

Article

Recoverability-Aware Fault-Tolerant Scheduling of UAVs for IoT Data Collection in Disaster Scenarios

Hailu Xin ¹, Weidong Bao ¹ , Hui Yan ^{2,*}, Ji Wang ¹, Xiaoqing Li ¹ , Yanjie Song ¹  and Lining Xing ³

¹ College of Systems Engineering, National University of Defense Technology, Changsha 410073, China; xinhailu@nudt.edu.cn (H.X.); wdbao@nudt.edu.cn (W.B.); wangji@nudt.edu.cn (J.W.); lixiaqing21@nudt.edu.cn (X.L.); songyj_2017@163.com (Y.S.)

² Information Support Force Engineering University, Wuhan 430019, China

³ School of Electrical & Information Engineering, Jiangsu University of Technology, Changzhou 213001, China; xln@jsut.edu.cn

* Correspondence: yanhui13@nudt.edu.cn

Highlights

What are the main findings?

- A recoverability-aware fault-tolerant scheduling framework is developed for UAV-assisted Internet of Things (IoT) data collection under random UAV failure risks by integrating UAV reliability modeling, Maximum Distance Separable (MDS)-coded backup, and collaborative scheduling.
- The proposed MDS-coded Fault-Tolerant Backup (MFTB) mechanism and Loss-Pressure-guided Data Collection Fault-tolerant Scheduling (LP-DCFS) algorithm reduce irrecoverable data loss and increase recovered data compared with baseline redundancy and scheduling methods under high-load and high-failure conditions.

What are the implications of the main findings?

- Under the modeled independent random crash/non-return faults and the evaluated simulation settings, the results support the joint design of data-layer protection and failure-aware scheduling for disaster-oriented UAV-assisted data collection, rather than reliance on path planning or redundancy enhancement alone.
- Under the modeled fault and resource assumptions, the findings provide simulation-based evidence that the proposed framework can improve UAV-IoT data recovery and motivate future physical UAV experiments under more complex disaster-like conditions.

Abstract

Reliable data collection is essential for disaster-oriented Internet of Things (IoT) systems, where damaged terrestrial communication infrastructure often leaves sensed data buffered at disconnected end devices. In Unmanned Aerial Vehicle (UAV)-Internet of Things device (IoTD) collaborative data collection, random UAV faults and limited energy and buffer resources further complicate mission execution, making fault-tolerant scheduling crucial for robust data recovery. To address these issues, a unified framework is developed by integrating dynamic UAV reliability modeling, Maximum Distance Separable (MDS)-coded fault-tolerant backup, and collaborative scheduling optimization. Within this framework, a data fault-tolerance mechanism, termed MFTB, and a bilevel collaborative scheduling algorithm, termed LP-DCFS, are proposed. Simulation results indicate that, in the evaluated scenarios, the proposed methods achieve better overall performance than the considered baselines. In a representative high-load, high-failure scenario, MFTB reduces data loss



Academic Editors: Bing Chen, Changyan Yi, Kun Zhu, Yutao Jiao and Zhiyi Tian

Received: 23 June 2026

Revised: 23 July 2026

Accepted: 24 July 2026

Published: 27 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

by 4.8% and 37.5% compared with Buffer-Limited Retransmission (BLR) and Replication, respectively, while LP-DCFS increases the amount of recovered data by 33.9%, 32.3%, and 53.1% compared with ACEPSO, ADE-DMRM, and DQN, respectively. Under the modeled independent random crash and non-return faults and the evaluated simulation settings, these results suggest that coordinating failure-risk characterization, data-protection mechanisms, and task-scheduling strategies can improve the robustness and data-recovery capability of disaster-oriented UAV-assisted data collection.

Keywords: Internet of Things; UAV-assisted data collection; fault-tolerant scheduling; MDS coding; data recovery; reliability

1. Introduction

In disaster emergency management and environmental monitoring [1], Internet of Things (IoT) technology enables terminals to continuously collect geographic, meteorological, and disaster-related information, thereby supporting emergency decision-making [2]. However, sudden disasters such as earthquakes, floods, and wildfires often damage communication infrastructure and backhaul links, leaving IoT devices (IoTDs) disconnected from the outside world. As a result, large volumes of monitoring data remain stranded locally, making reliable data aggregation and recovery a central challenge for disaster-oriented IoT systems.

When terrestrial communication infrastructure fails, Unmanned Aerial Vehicle (UAV)-assisted data collection provides a practical solution owing to its deployment flexibility and rapid response capability [3,4]. UAVs can act as aerial relays in regions with disrupted ground links [5], periodically visiting IoTDs and retrieving buffered data. Existing studies have mainly focused on UAV path planning, energy allocation, and multi-UAV cooperative scheduling. However, most of them assume reliable UAV operation and do not explicitly consider random failures in disaster scenarios, limiting their applicability to high-risk and uncertain data collection tasks.

