Hyper-DAG-Based Hierarchical Reinforcement Learning for Unifying Data and Resource Dependencies in Multi-UAV Rescue

1st Zilan Li
School of Computer Science and Information Security
Guilin University of Electronic Technology
Guilin, China
https://orcid.org/0009-0009-5774-2666

2nd Xuesong Wang*
School of Electronic Engineering and Automation
Guilin University of Electronic Technology
Guilin, China
wangxuesong@guet.edu.cn

3rd Zhongyi Zhai
School of Computer Science and Information Security
Guilin University of Electronic Technology
Guilin, China
zhaizhongyi@guet.edu.cn

4th Lingzhong Zhao
School of Computer Science and Information Security
Guilin University of Electronic Technology
Guilin, China
zhaolingzhong163@163.com

***Abstract*—Natural disaster rescue missions are characterized by extreme urgency, high complexity, and dynamic environmental uncertainties. In such scenarios,a fleet of Unmanned Aerial Vehicles (UAVs) act as critical assets for search and rescue, supply delivery, and communication recovery. However, traditional task scheduling approaches, often based on standard Directed Acyclic Graphs (DAGs), struggle to effectively model the implicit "resource coupling" (e.g., shared energy budgets and channel interference) alongside explicit data dependencies. This limitation frequently leads to task failures and resource imbalances under dynamic constraints. To address these challenges, this paper proposes a novel scheduling framework based on Hyper-DAG (Hypergraph Directed Acyclic Graph) and Hierarchical Reinforcement Learning (HRL). First, we introduce a Hyper-DAG modeling approach that unifies data, energy, and channel dependencies into a high-order graph structure, utilizing hyperedges to capture complex one-to-many resource constraints. Second, we develop an HRL-based multi-UAV scheduler integrated with the Multi-Agent Proximal Policy Optimization (MAPPO) algorithm. This scheduler employs a graph attention mechanism to dynamically perceive critical dependency bottlenecks and generates optimal task execution sequences in real-time. Extensive experiments across five distinct disaster scenarios (earthquake, flood, wildfire, landslide, and typhoon) demonstrate that our method significantly outperforms baseline algorithms. The proposed framework effectively satisfies the "Golden Rescue Window" time constraints, ensures balanced energy consumption (Gini coefficient < 0.3), and maintains high system reliability (failure rate $< 5\%$).**

***Index Terms*—Multi-UAV Collaboration, Task Scheduling, Hyper-DAG, Hierarchical Reinforcement Learning, Disaster Rescue.**

This work was supported in part by the Guangxi Natural Science Foundation of China, Nos.2023GXNSFAA026270, 2024GXNSFAA010340, National Natural Science Foundation of China, Nos.62262009, 62495083.

*Corresponding author.

I. Introduction

Frequent natural and human-induced disasters necessitate rapid and efficient post-disaster emergency response to reduce casualties and losses [1]. Leveraging agile deployment and communication relay capabilities, multi-UAV swarms can substantially enhance the efficiency of coordinated rescue operations [2]. However, under heterogeneous constraints—such as the "golden rescue window" in earthquakes, flood-induced channel fluctuations, and wildfire-driven rapid energy depletion—task scheduling is jointly constrained by data precedence relations and coupled resource dependencies (e.g., energy or channel constraints), further compounded by environmental uncertainty and multi-objective conflicts.

Although UAVs offer significant advantages in disaster response, efficient task scheduling is still constrained in three challenges. First, dependency structures are complex: conventional DAGs describe only data precedence [3]–[5], and have difficulty capturing heterogeneous dependencies such as resource coupling induced by shared energy budgets and channel interference [6]–[8]. Second, the environment is highly dynamic: nonlinear battery depletion and stochastic SNR variations caused by smoke and terrain changes make static policies hard to adapt to abrupt disturbances and prone to task failures [8], [9]. Third, objectives are intrinsically conflicting: meeting the "golden rescue window" often requires intensive resource utilization, which is at odds with energy balancing and system reliability, thus calling for adaptive decision-making mechanisms with structural awareness and principled multi-objective trade-offs [10]–[12].

This paper proposes a **Hyper-DAG based Hierarchical Reinforcement Learning (HD-HRL)** framework to address the aforementioned challenges. Our contributions are summarized as follows:

1) **Hyper-DAG Modeling:** We propose a Hyper-DAG formulation that applies hypergraph theory to UAV task scheduling. By introducing distinct hyperedges for data, energy, and channel dependencies, we unify explicit precedence constraints and implicit resource couplings into a single mathematical model. A specialized Hyper-DAG encoder is designed to extract high-order structural features.
2) **Hierarchical Scheduling Algorithm:** We develop a Hierarchical Reinforcement Learning framework combined with MAPPO (Multi-Agent Proximal Policy Optimization). This separates the decision-making process into high-level task sorting (guided by a graph attention mechanism that focuses on critical dependencies) and low-level execution. This structure effectively handles the large action space and balances multi-objective conflicts.
3) **Comprehensive Scenario Validation:** We rigorously evaluate our method across five simulated disaster scenarios: Earthquake, Flood, Wildfire, Landslide, and Typhoon. Experimental results show that our approach achieves a 96.4% task success rate and maintains an energy consumption Gini coefficient below 0.3, significantly outperforming state-of-the-art baselines in terms of efficiency and robustness.

The remainder of this paper is organized as follows. Section II reviews related work in DAG scheduling and RL for UAVs. Section III details the system architecture and the Hyper-DAG problem formulation. Section IV presents the proposed HD-HRL methodology, including the encoder design and training algorithm. Section V analyzes the experimental results across different disaster scenarios. Finally, Section VI concludes the paper.

