights saved during the training process to ensure parameter convergence; the task environment includes configurations such as multi-

species agents and task scale, with the maximum number of actions per round limited to 300. The evaluation metrics specifically include: makespan, which reflects the overall system efficiency; inference efficiency (average inference time per action, total inference time), which measures the computational overhead; and task success rate (the ratio of completed tasks to the total number of tasks), which evaluates decision effectiveness. To eliminate random seed errors, five independent test rounds are run for each strategy configuration, with the average value taken as the final result. The experiment was conducted under the scenario (n = 15, m = 5, p = 20, and d = 5).

Among them, the greedy strategy is a basic deterministic strategy for multi-agent decision-making, which directly selects the action with the highest output probability of the model at each decision step with no additional sampling overhead. The stochastic sampling strategy samples N candidate actions ($N \in \{1,5,10,20\}$) based on the model's action probability distribution and selects the action with the highest value for execution, balancing exploration and exploitation through a small number of samples. In the strategy execution phase, the greedy strategy directly selects the optimal action, while the stochastic sampling strategy samples N candidate actions based on the Categorical distribution. The maximum number of sampling cycles is set to $N \times 10$ to avoid infinite loops; if the sampling is insufficient, greedy actions are used as a fallback, and the action with the highest value is ultimately selected. The experimental results are shown in Figures 15–17.

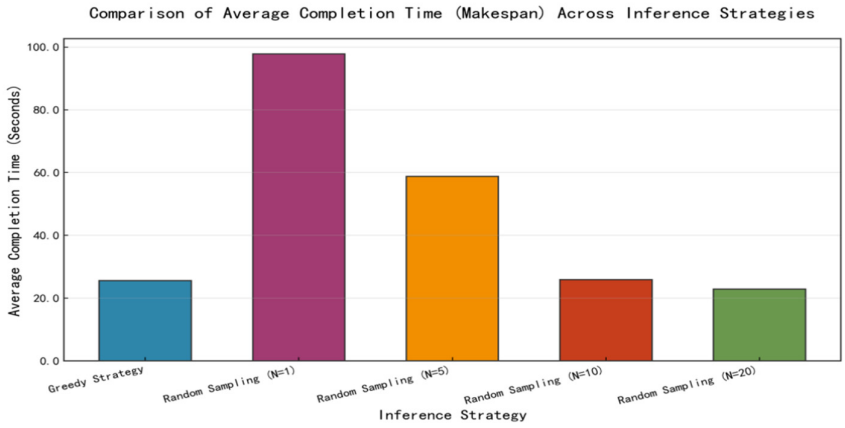

Figure 15. Comparison chart of average makespan under different inference strategies.

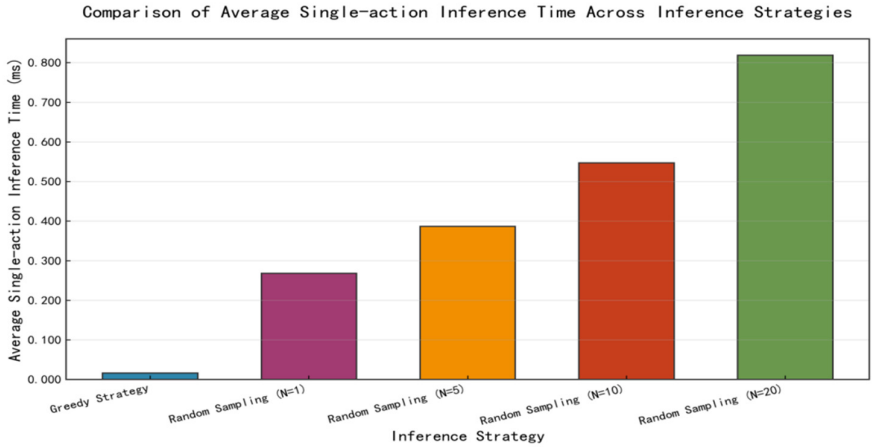

Figure 16. Comparison chart of average inference time per action under different inference strategies.