In extreme disaster scenarios, UAVs may terminate missions prematurely due to energy depletion, weather disturbances, or hardware failures, causing data loss and mission interruption. Thus, strategies based only on path optimization or task assignment are insufficient to ensure continuous operation and complete data recovery. Although multi-copy redundancy can improve fault tolerance, it also incurs substantial storage, communication, and energy overhead. A key challenge is therefore to establish a fault-tolerant air-ground scheduling framework that jointly optimizes data recoverability and task execution efficiency for resource-constrained IoT data collection.

Against this background, three challenges remain:

- **Balancing data protection and recoverability under limited buffer resources.** IoTDs are heterogeneous in data arrival rates and buffer capacities, while continuously generated data progressively occupy scarce local storage, increasing the risk of overflow and loss. Although stronger redundancy improves recoverability, it also consumes additional buffer and communication resources.
- **Modeling UAV failure dynamics under energy coupling.** UAV failure risk evolves with mission progression and residual energy over multiple collection rounds. Heterogeneity in endurance, flight cost, and communication load further strengthens the coupling between energy consumption and failure risk, making static-reliability models inadequate.

- **Joint optimization of scheduling and fragment allocation under random failures.** Under random UAV failures, data recoverability depends on redundancy configuration, access timing, visiting order, and fragment carrying and return. The coupling among access decisions, route ordering, and fragment allocation makes the problem a joint optimization of recoverability, execution efficiency, and resource consumption.

To address these challenges, this work develops a recoverability-aware collaborative fault-tolerant scheduling framework that integrates reliability modeling, data protection, and scheduling optimization within a unified decision process. Different from prior UAV fault-tolerant scheduling studies that mainly focus on mission robustness or task continuation after UAV faults, this work emphasizes whether continuously generated and locally buffered data can remain recoverable under UAV non-return risks. The main contributions are as follows:

- A dynamic UAV reliability model is developed by jointly accounting for mean time to failure and energy depletion, capturing reliability evolution over multiple mission rounds and providing a quantitative basis for scheduling under failure constraints.
- A Maximum Distance Separable (MDS)-coded fault-tolerant backup mechanism is proposed for buffer-constrained IoTs. Time-interval data blocks are encoded into fragments and distributed redundantly to preserve recoverability, while an adaptive protection-level decision mechanism adjusts redundancy intensity according to the system state at block formation.
- A joint optimization framework integrating terminal access decisions, visiting-order optimization, and block-level fragment allocation is developed, together with a loss-pressure-guided bilevel fault-tolerant scheduling algorithm, termed LP-DCFS, to jointly optimize data recoverability, task execution efficiency, and resource consumption under random failures.

2. Related Work

Existing studies can be grouped into three categories: optimization problems, scheduling and task assignment methods, and fault tolerance and redundancy design.

Optimization problems: UAV-assisted data collection has been formulated as different optimization problems according to network architecture, decision variables, and resource constraints. Bera et al. [6] modeled UAV data collection as a capacitated vehicle routing problem, in which the collection route is optimized under capacity constraints. Liu et al. [7] studied a UAV-assisted MEC problem by jointly optimizing UAV trajectory, transmit power, and computing frequency under energy constraints. Liu et al. [8] considered a U-MEC network with wireless power transfer and optimized device activation and computation throughput. Zeng et al. [9] formulated large-scale UAV-assisted task offloading as a delay-oriented optimization problem involving device waiting time. Asim et al. [10] investigated an IRS-assisted UAV optimization problem by jointly designing UAV trajectories and IRS phase shifts. These formulations mainly focus on route planning, resource allocation, task offloading, or communication enhancement, whereas recoverability-aware scheduling under random UAV failures has received less attention as a core problem setting.

Scheduling and task assignment methods: Various methods have been proposed for UAV-assisted scheduling and task assignment. Hao et al. [11] formulated UAV-supported data collection as a mixed-integer non-convex problem and solved it using BCD, SCA, and bisection search. Liu et al. [12] addressed deployment-based data collection for wireless sensor nodes using relaxation and Taylor expansion. Lin et al. [13] proposed a parameterized dueling deep Q-network to improve UAV energy efficiency. Zhang et al. [14] formulated UAV trajectory planning as a Markov decision process and solved it with a twin-delayed deep deterministic policy gradient algorithm. Li et al. [15] proposed a

hierarchical deep reinforcement learning framework for dynamic demand-aware multi-UAV scheduling, integrating offline path planning with online resource adjustment for IoT data collection. Xin et al. [16] formulated the joint scheduling of task assignment, path planning, and data processing in post-disaster scenarios and developed a genetic-assisted local optimization algorithm to solve it. Gong et al. [17] combined problem decomposition and particle swarm optimization to reduce the number of UAVs required for sensor collection. Wang et al. [18] proposed an SSF-based ant colony optimization method for joint UAV speed and altitude scheduling.