II. Related Work

A. Task Scheduling and DAG Scheduling Algorithms

Task scheduling based on directed acyclic graphs (DAGs) is a classical problem in parallel and distributed systems. Representative approaches include Kahn's topological sorting algorithm [3], the Critical Path Method (CPM) by Kelley and Walker [4], and the Heterogeneous Earliest Finish Time (HEFT) algorithm proposed by Topcuoglu *et al.* [5]. These methods typically assume that computing and communication resources are stable and known, and are therefore mainly suitable for static settings. However, this assumption often breaks down in multi-UAV disaster rescue. Conventional DAG models primarily describe data dependencies and perform task ordering and mapping accordingly (e.g., Pegasus [13] and dependent-task offloading models [6]), but they have difficulty capturing resource coupling such as shared energy budgets and channel interference. Wei *et al.* [7] highlight the necessity of modeling such resource dependencies by jointly optimizing channel allocation and offloading decisions. Moreover, battery depletion and link fluctuations make the environment highly dynamic. Sun *et al.* [8] point out that post-disaster air–ground networks are simultaneously constrained by energy and computing resources, and Luan *et al.* [9] also show that parameter variations can undermine the effectiveness of static schedules. In addition, classical methods usually optimize a single metric (e.g., minimizing makespan) [5], [9], whereas rescue missions require joint trade-offs among latency, energy fairness, and system reliability. Akter *et al.* [10] and Zhang *et al.* [11] provide useful modeling and solution insights from the perspective of multi-objective optimization.

B. Reinforcement Learning in UAV Scheduling

Reinforcement learning (RL) learns decision policies through interactions with the environment and has been widely applied to online decision-making problems such as UAV task offloading, trajectory planning, and task allocation. Hortelano *et al.* [12] survey RL and deep reinforcement learning (DRL) techniques for computation offloading and indicate that they can acquire adaptive offloading policies under time-varying conditions via trial-and-error learning. In single-agent settings, Deep Q-Network (DQN) and Deep Deterministic Policy Gradient (DDPG) are commonly used as baseline methods. Luo *et al.* [14] employ DQN to jointly optimize offloading and trajectory planning so as to reduce search uncertainty, whereas Xu *et al.* [15] develop an improved DDPG scheme to address continuous-action joint optimization of offloading and trajectory control in UAV-assisted mobile edge computing (MEC).

In multi-UAV scenarios, centralized training and decentralized execution (CTDE) has become a prevailing paradigm. Wei *et al.* [7] adopt MADDPG to enable collaborative decision-making for joint task offloading and channel resource allocation, and Wu *et al.* [16] leverage PG-MAPPO to improve cooperation efficiency for dynamic task allocation in highly time-varying settings such as maritime rescue. Despite these advances, existing RL-based scheduling approaches still suffer from two prominent limitations: (i) insufficient structural awareness, as dependency graphs are often flattened into generic state vectors, and (ii) combinatorial action-space explosion as task scales grow, which makes training unstable and difficult to converge without explicit structural guidance.

C. Graph Neural Networks and Hypergraph Modeling

Graph neural networks (GNNs) are a central tool for representation learning on structured data. The Graph Convolutional Network (GCN) proposed by Kipf and Welling [17] learns node representations through neighborhood aggregation, while the Graph Attention Network (GAT) introduced by Velickovic *et al.* [18] enhances expressiveness on complex topologies via attention mechanisms. In task scheduling, tasks and their dependencies are often modeled as graphs; GNNs are then used to extract embeddings that serve as state features for reinforcement learning (RL) to support decision-making. For example, Liu *et al.* [19] incorporate GAT into dynamic DAG scheduling and combine it with deep RL (DRL) to enable online scheduling.

Standard GNNs are designed for simple graphs and mainly capture binary relations, making it difficult to directly represent higher-order correlations where a single constraint simultaneously affects multiple tasks (e.g., a shared energy budget or interference induced by channel reuse) [20]. Hypergraphs allow a hyperedge to connect an arbitrary number of nodes, providing a natural way to model multi-way interactions and group-level constraints [21]. In hypergraph representation learning, Feng *et al.* [22] propose HGNN to perform feature propagation on hypergraphs and encode higher-order correlations; Yadati *et al.* [23] introduce HyperGCN, which integrates hyperedges into convolutional propagation; Zhang *et al.* [24] present Hyper-SAGNN, using self-attention to handle various types of hypergraphs and achieving superior performance; and Huang and Yang [25] propose UniGNN, a unified framework that generalizes multiple GNN message-passing mechanisms to hypergraphs. From an application perspective, hypergraphs have been validated in group-interaction scenarios such as social networks and recommender systems. Antelmi *et al.* [21] summarize their application landscape in social relation modeling and recommendation, and Yu *et al.* [26] propose a self-supervised multi-channel hypergraph convolutional network (MHCN) to model higher-order user relations in social recommendation and improve performance. In contrast, the systematic use of hypergraphs in UAV task scheduling remains largely underexplored.

III. System Architecture and Problem Formulation

A. System Overview

We consider a collaborative multi-UAV rescue system operating in a complex, heterogeneous disaster environment. As illustrated in Fig. 1, the proposed framework is organized into three logically coupled layers: the Physical Layer, the Hyper-DAG Layer, and the Decision Layer.

1) Physical Layer (Heterogeneous Constraints): The top layer of Fig. 1 depicts the physical reality of the rescue mission. The system involves a cluster of N UAVs deployed across different disaster zones.

- **Wildfire Scenario (Left):** In high-temperature zones, UAVs (indicated with low-battery icons) experience non-linear energy decay. Tasks executed here are strictly constrained by the residual energy budget.
- **Flood Scenario (Right):** In waterlogged areas, signal propagation is unstable. UAVs (indicated with signal wave icons) face stochastic SNR fluctuations, affecting real-time data transmission.