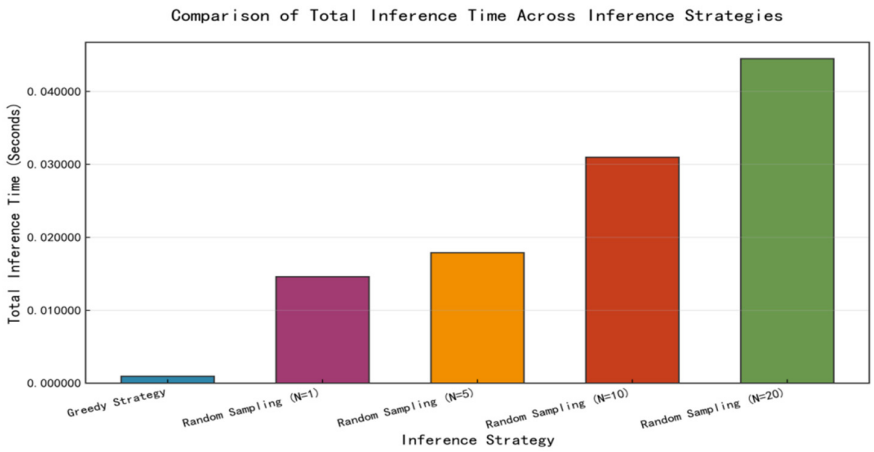

Figure 17. Comparison chart of total inference time under different inference strategies.

The results show that both strategies achieve a 100% task success rate, but exhibit significantly differentiated characteristics in the balance between time efficiency and decision quality, providing a scientific basis for strategy selection in different scenarios.

The greedy strategy demonstrates a strong advantage in inference efficiency: its average inference time per action is only 1.66×10^{-2} ms, and the total inference time is as low as 9.57×10^{4} s, which is far superior to all stochastic sampling strategies (with average inference time per action ranging from 2.68×10^{-1} ms to 8.19×10^{1} ms and total inference time from 1.46×10^{2} s to 4.45×10^{2} s). Meanwhile, the average makespan of the greedy strategy is 25.6, which is close to the 25.9 of the stochastic sampling strategy at N = 10. It achieves extremely low computational overhead while ensuring decision quality, verifying its applicability in scenarios with high real-time requirements and resource constraints.

The stochastic sampling strategy presents a trade-off relationship among sampling number, makespan, and inference overhead. With the increase in the sampling number, N, the makespan is continuously optimized: the makespan reaches as high as 97.8 at N = 1 and drops to 22.9 at N = 20, a reduction of 10.5% compared with the greedy strategy, demonstrating the potential to improve decision quality by expanding the candidate action space. However, the inference overhead increases synchronously at the same time: at N = 20, the average inference time per action and total inference time are 49.3 times and 46.5 times those of the greedy strategy, respectively, indicating that the stochastic sampling strategy needs to trade computational resource consumption for better task scheduling effects. Notably, the stochastic sampling strategy at N = 10 already achieves a makespan comparable to that of the greedy strategy (25.9 vs. 25.6), but the inference overhead is still significantly higher than the latter; while the strategy at N = 20 further optimizes the makespan, and it leads to a substantial increase in computational cost.

In summary, the greedy strategy is suitable for scenarios with stringent real-time and resource constraints due to its high efficiency and low consumption characteristics, while the stochastic sampling strategy can flexibly adapt to the demand of prioritizing decision quality through the adjustment of sampling number. The experimental results clearly reveal the performance of different inference strategies, providing a key reference for the multi-agent task allocation system to select strategies according to the demand priorities of practical application scenarios and improving the flexibility and practicality of system deployment.

The proposed algorithm also exhibits favorable generalization ability in three-dimensional dynamic environments and heterogeneous kinematic models.