Fault tolerance and redundancy design: Fault tolerance has received increasing attention in high-risk UAV systems. Yan et al. [19] developed a fault-tolerant scheduling model for heterogeneous UAVs and jointly optimized task throughput and fault tolerance through the UBFT mechanism. Ramezani et al. [20] proposed an air-ground collaborative strategy in which a UGV takes over the mission after UAV disconnection. Li et al. [21] incorporated UAV failure probabilities into multi-UAV path planning and optimized mission completion probability. Recent robust-control and fault-estimation studies have also addressed UAV reliability from the perspective of disturbance reconstruction and actuator-fault estimation. For example, Hernandez-Perez et al. [22] adapted a sliding-mode observer to a fractional-order observer for disturbance reconstruction of a UAV model, while Borja-Jaimes et al. [23] developed a fractional-order sliding-mode observer for actuator fault estimation in a quadrotor UAV. These studies provide useful control-oriented perspectives on UAV resilience under disturbances and actuator faults. In contrast, the present work focuses on recoverability-aware data collection after UAV non-return or crash faults, where the key problem is how to jointly optimize UAV reliability modeling, coded data protection, terminal access, route scheduling, and fragment allocation. At the data level, Sistu et al. [24] analyzed the robustness gain of erasure coding in UAV-based data collection and optimized sensor-side redundancy budgeting, while He et al. [25] proposed a Cauchy-structured MDS coding strategy to adapt redundancy rates over dynamic links. These studies demonstrate the effectiveness of coding-based redundancy in improving data robustness under unreliable communication or collection conditions.

Overall, existing studies have provided important foundations for UAV-assisted data collection and UAV fault tolerance. Planning-oriented studies mainly improve collection efficiency through trajectory design, task allocation, resource management, or delay reduction, while fault-tolerant UAV studies mainly enhance mission robustness under UAV faults through reliability-aware routing, task takeover, fault estimation, or robust control. However, in disaster-oriented IoT data collection, collection efficiency and fault tolerance are not independent objectives. UAV failure risk affects whether continuously generated and locally buffered data can still be preserved and recovered, while redundancy protection and collaborative scheduling jointly determine how limited communication, storage, and flight resources are used under such uncertainty. Therefore, existing studies still lack a recoverability-aware data collection framework that integrates failure-risk characterization, data protection, and multi-UAV collaborative scheduling within a unified decision process.

3. System Model and Problem Formulation

3.1. Motivation Scenario

We consider a post-disaster UAV-IoTD data collection scenario in which terrestrial communication links are disrupted and IoT devices cannot directly report their buffered sensing data to the base station (BS) in Figure 1. At the system-model level, the disaster environment is abstracted by focusing on the data-collection consequences of communication disruption and UAV non-return risks. IoT devices are assumed to be fixed during the scheduling horizon and to continuously generate monitoring data that must be buffered locally before UAV

collection. Multiple UAVs periodically visit IoTDs and return collected data to the BS under energy, storage, and communication constraints. UAV failures are modeled as random crash/non-return events, while terrain profiles, buildings, obstacle blockage, correlated environmental hazards, and adversarial interference are not explicitly represented in the current formulation. The system is organized into a ground layer and an aerial layer.

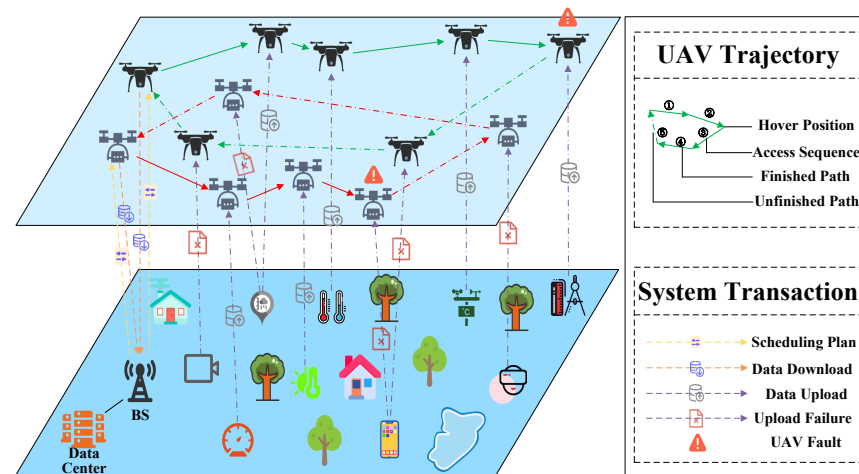


Figure 1. Scenario of UAV-assisted Data Collection under Failure Risks.

The ground layer includes an emergency BS, a data center, and multiple IoTDs distributed across the disaster area. The IoTDs continuously collect environmental information, while the BS, connected to the data center through wired links, handles task dispatching, status aggregation, and centralized data management. When terrestrial links are disrupted, IoTDs rely on UAV visits to upload buffered data.

The aerial layer consists of heterogeneous UAVs with different energy capacities, communication capabilities, and reliability levels. Each UAV departs from the BS, visits designated IoTDs in sequence, and returns after data collection. To improve data recoverability under failures, redundancy protection is incorporated into the collection process. If a UAV terminates its mission prematurely due to energy depletion, weather disturbances, or device abnormalities, unfinished tasks are rescheduled according to the current system state.