These physical states $S_k(t) = \langle C_k, B_k(t), \eta_k(t) \rangle$ (Computation, Battery, SNR) are continuously monitored and fed into the abstract modeling layer.

2) Hyper-DAG Layer (Unified Dependency Modeling): To bridge the gap between physical resource constraints and logical task execution, we introduce the **Hyper-DAG Layer** (Middle layer of Fig. 1). Unlike traditional graphs that only model linear data flow ($v1 \to v2$), our model utilizes **Hyperedges** (colored regions) to capture one-to-many resource couplings:

- **Data Dependency (Black Arrows):** Standard directed edges representing hard precedence (e.g., $v1$ must complete before $v2$ starts).
- **Energy Hyperedge (Orange Blob):** As shown in the figure, nodes $v4, v5, v6$ are encapsulated in an orange region. This implies these tasks share a critical energy budget derived from the wildfire zone UAVs.
- **Channel Hyperedge (Blue Blob):** Similarly, nodes $v8, v9, v10$ are grouped in a blue region, indicating they compete for the limited communication bandwidth in the flood zone.

3) Decision Layer (HRL-Based Scheduling): The bottom layer represents the decision-making brain. The structured information from the Hyper-DAG is processed by a **Hyper-DAG Encoder** (using Graph Attention Networks) to generate a high-dimensional state embedding. This embedding is then passed to the **HRL Scheduler** (Actor-Critic network) powered by the MAPPO algorithm. The scheduler outputs a joint policy consisting of:

1) **Task Sequence:** The optimal execution order (e.g., prioritizing $v4$ to release energy resources).
2) **Offloading Decision:** Selecting the best compute node for each task among Local execution, Peer-to-Peer offloading, or Base Station (BS) offloading.

Finally, the execution results are fed back to the physical layer via a "State Update" loop, enabling closed-loop dynamic control.

The physical environment is highly dynamic and scenario-dependent. For each UAV u_k, its state at time t is modeled as a tuple $S_k(t) = \langle C_k, B_k(t), \eta_k(t) \rangle$, where:

- C_k: The computation capacity (CPU cycles/s).
- $B_k(t)$: The residual battery energy. In high-temperature scenarios like **Wildfires**, energy consumption rates accelerate non-linearly due to cooling requirements.
- $\eta_k(t)$: The Signal-to-Noise Ratio (SNR). This is modeled as a stochastic process $\eta_k(t) \sim \mathcal{N}(\mu(t), \sigma_{env}^2)$, where $\mu(t)$ fluctuates significantly in **Flood** or **Landslide** scenarios due to water vapor absorption and terrain obstruction.

B. Hyper-DAG Modeling

To address the "Resource Coupling" challenge, we generalize the traditional DAG into a **Hyper-DAG**, represented by a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here, $\mathcal{E}$ is the set of hyperedges classifying three distinct dependency types:

1) Task Attributes: Each task $v_i \in \mathcal{V}$ is defined by $\phi_i = \langle w_i, d_i, p_i, E_i^{req}, \eta_i^{req} \rangle$, representing computational workload (cycles), data size (bytes), priority weight $p_i \in [0, 1]$, minimum energy requirement E_i^{req}, and minimum SNR requirement η_i^{req}, respectively.