The good generalization of the algorithm to 3D environments stems from the following:

(1) Relative coordinate features are adopted in the observation space design. The 3D space only requires adding a relative z-coordinate based on the relative x and y coordinates, without modifying the network structure.
(2) The LSTM with attention fusion network captures the collaboration dependencies and temporal correlations between tasks and UAVs, which are independent of spatial dimensions. Thus, the core scheduling logic is not affected by the expansion of environmental dimensions.
(3) The decentralized decision-making architecture allows UAVs to autonomously adjust task selection strategies according to 3D spatial distances, without the intervention of a global scheduling center.

The core reasons for the algorithm's adaptability to heterogeneous kinematic models are as follows:

(1) The observation space contains the feature of remaining flight time for each UAV, which can reflect the flight efficiency differences caused by heterogeneous speeds in real time and provide an accurate decision-making basis for the policy network.
(2) The local dynamic countdown in the cascaded submission timeout mechanism is calculated based on the estimated flight time, which can adapt to the speed characteristics of different UAVs and avoid deadlock caused by heterogeneous kinematics.
(3) The policy network of the PPO algorithm outputs actions through a probability distribution, which can dynamically adjust task selection preferences according to the environmental feedback of heterogeneous kinematics, thereby realizing the adaptive optimal allocation of resources.

6. Conclusions

Aiming at the core challenges in task allocation for heterogeneous UAV systems, including over-reliance on centralized scheduling, training deadlock, inadequate capture of temporal collaboration, and unstable training under sparse reward conditions, this paper proposes a distributed task allocation algorithm based on improved reinforcement learning, which provides an efficient solution for UAV collaborative scheduling in complex scenarios.

The algorithm achieves performance breakthroughs through four core innovations: the decentralized decision-making architecture breaks free from the reliance on a global scheduling center, enabling UAVs to form collaborative groups autonomously and adapt to the requirements of distributed deployment; the cascaded commit timeout mechanism achieves a zero deadlock rate without additional communication overhead via local dynamic countdown; the LSTM + attention fusion network effectively captures temporal correlations and collaborative dependencies, supporting zero-shot generalization for UAVs and tasks of arbitrary scales; the sparse reward training mechanism based on PPO improves sample utilization efficiency and training stability through clipped probability ratio and multi-round sampling advantage estimation. The experimental results show that the algorithm achieves a 100% task success rate in four scenarios of different scales, and its core metrics such as makespan, time cost, and waiting time are significantly superior to those of mainstream baseline methods, including the Genetic Algorithm and the Hungarian Algorithm. The robustness tests verify that the algorithm can still maintain a task success rate of over 80% and excellent comprehensive efficiency in extreme scenarios with 50% UAV failure, 50% parameter variation, and 50% dynamic task rate. The analysis of inference strategies provides a flexible reference for strategy selection in different scenarios, further enhancing the engineering practicality of the algorithm.

Although this study effectively solves key problems in the dynamic cooperative scheduling of heterogeneous UAVs, it still has certain limitations. First, the task assignment model does not currently consider the battery capacity constraints of UAVs and the differ-

entiated power consumption requirements of heterogeneous tasks. This study assumes that the battery capacity of UAVs can meet the flight and execution requirements of any task assignment, which is somewhat inconsistent with real-world application scenarios. Second, the existing research is conducted based on a two-dimensional simulation environment and has not been extended to the joint optimization of path planning and task assignment in three-dimensional dynamic environments. Third, the current work is mainly based on simulation environments without physical experiments, and its performance in real-world scenarios needs to be further verified.

In response to the above limitations, future research will be carried out in three aspects. First, introduce battery capacity constraints and a heterogeneous task power consumption model and incorporate indicators such as UAV remaining power, task power consumption, and flight energy consumption into the observation space to improve the engineering practicability of the model. Second, further explore the joint optimization of path planning and task assignment in three-dimensional dynamic environments, as well as the dynamic balance between the efficiency and energy consumption in multi-objective optimization scenarios. Third, carry out physical experimental verification of the proposed algorithm, including the construction of experimental platforms and engineering-oriented optimization, so as to realize the deployment of the algorithm from simulation to reality. This work will provide more comprehensive technical support for the large-scale application of the low-altitude economy and intelligent logistics.