A UAV trajectory is represented as a directed path consisting of multiple hovering positions and an associated visiting sequence. Figure 1 schematically illustrates two representative UAV trajectories. The trajectories are colored green and red to distinguish the UAVs, while the numbered arrows in the “UAV Trajectory” legend indicate the visiting order of hovering positions. For both trajectories, solid and dash-dotted segments denote completed and uncompleted portions of the planned routes, respectively, with the latter resulting from UAV faults. The warning symbols and crossed-document symbols indicate UAV faults and upload failures, respectively. Assuming a constant flight altitude, each hovering position corresponds to the location of an IoTD. UAV trajectories are determined by the scheduling strategy under energy, communication, and fault-tolerance constraints.

3.2. System Model

We consider a system with M UAVs and N IoTDs, where the UAV and IoTD sets are $\mathcal{U} = \{u_1, u_2, \dots, u_M\}$ and $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$, respectively. An emergency BS is used for task scheduling and data aggregation. For each IoTD $d_i \in \mathcal{D}$, its state is

$$d_i = (C_i, \tau_i), \quad (1)$$

where $C_i = (x_i, y_i)$ denotes the two-dimensional coordinates of d_i , and τ_i denotes its most recent data-sealing time.

To describe UAV access relationships, the IoTs and the BS are represented as nodes in a graph, as illustrated in the left part of Figure 2. Since a UAV can fly in a straight line between any two nodes, the edge weights are geometric distances, yielding a complete graph that characterizes access ordering and reachability within a mission round.

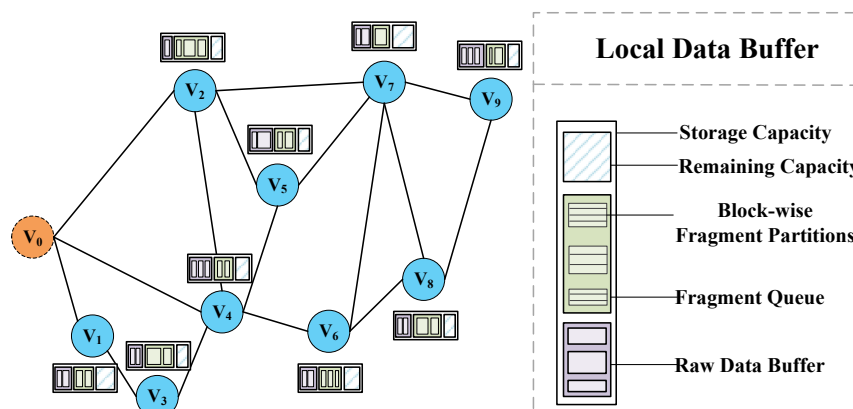


Figure 2. Spatial abstraction of IoT nodes and local buffer structure.

The right part of Figure 2 shows the local buffer structure of an IoT, which consists of a raw buffer, a fragment queue, and the remaining capacity. The raw buffer stores newly generated but uncoded data since the last update, with occupancy L_i^{raw} , while the fragment queue stores coded fragments not yet collected by a UAV or successfully returned to the BS, with occupancy L_i^{frag} . The storage constraint is

$$L_i^{\text{raw}} + L_i^{\text{frag}} \leq L_i^{\text{max}}, \quad (2)$$

where L_i^{max} denotes the local storage capacity of node d_i .

Upon a state refresh, the raw data accumulated since τ_i are sealed into a time-interval data block, encoded by an (n, k) code, and written into the fragment queue, after which τ_i is updated. Thus, fragments of the same data block may be returned in batches by different UAVs over different mission rounds.

For each UAV $u_j \in \mathcal{U}$, its state is

$$u_j = (c_j, h_j, s_j, \varepsilon_j, e_j, f_j), \quad (3)$$

where c_j denotes storage capacity, h_j and s_j denote flight altitude and speed, ε_j denotes rated power, e_j denotes residual energy, and f_j denotes the mean time to failure (MTTF). The mission of u_j is represented by the ordered visiting sequence

$$\Gamma_j = [d_{j,1}, d_{j,2}, \dots, d_{j,L_j}], \quad (4)$$

where L_j denotes the number of visited nodes in one mission period.

During mission execution, UAV u_j must satisfy $e_j \geq e_j^{\min}$ for task completion and safe return. For tractability, all UAVs are assumed to fly at a constant altitude, and all IoT locations are fixed and known. The geometric distance between UAV u_j and IoT d_i is

$$\delta_{i,j} = \|C_i - C_j\|_2, \quad (5)$$

where C_j denotes the ground projection of UAV u_j .