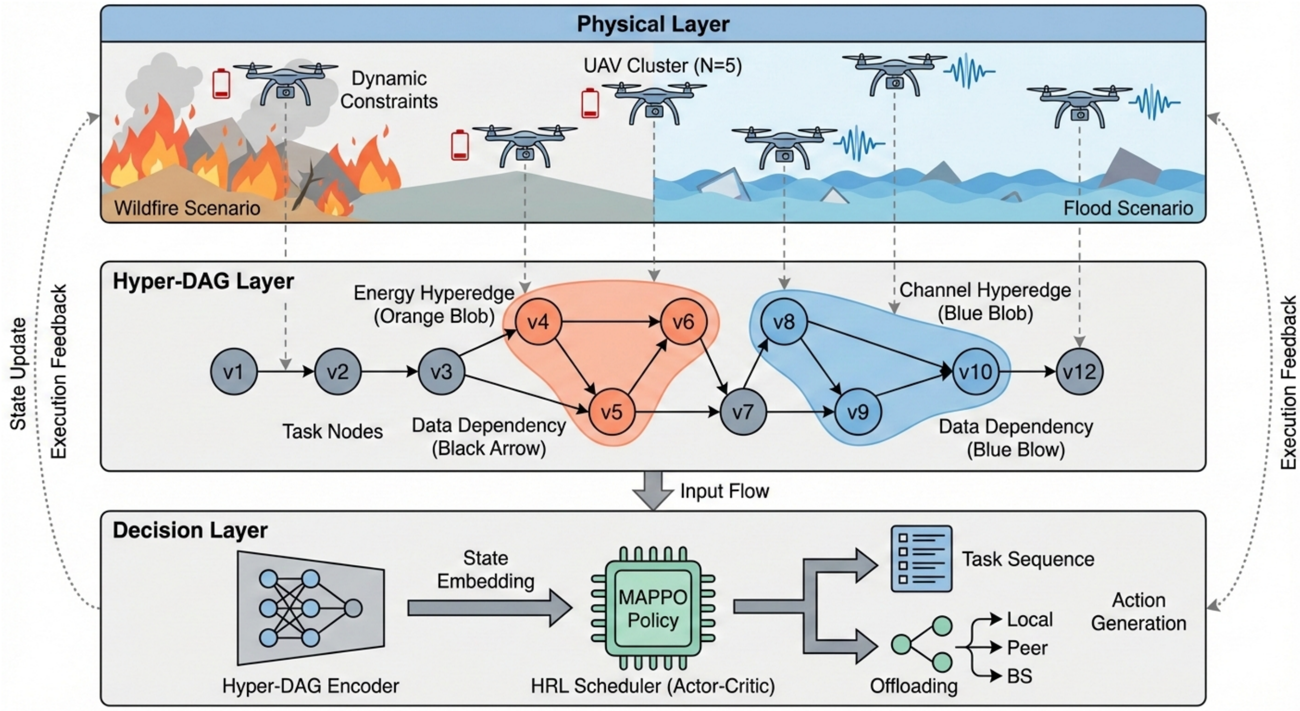


Fig. 1. The proposed three-layer system architecture for multi-UAV disaster rescue. **(Top) Physical Layer:** UAVs operate in diverse scenarios with dynamic constraints. The left side depicts a **Wildfire** scenario where UAVs face rapid battery depletion (Energy constraints), while the right side shows a **Flood** scenario where signal stability fluctuates (Channel constraints). **(Middle) Hyper-DAG Layer:** The mission is modeled as a high-order graph where tasks are coupled not only by data precedence (Black arrows) but also by resource hyperedges. The **Orange Blob** represents an Energy Hyperedge grouping tasks constrained by the limited battery of fire-monitoring UAVs, and the **Blue Blob** represents a Channel Hyperedge for tasks requiring stable communication. **(Bottom) Decision Layer:** The state information is encoded by the Hyper-DAG Encoder and fed into the MAPPO-based HRL Scheduler to generate the optimal Task Sequence and Offloading decisions (Local/Peer/BS).

2) Unified Dependency Hyperedges: We define the edge set $\mathcal{E} = \mathcal{E}_d \cup \mathcal{E}_e \cup \mathcal{E}_c$:

1) **Data Dependency ($\mathcal{E}_d$):** Represents hard precedence constraints (e.g., "Infrastructure Assessment" must precede "Communication Recovery" in an **Earthquake**). A directed edge $(v_i, v_j) \in \mathcal{E}_d$ implies:

$$\text{Start}(v_j) \geq \text{Finish}(v_i) \tag{1}$$

2) **Energy Dependency ($\mathcal{E}_e$):** A hyperedge $e \in \mathcal{E}_e$ connects a subset of tasks that compete for the energy budget of specific UAVs. This explicitly models constraints where executing a high-energy task (e.g., continuous monitoring) creates a bottleneck for subsequent tasks. A task v_i is feasible only if:

$$\exists u_k \in \mathcal{U}, \quad B_k(t) \geq E_i^{req} \tag{2}$$

3) **Channel Dependency ($\mathcal{E}_c$):** Similarly, tasks requiring real-time transmission (e.g., **Typhoon** live streaming) are coupled by channel quality constraints. A task is feasible only if:

$$\exists u_k \in \mathcal{U} \cup \{BS\}, \quad \eta_k(t) \geq \eta_i^{req} \tag{3}$$

C. Dynamic Executability Analysis

Unlike static scheduling, the set of executable tasks $\mathcal{A}_{exec}(t)$ in our Hyper-DAG evolves dynamically. A task is executable only when its topological predecessors are cleared *and* system resources currently satisfy its hyperedge constraints. The filtering logic is formalized in **Algorithm 1**.

D. Multi-Objective Optimization Formulation

We formulate the problem as a Constrained Markov Decision Process (CMDP). The objective is to maximize cumulative rewards while satisfying strictly defined Key Performance Indicators (KPIs).

To balance conflicting objectives (Speed,Balance and Reliability), we define a composite reward function R_t:

$$R_t = r_{time} + r_{bal} + r_{prio} - r_{pen} \tag{4}$$

1) Time Efficiency:

$$r_{time} = \alpha_1 \cdot \left(2.0 - \frac{t}{T_{max}}\right) \tag{5}$$

2) Energy Balance (Gini Coefficient): To prevent single-point failures in long-duration missions, we enforce load balancing using the Gini coefficient:

$$r_{bal} = \alpha_2 \cdot \left(1.0 - \frac{G(\mathbf{E}_{cons})}{\xi_{thresh}}\right), \quad G(\mathbf{E}) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}|e_i - e_j|}{2N\sum_{i=1}^{N} e_i} \tag{6}$$

where $\xi_{thresh} = 0.3$ is the acceptable imbalance threshold.

Algorithm 1 Hyper-DAG Based Executable Task Filtering

Require: Hyper-DAG $\mathcal{G}$, Completed Set $\mathcal{D}(t)$, UAV States $S(t)$
Ensure: Executable Task Set $\mathcal{A}_{exec}(t)$
$\mathcal{A}_{exec}(t) \leftarrow \emptyset$
for each task $v_i \in \mathcal{V} \setminus \mathcal{D}(t)$ **do**
  // Step 1: Check Data Dependency
  if $\exists v_j \in \text{Pred}(v_i)$ such that $v_j \notin \mathcal{D}(t)$ **then**
    continue
  end if
  // Step 2: Check Resource Constraints (Energy & SNR)
  $flag \leftarrow$ False
  for each UAV $u_k \in \mathcal{U}$ **do**
    if $B_k(t) \geq E_i^{req}$ **and** $\eta_k(t) \geq \eta_i^{req}$ **then**
      $flag \leftarrow$ True; **break**
    end if
  end for
  // Check Base Station (Assumed infinite energy)
  if not $flag$ **and** $\eta_{BS} \geq \eta_i^{req}$ **then**
    $flag \leftarrow$ True
  end if
  if $flag$ is True **then**
    Add v_i to $\mathcal{A}_{exec}(t)$
  end if
end for
return $\mathcal{A}_{exec}(t)$

3) Priority & Penalty:

$$r_{prio} = \alpha_3 \cdot p_{a_t}, \quad r_{pen} = \alpha_4 \cdot \sum_{k=1}^{N} \mathbb{I}\left(\frac{B_k(t)}{B_{max}} < 0.2\right) \tag{7}$$

Here, $\mathbb{I}(\cdot)$ is the indicator function, heavily penalizing the system if any UAV enters a low-battery state ($< 20\%$).