Author Contributions: Conceptualization, P.S. and G.Y.; methodology, G.Y.; software, Y.Z.; validation, Y.Z., X.D. and J.C.; formal analysis, G.Y.; investigation, X.X.; resources, J.Z.; data curation, G.Y.; writing—original draft preparation, X.X.; writing—review and editing, P.S.; visualization, G.Y.; supervision, X.X.; project administration, J.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 datasets generated and analyzed during the study are available from the corresponding author on reasonable request.

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

References

1. Lyu, M.; Zhao, Y.; Huang, C.; Huang, H. Unmanned aerial vehicles for search and rescue: A survey. *Remote Sens.* **2023**, *15*, 3266. [CrossRef]
2. Yang, G.; Mo, Y.; Lv, C.; Zhang, Y.; Li, J.; Wei, S. A dual-layer task planning algorithm based on UAVs-human cooperation for search and rescue. *Appl. Soft Comput.* **2025**, *181*, 113488. [CrossRef]
3. Zhang, F.; Guo, A.; Hu, Z.; Liang, Y. A novel image fusion method based on UAV and Sentinel-2 for environmental monitoring. *Sci. Rep.* **2025**, *15*, 27256. [CrossRef] [PubMed]
4. Bi, W.; Zhang, M.; Chen, H.; Zhang, A. Cooperative task allocation method for air-sea heterogeneous unmanned system with an application to ocean environment information monitoring. *Ocean Eng.* **2024**, *309*, 118496. [CrossRef]
5. Hachiya, D.; Mas, E.; Koshimura, S. A reinforcement learning model of multiple UAVs for transporting emergency relief supplies. *Appl. Sci.* **2022**, *12*, 10427. [CrossRef]
6. Aljohani, M.; Mukkamala, R.; Olariu, S. Delivery of Medical Supplies to Remote Locations via Unmanned Aerial Vehicles: Approaches, Challenges, and Solutions. *Transp. Res. Procedia* **2025**, *84*, 73–80. [CrossRef]
7. Ghaffar, M.A.; Peng, L.; Aslam, M.U.; Adeel, M.; Dassari, S. Vehicle-uav integrated routing optimization problem for emergency delivery of medical supplies. *Electronics* **2024**, *13*, 3650. [CrossRef]
8. Aela, P.; Chi, H.L.; Fares, A.; Zayed, T.; Kim, M. UAV-based studies in railway infrastructure monitoring. *Autom. Constr.* **2024**, *167*, 105714. [CrossRef]
9. Merei, A.; Mcheick, H.; Ghaddar, A. Survey on path planning for UAVs in healthcare missions. *J. Med. Syst.* **2023**, *47*, 79. [CrossRef]