3.3. Communication Model

The UAV–IoT link is assumed to be line-of-sight (LoS) [26], and non-line-of-sight fading is ignored. The channel gain is

$$g_{i,j} = g_0 \left(\frac{h_j^2}{\delta_{i,j}^2 + h_j^2} \right)^{\frac{\alpha}{2}}, \quad (6)$$

where g_0 is the reference channel gain, α is the path-loss exponent, and $\delta_{i,j}$ is the horizontal distance between UAV u_j and IoT d_i . The corresponding transmission rate is

$$R_{i,j} = B_j \log_2 \left(1 + \frac{P_i g_{i,j}}{B_j n_0} \right), \quad (7)$$

where P_i is the uplink transmit power, B_j is the communication bandwidth of UAV u_j , and n_0 is the noise power spectral density.

3.4. Energy Consumption Model

A UAV mission consists of flight and hovering modes. Following [27], the propulsion power is modeled as

$$P(v) = P_0 \left(1 + \frac{3v^2}{U_t^2} \right) + P_1 \left(\sqrt{1 + \frac{v^2}{4v_0^2}} - \frac{v^2}{2v_0^2} \right) + \frac{1}{2} d_0 \rho_0 s A v^3, \quad (8)$$

where P_0 and P_1 denote the blade profile and induced power under hovering, U_t is the rotor tip speed, v_0 is the mean induced velocity, and d_0 , ρ_0 , s , and A denote the fuselage drag ratio, air density, rotor solidity, and rotor disc area, respectively. Accordingly, the rated flight and hovering power are $\varepsilon_j^f = P(v = v_n)$ and $\varepsilon_j^h = P(v = 0)$.

For UAV u_j , the total mission energy consumption is

$$E_j^{\text{total}} = \varepsilon_j^f \frac{1}{s_j} \sum_{l=0}^{L_j} \delta_{d_{j,l}, d_{j,l+1}} + \varepsilon_j^h \sum_{l=1}^{L_j} \frac{S_{d_{j,l},j}}{R_{d_{j,l},j}}, \quad (9)$$

where $d_{j,0}$ and d_{j,L_j+1} denote the departure and return BS, respectively. The first term represents flight energy and the second term represents hovering energy. To ensure safe mission completion,

$$E_j^{\text{total}} \leq e_j - e_j^{\min}, \quad (10)$$

where e_j denotes the available energy of UAV u_j at the beginning of the current mission round, and $e_j^{\min}$ denotes the minimum energy reserve for safe return.

3.5. UAV Reliability Model

The reliability of UAVs in disaster scenarios is affected by mission duration, energy depletion, and environmental disturbances. Conventional reliability analysis usually assumes a constant failure rate determined by the mean time to failure (MTTF) [19]. To capture energy-dependent failure behavior in long-duration or multi-round missions, energy depletion is incorporated into the classical MTTF-based model [28]. This formulation is motivated by the fact that UAVs with lower residual energy have less margin for safe return, emergency maneuvering, communication retransmission, and hovering operations, and therefore face a higher probability of mission interruption or non-return. The expo-

nential adjustment provides a simple monotonic mapping from residual-energy depletion to an increased effective failure rate, while retaining the tractability of the MTTF-based reliability expression for scheduling optimization.

Let $\{t_1, t_2, \dots, t_n\}$ denote the observed intervals between consecutive failures. The MTTF and the corresponding baseline failure rate are

$$\text{MTTF} = \frac{1}{n} \sum_{i=1}^n t_i, \quad \lambda_0 = \frac{1}{\text{MTTF}}. \quad (11)$$

Under the constant failure-rate assumption, the reliability over mission duration T is

$$R_0(T) = \exp(-\lambda_0 T). \quad (12)$$

Using the total energy consumption E_j^{total} defined in (9), the normalized residual energy of UAV u_j is

$$\zeta_j = \frac{e_j - E_j^{\text{total}}}{e_j - e_j^{\min}}, \quad \zeta_j \in [0, 1], \quad (13)$$

where e_j denotes the available energy at the beginning of the current mission round, and $e_j^{\min}$ denotes the minimum energy required for safe return.

The failure rate is then modeled as

$$\lambda(\zeta_j) = \lambda_0 e^{(1-\zeta_j)}, \quad (14)$$

Accordingly, the energy-adjusted reliability of UAV u_j over duration T is

$$R(\zeta_j, T) = \exp[-\lambda_0 e^{(1-\zeta_j)} T]. \quad (15)$$

The failure rate is treated as an equivalent constant hazard during a single mission round, conditional on the residual-energy state at the beginning of that round. Across different mission rounds, however, the residual energy ζ_j is updated according to mission energy consumption, so the effective failure rate $\lambda(\zeta_j)$ varies from round to round rather than remaining globally constant over the whole scheduling horizon.

3.6. MDS-Coded Fault-Tolerant Backup Model

Under energy depletion and increasing failure risk, a UAV may terminate prematurely and fail to return part of the collected data. Replication-based redundancy can mitigate data loss, but it incurs high storage, communication, and energy overhead in resource-constrained disaster scenarios. To improve data recoverability, an MDS-coded fault-tolerance mechanism, referred to as the MDS-coded Fault-Tolerant Backup Model (MFTB), is introduced at the data layer [29]. MFTB fragments raw data and generates linear redundancy, enabling recovery once the BS receives no fewer than the recovery-threshold number of valid fragments.