IV. Hyper-DAG Based Dependent Task Offloading

The proposed framework adopts a Hierarchical Reinforcement Learning (HRL) architecture to handle the complexity of Hyper-DAGs. As shown in the system overview (Fig. 1), the decision process is decomposed into three logical stages: (1) **Structure Learning** (Encoder), (2) **Task Prioritization** (Scheduler), and (3) **Resource-Aware Offloading** (Execution).

A. Hyper-DAG Structure Learning

Effective scheduling requires understanding both the topology and the resource bottlenecks. For instance, in a **Wildfire**, a cluster of tasks might rely on a specific UAV that is running low on battery. To capture this, we design a Hyper-DAG Encoder.

1) Feature Embedding: We first map the raw features $\mathbf{x}_i$ of task v_i and the attributes of hyperedges e_{ji} into a latent space using Multi-Layer Perceptrons (MLPs):

$$\mathbf{h}_i^{(0)} = \text{MLP}_{node}(\mathbf{x}_i), \quad \mathbf{e}_{ji} = \text{MLP}_{edge}(\text{OneHot}(type_{ji})) \tag{8}$$

where $type_{ji} \in \{\text{Data}, \text{Energy}, \text{Channel}\}$ allows the model to distinguish between strict precedence and resource coupling.

2) Dependency-Aware Attention: We employ a Multi-Head Attention mechanism to aggregate dependency information. This is crucial for dynamic scenarios like **Floods**, where channel instability makes "Channel Dependency" edges critical. The attention mechanism updates task embeddings by focusing on the most constraining predecessors:

$$\mathbf{H}^{(1)} = \text{MultiHeadAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \odot \mathbf{M}_{mask} \tag{9}$$

where $\mathbf{M}_{mask}$ ensures that the scheduler ignores completed tasks, focusing only on active dependencies.

B. HRL-Based Task Prioritization

The high-level Actor determines the optimal execution sequence. It takes the graph embedding $\mathbf{H}^{(1)}$ and the global state $\mathbf{s}_{global}$ (including UAV battery maps and mission progress) as input.

1) Policy Generation with Action Masking: The fused state representation is passed to the policy network π_θ. To guarantee topological validity, we apply an **Action Mask** derived from Algorithm 1:

$$\pi_\theta(a_t = v_i | s_t) = \text{Softmax}\left(\begin{cases} l_i & \text{if } v_i \in \mathcal{A}_{exec}(t) \\ -\infty & \text{otherwise} \end{cases}\right) \tag{10}$$

where l_i is the logit output. This ensures that the agent never selects a task blocked by data or resource constraints (e.g., waiting for data in an **Earthquake** assessment chain).

C. Resource-Aware Offloading Strategy

Once task v_i is selected, the low-level execution layer determines *where* to process it. We adopt a heuristic strategy robust to disaster dynamics:

- **Local Execution (Energy-Sufficient Mode):** Preferred for continuous monitoring tasks (e.g., in **Typhoon** scenarios) to save bandwidth, provided the UAV has sufficient energy ($B_k > 10$ Wh).
- **Base Station Offloading (High-SNR Mode):** Optimal for computation-heavy tasks like 3D terrain modeling (**Landslide**), but only feasible if the channel quality is stable ($\eta_k > 15$ dB).
- **Peer-to-Peer Offloading (Relay Mode):** Used in extreme conditions like **Wildfires**, where a UAV may suffer from both low battery (due to heat) and poor connectivity (due to smoke). The task is offloaded to a neighbor UAV to prevent single-point failure.

D. MAPPO Training

The framework is trained using the MAPPO algorithm. The Critic network estimates the global value function to reduce variance. The loss function includes an entropy term to encourage exploration, preventing the agent from getting stuck in local optima (e.g., repeatedly executing low-priority tasks):

$$\mathcal{L} = \mathcal{L}_{actor}(\theta) + 0.5\mathcal{L}_{critic}(\phi) - \lambda_{ent}\mathcal{H}(\pi_\theta) \tag{11}$$

where λ_{ent} is the regularization coefficient ensuring policy diversity in complex search spaces.

E. Methodology Summary

In summary, the proposed HD-HRL framework consists of four stages: physical-state monitoring, Hyper-DAG-based structural encoding, HRL-based task-priority scheduling, and resource-aware task execution. The Hyper-DAG encoder captures high-order dependency bottlenecks, the high-level scheduler determines a feasible task execution sequence under dynamic constraints, and the execution layer selects an appropriate offloading mode for each selected task. Combined with the MAPPO-based training mechanism, these modules are jointly optimized into an offloading framework for multi-UAV disaster rescue.

V. Experiments

A. Experimental Setup

1) Simulation Environment: We implement a unified multi-UAV rescue simulation platform based on Python and PyTorch. The simulation covers a $1000m \times 1000m$ disaster area with $N = 5$ UAVs and $M = 20$ heterogeneous tasks. The task dependencies and environmental constraints (e.g., LoS probability, background noise) vary according to the five specific disaster scenarios described in Section I.