10. Song, M.; Cheng, L. Solving the reliability-oriented generalized assignment problem by Lagrangian relaxation and Alternating Direction Method of Multipliers. *Expert Syst. Appl.* **2022**, *205*, 117644. [CrossRef]
11. Amirteimoori, A.; Kia, R.; Mohamed, M.; Weber, G.-W. An innovative framework integrating MILP and a parallel optimal algorithm for UAV-Enabled last-Mile delivery. *Int. J. Prod. Res.* **2025**, *64*, 777–797. [CrossRef]
12. Zhang, J.; Chen, Y.; Yang, Q.; Lu, Y.; Shi, G.; Wang, S.; Hu, J. Dynamic task allocation of multiple UAVs based on improved A-QCDPSO. *Electronics* **2022**, *11*, 1028. [CrossRef]
13. Alqefari, S.; Menai, M.E.B. A Hybrid Method to Solve the Multi-UAV Dynamic Task Assignment Problem. *Sensors* **2025**, *25*, 2502. [CrossRef] [PubMed]
14. Liu, D.; Dou, L.; Zhang, R.; Zhang, X.; Zong, Q. Multi-agent reinforcement learning-based coordinated dynamic task allocation for heterogenous UAVs. *IEEE Trans. Veh. Technol.* **2022**, *72*, 4372–4383. [CrossRef]
15. Freitas, E.P.D.; Basso, M.; Silva, A.A.S.D.; Vizzotto, M.R.; Corrêa, M.S.C. A Distributed Task Allocation Protocol for Cooperative Multi-UAV Search and Rescue Systems. In Proceedings of the 2021 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 15–18 June 2021; pp. 909–917.
16. Yan, H.; Zhao, W.; Chen, C.; You, Y.; Gao, X.; Zhang, D.; Cao, W.; Bao, W. MCTA: Multi-UAV Collaborative Target Allocation to Monitor Targets with Dynamic Importance. In Proceedings of the 2020 6th International Conference on Big Data and Information Analytics (BigDIA), Shenzhen, China, 4–6 December 2020; pp. 50–57.
17. Primatesta, S. Comprehensive Task Optimization Architecture for Urban UAV-Based Intelligent Transportation System. *Drones* **2024**, *8*, 473. [CrossRef]
18. Ompusunggu, V.M.M.O.; Hardhienata, M.K.D.; Priandana, K. Application of Ant Colony Optimization for the Selection of Multi-UAV Coalition in Agriculture. In Proceedings of the 2020 International Conference on Computer Science and Its Application in Agriculture (ICOSICA), Bogor, Indonesia, 16–17 September 2020; pp. 1–8.
19. Song, B.D.; Park, H.; Park, K. Toward flexible and persistent UAV service: Multi-period and multi-objective system design with task assignment for disaster management. *Expert Syst. Appl.* **2022**, *206*, 117855. [CrossRef]
20. Wang, J.F.; Jia, G.W.; Lin, J.C.; Hou, Z.-X. Cooperative task allocation for heterogeneous multi-UAV using multi-objective optimization algorithm. *J. Cent. South Univ.* **2020**, *27*, 432–448. [CrossRef]
21. Han, S.; Özer, B.; Alioğlu, B.; Polat, Ö.; Aktin, A.T. A mathematical model for the delivery routing problem via drones. *Pamukkale Univ. J. Eng. Sci.* **2019**, *25*, 89–97. [CrossRef]
22. Dong, H.; Wu, N.; Feng, G.; Gao, X. Research on computing task allocation method based on multi-UAVs collaboration. In *Proceedings of the 2020 IEEE International Conference on Smart Internet of Things (SmartIoT)*; IEEE: New York, NY, USA, 2020; pp. 86–93.
23. Moon, S.T.; Lee, D.; Lee, D.; Kim, D.; Bang, H. Energy-Efficient Swarming Flight Formation Transitions Using the Improved Fair Hungarian Algorithm. *Sensors* **2021**, *21*, 1260. [CrossRef]
24. Gao, S.; Wu, J.; Ai, J. Multi-UAV reconnaissance task allocation for heterogeneous targets using grouping ant colony optimization algorithm. *Soft Comput.* **2021**, *25*, 7155–7167. [CrossRef]
25. Zhang, X.; Yue, W. Collaborative task allocation for heterogeneous UAV coalitions with resource requirements based on multi-genotype genetic algorithm. *J. Supercomput.* **2025**, *81*, 701. [CrossRef]
26. Li, C.; Li, G.; Liu, Y.; Zheng, Q.; Yang, G.; Liu, K.; Diao, X. Cooperative Task Allocation for Unmanned Aerial Vehicle Swarm Using Multi-Objective Multi-Population Self-Adaptive Ant Lion Optimizer. *Drones* **2025**, *9*, 733. [CrossRef]
27. Xu, S.; Li, L.; Zhou, Z.; Mao, Y.; Huang, J. A task allocation strategy of the UAV swarm based on multi-discrete wolf pack algorithm. *Appl. Sci.* **2022**, *12*, 1331. [CrossRef]
28. Seid, A.M.; Boateng, G.O.; Anokye, S.; Kwantwi, T.; Sun, G.; Liu, G. Collaborative Computation Offloading and Resource Allocation in Multi-UAV Assisted IoT Networks: A Deep Reinforcement Learning Approach. *IEEE Internet Things J.* **2021**, *99*, 12203–12218. [CrossRef]
29. Liu, Z.; Qiu, C.; Zhang, Z. Sequence-to-sequence multi-agent reinforcement learning for multi-UAV task planning in 3D dynamic environment. *Appl. Sci.* **2022**, *12*, 12181. [CrossRef]
30. Dai, W.; Lu, H.; Xiao, J.; Zeng, Z.; Zheng, Z. Multi-robot dynamic task allocation for exploration and destruction. *J. Intell. Robot. Syst.* **2020**, *98*, 455–479. [CrossRef]
31. Yin, Y.; Guo, Y.; Su, Q.; Wang, Z. Task Allocation of Multiple Unmanned Aerial Vehicles Based on Deep Transfer Reinforcement Learning. *Drones* **2022**, *6*, 215. [CrossRef]
32. Zhang, J.; Ren, J.; Cui, Y.; Fu, D.; Cong, J. Multi-USV task planning method based on improved deep reinforcement learning. *IEEE Internet Things J.* **2024**, *11*, 18549–18567. [CrossRef]
33. Liu, D.; Fei, B.; Bao, W.; Zhu, X.; Li, X. DAWN: Dynamic Task Planning of Multi-UAV with Two-Layer Optimization Mechanism in Uncertain Environments. *IEEE Internet Things J.* **2024**, *11*, 37813–37830. [CrossRef]
34. Eser, M.; Yilmaz, A.E. A Gossip-Based Auction Algorithm for Decentralized Task Rescheduling in Heterogeneous Drone Swarms. *IEEE Trans. Aerosp. Electron. Syst.* **2025**, *61*, 6673–6696. [CrossRef]