Coded fragments are distributed across different UAVs for carrying and return transmission. Since data blocks generated at different times may face different service risks and buffer conditions, their protection levels are adjusted within the same MFTB framework.

Figure 3 illustrates an example with $(n, k) = (8, 5)$, where a data block is divided into five source symbols and encoded into eight fragments. The original data can be recovered once any five valid fragments reach the BS; hence, up to three fragment losses can be tolerated.

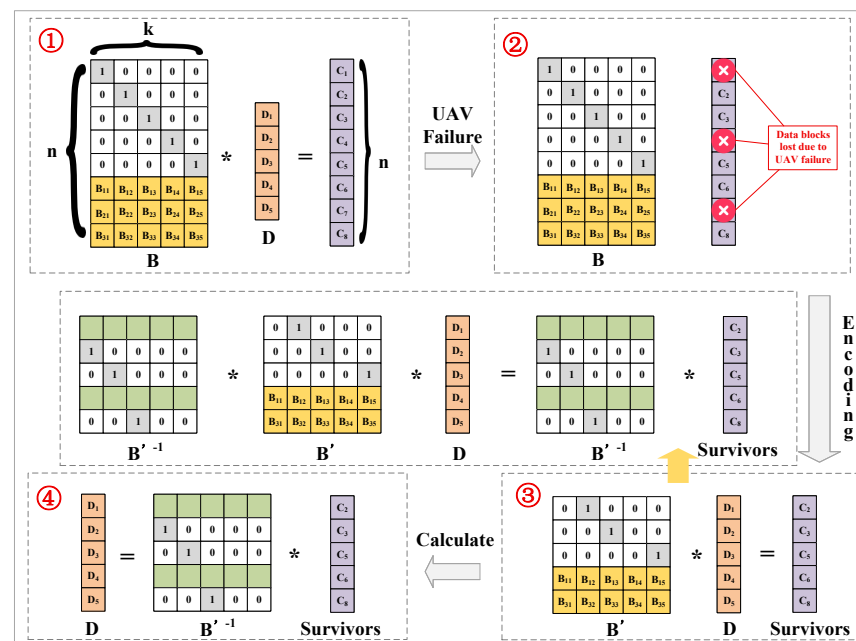


Figure 3. Illustration of MDS-coded data fragmentation and UAV-assisted return transmission (example with $(n, k) = (8, 5)$).

3.6.1. Model Construction and Linear Encoding

The data generated at node i accumulate over time. Let τ_i denote the most recent sealing time of node i . For a time-interval data block formed at time t , the raw data accumulated over $(\tau_i, t]$ are sealed into $\mathcal{W}_{i,p}$, where p indexes the p -th block generated by node i and $S_{i,p}$ denotes its size.

Each block $\mathcal{W}_{i,p}$ is encoded using an $(n_{i,p}, k_{i,p})$ code, where $n_{i,p}$ is the number of generated fragments and $k_{i,p}$ is the recovery threshold. Represent $\mathcal{W}_{i,p}$ by a source-symbol vector over the finite field $\mathbb{F}$:

$$w_{i,p} = (w_{i,p}(1), w_{i,p}(2), \dots, w_{i,p}(k_{i,p}))^T. \quad (16)$$

Let $B_{i,p} \in \mathbb{F}^{n_{i,p} \times k_{i,p}}$ denote the encoding matrix. The coded-fragment vector is

$$b_{i,p} = B_{i,p} w_{i,p} = [b_{i,p,1}, b_{i,p,2}, \dots, b_{i,p,n_{i,p}}]^T. \quad (17)$$

The MDS coding used in this work is designed for erasure resilience and data recoverability rather than confidentiality protection. The encoding matrix $B_{i,p}$ is assumed to be public, and the generated coded fragments are not threshold secret shares. Therefore, if the collected data contain sensitive information, additional encryption or secret-sharing mechanisms would be required before fragment generation or transmission. Such security mechanisms, as well as Byzantine robustness and adversarial channel protection, are outside the scope of the present recoverability-aware scheduling formulation.

If UAV failures occur during mission execution or return transmission, the BS may receive only $h_{i,p}$ fragments, where $h_{i,p} \leq n_{i,p}$. Let $B'_{i,p} \in \mathbb{F}^{h_{i,p} \times k_{i,p}}$ denote the submatrix corresponding to the received fragments. Then

$$b'_{i,p} = B'_{i,p} w_{i,p}. \quad (18)$$

3.6.2. Data Recoverability Analysis

Recoverability depends on whether the received encoding submatrix $B'_{i,p}$ has full column rank.