The key simulation parameters are listed in Table I. To reflect real-world heterogeneity, UAV capabilities (CPU, Battery) are initialized using uniform distributions.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Number of UAVs (N)	5
Number of Tasks (M)	20
UAV CPU Frequency	$\mathcal{U}[4.0, 8.0]$ GHz
UAV Battery Capacity	$\mathcal{U}[15.0, 25.0]$ Wh
Transmission Bandwidth	5 MHz
Gaussian Noise Power	-100 dBm
Golden Rescue Window (T_{max})	900 s (15 min)
PPO Hyperparameters	
Learning Rate (Actor/Critic)	3×10^{-4} / 1×10^{-3}
Discount Factor (γ)	0.99
Clip Parameter (ϵ)	0.2
Entropy Coefficient (λ_{ent})	0.01

2) Baselines: To evaluate the proposed HD-HRL framework, we compare it against three representative baselines:

- **FIFO-PPO (Hybrid Heuristic):** Uses a rigid First-In-First-Out rule for sorting and PPO for offloading. This baseline highlights the necessity of *dependency-aware sorting*.
- **DDPG (Continuous RL):** Maps states directly to sorting scores without graph structural awareness. This validates the effectiveness of our *Hyper-DAG structure learning*.
- **DQN (Discrete RL):** Formulates a joint action space. This baseline demonstrates the advantages of our *Hierarchical (HRL)* decomposition in handling large action spaces.

B. Performance Analysis

1) Training Convergence: We first evaluate the learning efficiency. As illustrated in Fig. 2 (Left), our HD-HRL method (Green curve) converges significantly faster and achieves a higher stable reward compared to DDPG and DQN. This improvement is attributed to the Hyper-DAG encoder, which provides a rich structural prior, effectively reducing the search space for the agent. Fig. 2 (Right) confirms that our method achieves a task success rate of **96.4%**, whereas FIFO-PPO saturates at 89.2% due to its inability to resolve complex dependency deadlocks. The narrow shaded region of the HD-HRL curve further demonstrates the stability of our policy.

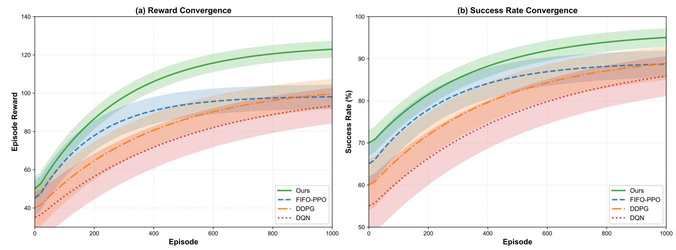


Fig. 2. Training convergence analysis over 1000 episodes. **(Left)** Average Episode Reward, showing HD-HRL's superior learning efficiency compared to baselines. **(Right)** Task Success Rate, where HD-HRL achieves high stability (96.4%) with lower variance (indicated by the shaded regions).

2) Multi-Scenario Robustness: To verify adaptability, Fig. 3 illustrates the normalized performance across five distinct disaster scenarios.

While FIFO-PPO performs adequately in the "Earthquake" scenario (where dependencies are static), its performance collapses in "Wildfire" and "Landslide" scenarios. This is because standard heuristics fail to account for dynamic energy consumption rates and signal fluctuations. In contrast, HD-HRL maintains a consistent full-pentagon shape (Fig. 3), proving the effectiveness of the Attention Mechanism in dynamically attending to critical resource bottlenecks—such as prioritizing Energy hyperedges in Wildfires to prevent node failure.

3) Energy Efficiency Mechanism: A key contribution of this work is the explicit handling of energy coupling. We analyze the internal mechanism in Fig. 4.

As visualized in Fig. 4(a), the FIFO-PPO approach results in unbalanced battery consumption, where UAV 1 and UAV 2 are overloaded and fail early (around Step 65). In contrast, HD-HRL (Right panel of Fig. 4a) effectively distributes the load, resulting in a synchronized battery descent for all 5 UAVs. Fig. 4(b) quantifies this using the Gini coefficient. While FIFO-PPO hovers around 0.46 (High Imbalance), HD-HRL converges to **0.28**, satisfying the strictly defined energy balance constraint (< 0.3).

C. Ablation Study and Case Analysis

To quantify the contribution of each module, we conducted ablation studies as shown in Fig. 5. We compare the complete model against two variants: (1) **w/o Hyper-DAG Encoder**