35. Li, S.; Zhao, Z.; Wang, D.; Li, K.; Liu, G.; Wang, Q. A Reinforcement Learning Framework for Efficient Task Allocation Among AGVs in Smart Warehouse. *IEEE Internet Things J.* **2025**, *12*, 16947–16961. [CrossRef]
36. Abou Houran, M.; Srivastava, G.; Mirza, J.; Ranjha, A.; Javed, M.A.; Zafar, M.H. Centralized Task Allocation for Multiple UAVs in Time-Constraint Industrial IoT Operations. *IEEE Internet Things J.* **2025**, *12*, 37529–37537. [CrossRef]
37. Yan, Y.; Bi, W.; Ma, G.; Zhang, A. Collaborative Task Allocation for Large-Scale Heterogeneous AAV Swarm: A Hierarchical Coalition Formation Game Method. *IEEE Internet Things J.* **2025**, *12*, 27237–27254. [CrossRef]
38. Wan, L.; Li, B.; Sun, L.; Wang, J.; Wang, X.; Xu, G. Cooperative Multi-UAV Jamming in 3D Uncertain Environments Using Multi-Agent Reinforcement Learning. *IEEE Internet Things J.* **2025**, *13*, 8134–8142. [CrossRef]
39. Fu, B.; Smith, W.; Rizzo, D.M.; Castanier, M.; Ghaffari, M.; Barton, K. Robust task scheduling for heterogeneous robot teams under capability uncertainty. *IEEE Trans. Robot.* **2022**, *39*, 1087–1105. [CrossRef]
40. Miuccio, L.; Riolo, S.; Samarakoon, S.; Bennis, M.; Panno, D. On learning generalized wireless MAC communication protocols via a feasible multi-agent reinforcement learning framework. *IEEE Trans. Mach. Learn. Commun. Netw.* **2024**, *2*, 298–317. [CrossRef]
41. Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; Wu, Y. The surprising effectiveness of ppo in cooperative multi-agent games. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24611–24624.