Theorem 1 (Necessary and sufficient condition for data-block recoverability). *For any encoding matrix $B_{i,p}$, unique recovery of $\mathcal{W}_{i,p}$ at the BS is possible if and only if the received encoding submatrix $B'_{i,p}$ has full column rank, i.e., $\text{rank}(B'_{i,p}) = k_{i,p}$.*

Proof. If $\text{rank}(B'_{i,p}) = k_{i,p}$, then $B'_{i,p}$ has full column rank, and the linear system $b'_{i,p} = B'_{i,p}w_{i,p}$ admits a unique solution. Conversely, if $\text{rank}(B'_{i,p}) < k_{i,p}$, there exists a nonzero vector z such that $B'_{i,p}z = 0$. Thus, $w_{i,p}$ and $w_{i,p} + z$ yield the same observation, so the original data cannot be uniquely recovered. $\square$

To guarantee recovery from any sufficient fragment subset, the encoding matrix $B_{i,p}$ is required to satisfy the MDS property. A Vandermonde-structured matrix associated with RS coding satisfies this property.

Theorem 2 (Recoverability of MDS coding). *If the encoding matrix $B_{i,p}$ satisfies the MDS property, then the complete data block $\mathcal{W}_{i,p}$ can be recovered from any $k_{i,p}$ received fragments. Accordingly, the system can tolerate up to $n_{i,p} - k_{i,p}$ fragment losses.*

Proof. Consider any $k_{i,p}$ rows of $B_{i,p}$, and denote the corresponding submatrix by $B[S, :]$. For the Vandermonde structure

$$B_{i,p} = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{k_{i,p}-1} \\ 1 & \alpha_2 & \alpha_2^2 & \cdots & \alpha_2^{k_{i,p}-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_{n_{i,p}} & \alpha_{n_{i,p}}^2 & \cdots & \alpha_{n_{i,p}}^{k_{i,p}-1} \end{bmatrix},$$

where α_j are distinct elements in $\mathbb{F}$, any $k_{i,p} \times k_{i,p}$ submatrix satisfies

$$\det(B[S, :]) = \prod_{u < v} (\alpha_{s_v} - \alpha_{s_u}) \neq 0.$$

Therefore, any $k_{i,p}$ received fragments form an invertible submatrix, and $\mathcal{W}_{i,p}$ is recoverable whenever at least $k_{i,p}$ valid fragments survive. $\square$

Theorems 1 and 2 show that MDS-coded fragmentation enables recovery from any set of at least $k_{i,p}$ valid fragments. The remaining issue is how to select an appropriate redundancy level for each data block under finite buffer resources.

3.6.3. Adaptive Protection-Level Decision Based on Marginal Net Preservation Gain

A uniform redundancy configuration cannot adapt to the heterogeneous service risks and buffer conditions of different data blocks. Stronger redundancy improves recovery probability, but also increases fragment occupancy. To balance this tradeoff, the protection level is selected according to the marginal net preservation gain.

For a time-interval data block $\mathcal{W}_{i,p}$ generated by node i at time t , define the discrete protection-level set as

$$\mathcal{G} = \{g_0, g_1, \dots, g_L\}, \quad (19)$$

where the protection strength increases with the level index. For each protection level g_ℓ , let the corresponding coding-parameter pair be $(n^{(\ell)}, k^{(\ell)})$, satisfying

$$\frac{n^{(0)}}{k^{(0)}} \leq \frac{n^{(1)}}{k^{(1)}} \leq \cdots \leq \frac{n^{(L)}}{k^{(L)}}. \quad (20)$$

If the protection level of $\mathcal{W}_{i,p}$ is upgraded from g_ℓ to $g_{\ell+1}$, the marginal recovery gain is

$$\Delta G_{i,p}^{(\ell)}(t) = S_{i,p} \left[P_{i,p}^{\text{rec}}(n^{(\ell+1)}, k^{(\ell+1)}, \bar{q}(t)) - P_{i,p}^{\text{rec}}(n^{(\ell)}, k^{(\ell)}, \bar{q}(t)) \right], \quad (21)$$

where $\bar{q}(t)$ is the average successful return probability of the active UAVs. Under the independent and identically distributed approximation,

$$P_{i,p}^{\text{rec}}(n, k, \bar{q}) = \sum_{h=k}^n \binom{n}{h} \bar{q}^h (1 - \bar{q})^{n-h}. \quad (22)$$

Under correlated fragment losses, this independence-based approximation may overestimate the benefit of fragment distribution because multiple fragments can be lost simultaneously. Modeling such spatial or temporal dependence is beyond the scope of the present formulation.

Let $C_i^{\text{rem}}(t)$ denote the remaining buffer space, $T_i^c(t)$ denote the time interval to the next state refresh, and λ_i denote the data arrival rate. The marginal buffer cost of upgrading from g_ℓ to $g_{\ell+1}$ is

$$\Delta C_{i,p}^{(\ell)}(t) = \lambda_i T_i^c(t) \cdot \min \left\{ \frac{\left(\frac{n^{(\ell+1)}}{k^{(\ell+1)}} - \frac{n^{(\ell)}}{k^{(\ell)}} \right) S_{i,p}}{C_i^{\text{rem}}(t)}, 1 \right\}. \quad (23)$$

Protection levels are upgraded sequentially. For $\ell = 0, 1, \dots, L-1$, the upgrade from g_ℓ to $g_{\ell+1}$ is accepted if

$$\Delta G_{i,p}^{(\ell)}(t) > \Delta C_{i,p}^{(\ell)}(t). \quad (24)$$

Otherwise, the process terminates, and the final protection level is the highest accepted level.