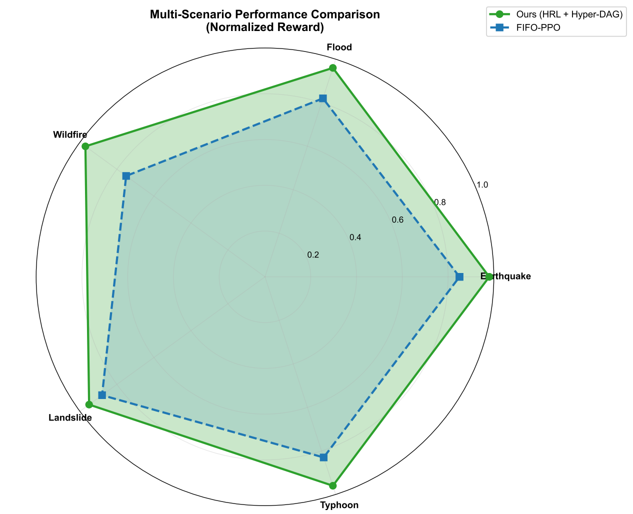
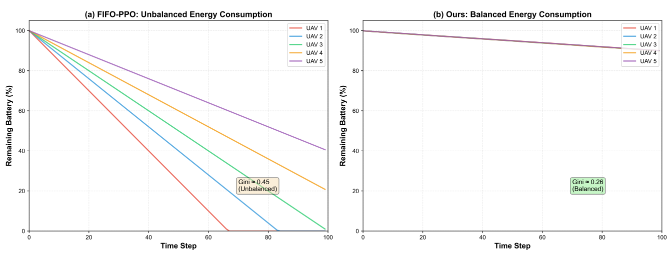
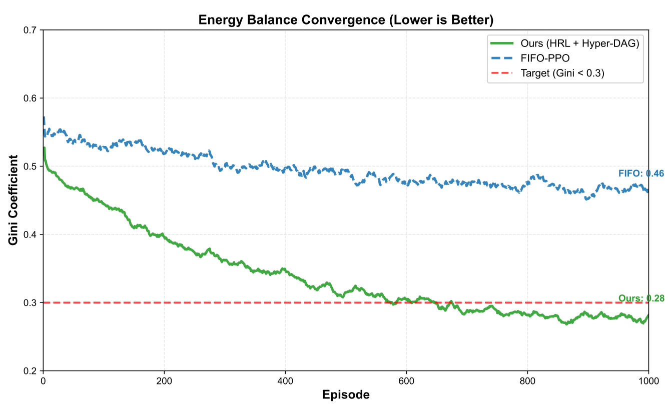


Fig. 3. Normalized performance comparison across five distinct disaster scenarios. The proposed HD-HRL (Green) forms a full pentagon ("Hexagon Warrior"), demonstrating robust adaptability. In contrast, FIFO-PPO (Blue) collapses in resource-constrained scenarios like Wildfire and Landslide.



(a) Battery Depletion Comparison



(b) Gini Coefficient

Fig. 4. Energy efficiency mechanism analysis. **(a)** Unbalanced battery consumption in FIFO-PPO leads to premature node failures (lines hitting zero at Step 65/85). In contrast, HD-HRL achieves synchronized battery descent. **(b)** The Gini coefficient of HD-HRL converges to **0.28** (below the target 0.3), validating effective load balancing.

(using simple MLP), and (2) **w/o Attention Mechanism** (using mean pooling).

The dual-axis plot reveals distinct roles for each component:

- Removing the **Hyper-DAG Encoder** results in a sharp drop in Success Rate (**-3.9%**), indicating that the high-order graph structure is essential for resolving strict precedence constraints and preventing deadlocks.
- Removing the **Attention Mechanism** leads to a significant increase in Average Latency (**+0.7s**, from 8.8s to 9.5s) and a moderate drop in success rate. This suggests that without attention, the model fails to dynamically prioritize the most critical bottlenecks (e.g., a fading channel), leading to inefficient waiting times.

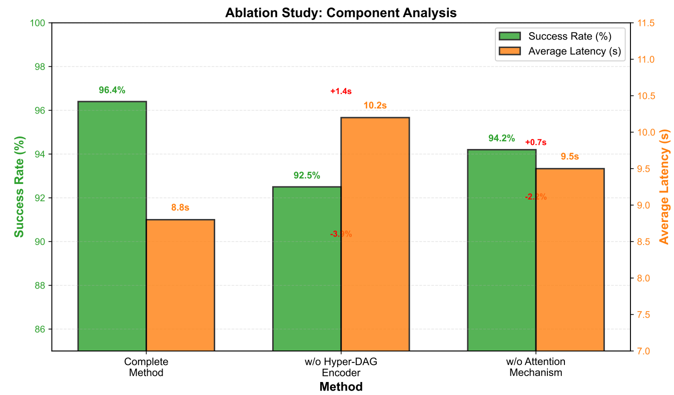


Fig. 5. Ablation study on component contributions using a dual-axis plot. Removing the Hyper-DAG Encoder significantly drops Success Rate (Green Bars), while removing the Attention Mechanism primarily increases Latency (Orange Bars).

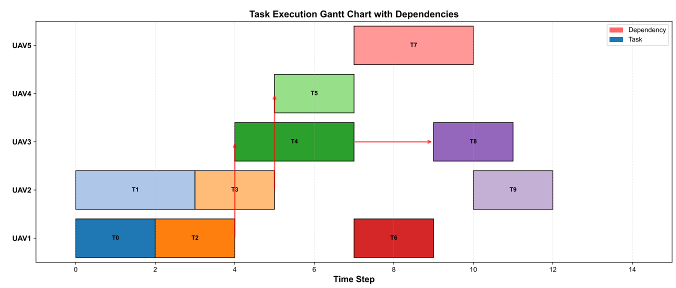


Fig. 6. Task execution Gantt chart for a sample episode. Red arrows indicate strict data dependencies (e.g., T3 → T4 and T4 → T8), confirming that the scheduler strictly adheres to topological constraints while maximizing parallelism across 5 UAVs.

Finally, to demonstrate the logical correctness of our scheduler, we present a Gantt chart of a sample episode in Fig. 6.

The red arrows in Fig. 6 clearly illustrate that task execution strictly follows the topological dependencies defined in the Hyper-DAG (e.g., Task T4 starts strictly after T3 completes), ensuring mission validity. Simultaneously, the chart shows high parallelism across the 5 UAVs, validating the efficiency of our HRL-based scheduling policy.

VI. Conclusion

This paper addresses the challenges of task dependencies, strong resource coupling, and environmental uncertainty in multi-UAV disaster rescue, we propose a unified decision-making framework, termed HD-HRL, based on Hyper-DAG modeling and hierarchical reinforcement learning. By introducing hyperedges, the framework jointly captures high-order resource constraints (e.g., shared energy budgets and channel contention) together with data and priority dependencies, enabling the executable task set to be updated dynamically with the resource state. At the decision layer, a hypergraph encoder with an attention mechanism extracts dependency bottleneck features, and MAPPO learns feasible task-ordering policies under action-masking constraints; resource-aware offloading and dynamic state updates are further integrated to improve robustness. Simulation results demonstrate faster and more stable convergence and better cross-scenario generalization than conventional DAG-based models and representative baselines. Ablation studies confirm the critical role of hypergraph encoding and attention in bottleneck identification, deadlock avoidance, and latency reduction. Future work will explore end-to-end offloading learning and advanced multi-UAV scheduling mechanisms.

References

[1] Y. Wang, Z. Su, N. Zhang, and D. Fang, "Disaster relief wireless networks: Challenges and solutions," *IEEE Wireless Communications*, vol. 28, no. 5, pp. 148–155, 2021.

[2] M. Ishiwatari, "Leveraging drones for effective disaster management: a comprehensive analysis of the 2024 noto peninsula earthquake case in japan," *Progress in Disaster Science*, vol. 23, p. 100348, 2024.

[3] A. B. Kahn, "Topological sorting of large networks," *Communications of the ACM*, vol. 5, no. 11, pp. 558–562, 1962.

[4] J. E. Kelley Jr and M. R. Walker, "Critical-path planning and scheduling," in *Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference*, 1959, pp. 160–173.

[5] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE transactions on parallel and distributed systems*, vol. 13, no. 3, pp. 260–274, 2002.

[6] G. Zhao, H. Xu, Y. Zhao, C. Qiao, and L. Huang, "Offloading dependent tasks in mobile edge computing with service caching," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1997–2006.

[7] Y. Wei, S. Wu, Z. Ji, Z. Yu, C. Jiang, and L. Kuang, "Multi-uav collaborative edge computing algorithm for joint task offloading and channel resource allocation," *Journal of Communications and Information Networks*, vol. 9, no. 2, pp. 137–150, 2024.

[8] G. Sun, L. He, Z. Sun, Q. Wu, S. Liang, J. Li, D. Niyato, and V. C. Leung, "Joint task offloading and resource allocation in aerial-terrestrial uav networks with edge and fog computing for post-disaster rescue," *IEEE Transactions on Mobile Computing*, vol. 23, no. 9, pp. 8582–8600, 2024.

[9] Q. Luan, H. Cui, X. Yu, and B. Sikdar, "Task rescheduling for uav-assisted emergency communications under uncertainty," in *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE, 2023, pp. 3039–3044.

[10] S. Akter, D.-Y. Kim, and S. Yoon, "Task offloading in multi-access edge computing enabled uav-aided emergency response operations," *IEEE Access*, vol. 11, pp. 23 167–23 188, 2023.

[11] H. Zhang, K. Liao, Y. Tai, W. Ma, G. Cao, W. Sun, and L. Xu, "Decentralized and fault-tolerant task offloading for enabling network edge intelligence," *IEEE Systems Journal*, vol. 18, no. 2, pp. 1459–1470, 2024.

[12] D. Hortelano, I. de Miguel, R. J. D. Barroso, J. C. Aguado, N. Merayo, L. Ruiz, A. Asensio, X. Masip-Bruin, P. Fernández, R. M. Lorenzo *et al.*, "A comprehensive survey on reinforcement-learning-based computation offloading techniques in edge computing systems," *Journal of Network and Computer Applications*, vol. 216, p. 103669, 2023.

[13] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. F. Da Silva, M. Livny *et al.*, "Pegasus, a workflow management system for science automation," *Future Generation Computer Systems*, vol. 46, pp. 17–35, 2015.

[14] Q. Luo, T. H. Luan, W. Shi, and P. Fan, "Deep reinforcement learning based computation offloading and trajectory planning for multi-uav cooperative target search," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 504–520, 2022.

[15] F. Xu, S. Wang, W. Su, and L. Zhang, "Joint optimization task offloading and trajectory control for unmanned-aerial-vehicle-assisted mobile edge computing," *Computers and Electrical Engineering*, vol. 111, p. 108916, 2023.

[16] X. Wu, Q. Yan, J. Wang, Y. Zhou, Q. Huang, and C. Jiang, "Dynamic task allocation for uav swarms in maritime rescue scenarios based on pg-mappo," *IEEE Internet of Things Journal*, 2025.

[17] T. N. Kipf and M. Welling, "Semi-supervised learning with graph convolutional networks," 2017.

[18] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio *et al.*, "Graph attention networks," pp. 10–48 550, 2017.

[19] Z. Liu, L. Huang, Z. Gao, M. Luo, S. Hosseinalipour, and H. Dai, "Ga-drl: Graph neural network-augmented deep reinforcement learning for dag task scheduling over dynamic vehicular clouds," *IEEE Transactions on Network and Service Management*, vol. 21, no. 4, pp. 4226–4242, 2024.

[20] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[21] A. Antelmi, G. Cordasco, M. Polato, V. Scarano, C. Spagnuolo, and D. Yang, "A survey on hypergraph representation learning," *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–38, 2023.

[22] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 3558–3565.

[23] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "Hypergcn: A new method for training graph convolutional networks on hypergraphs," vol. 32, 2019.

[24] R. Zhang, Y. Zou, and J. Ma, "Hyper-sagnn: a self-attention based graph neural network for hypergraphs," 2019.

[25] J. Huang and J. Yang, "Unignn: a unified framework for graph and hypergraph neural networks," 2021.

[26] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang, "Self-supervised multi-channel hypergraph convolutional network for social recommendation," in *Proceedings of the web conference 2021*, 2021, pp. 413–424.