The contribution of MFTB does not lie in MDS coding itself, but in its recoverability-aware use under finite IoT buffer constraints. Candidate coding configurations are evaluated through the marginal net preservation gain, which balances the increase in recovery probability against the additional fragment-storage cost under the current UAV return reliability. MFTB therefore adapts the protection level to the system state instead of using a fixed redundancy configuration.

3.7. Problem Formulation

Based on the preceding models, we formulate fault-tolerant scheduling as maximizing the expected recoverable information over multiple mission rounds under UAV failures.

3.7.1. Objective Function

Let $\{\mathcal{W}_{i,1}, \mathcal{W}_{i,2}, \dots, \mathcal{W}_{i,P_i}\}$ denote the data blocks generated by node i over the scheduling period, and let $S_{i,p}$ denote the information size of $\mathcal{W}_{i,p}$. Each block $\mathcal{W}_{i,p}$ is MDS-coded with parameter pair $(n_{i,p}, k_{i,p})$, where $n_{i,p}$ and $k_{i,p}$ denote the number of generated fragments and the recovery threshold, respectively. Thus, $\mathcal{W}_{i,p}$ is recoverable once the BS receives any $k_{i,p}$ valid fragments, and the coding parameters satisfy

$$1 \leq k_{i,p} \leq n_{i,p}, \quad k_{i,p}, n_{i,p} \in \mathbb{Z}_+. \quad (25)$$

The scheduling period is divided into R mission rounds, denoted by $\mathcal{R} = \{1, 2, \dots, R\}$. Let $X_{j,r} \in \{0, 1\}$ denote whether UAV u_j successfully returns to the BS after the r -th mission round. Its success probability is determined by the survival reliability in that round, i.e.,

$$\Pr(X_{j,r} = 1) = R_j^{(r)}. \quad (26)$$

In the current formulation, the return events $X_{j,r}$ are assumed to be statistically independent across UAVs and mission rounds, conditional on the corresponding round-level reliability values. This assumption represents independent random crash/non-return faults caused by energy depletion or device abnormality. Common-cause, spatially correlated, or fleet-wide disaster-induced failures are not modeled in the present formulation and are treated as a limitation of the current framework.

Let $v_i^{j,r} \in \{0, 1\}$ indicate whether UAV u_j visits node i in round r , and let $a_{i,p,\ell}^{j,r} \in \{0, 1\}$ indicate whether the ℓ -th fragment of data block $\mathcal{W}_{i,p}$ ($\ell = 1, \dots, n_{i,p}$) is carried by UAV u_j in round r . Then, the total number of valid fragments of data block $\mathcal{W}_{i,p}$ eventually received at the BS over the scheduling period is

$$H_{i,p} = \sum_{j=1}^M \sum_{r=1}^R \sum_{\ell=1}^{n_{i,p}} a_{i,p,\ell}^{j,r} X_{j,r}. \quad (27)$$

According to the recovery condition of MDS coding, data block $\mathcal{W}_{i,p}$ is fully recoverable if and only if $H_{i,p} \geq k_{i,p}$. The expected recoverable information is therefore defined as

$$y_{i,p} = S_{i,p} \cdot \Pr(H_{i,p} \geq k_{i,p}), \quad (28)$$

and the total expected recoverable information over the scheduling period is

$$F_1 = \sum_{i=1}^N \sum_{p=1}^{P_i} y_{i,p}. \quad (29)$$

3.7.2. Optimization Problem

Accordingly, the optimization problem is formulated as

$$\mathcal{P}: \quad \max_{\{v,a\}} \quad F_1 = \sum_{i=1}^N \sum_{p=1}^{P_i} S_{i,p} \cdot \Pr(H_{i,p} \geq k_{i,p}) \quad (30)$$

$$\text{s.t.} \quad a_{i,p,\ell}^{j,r} \leq v_i^{j,r}, \quad \forall i, p, \ell, j, r, \quad (31)$$

$$\sum_{j=1}^M \sum_{r=1}^R a_{i,p,\ell}^{j,r} \leq 1, \quad \forall i, p, \ell, \quad (32)$$

$$v_i^{j,r} \in \{0, 1\}, \quad a_{i,p,\ell}^{j,r} \in \{0, 1\}. \quad (33)$$

Constraint (31) couples fragment-carrying decisions with UAV access decisions, Constraint (32) prevents repeated carrying of the same fragment, and Constraint (33) specifies the binary domains of the decision variables.

4. Optimization Method

The coupling among UAV failures, access timing, visiting order, and fragment allocation makes problem $\mathcal{P}$ challenging. To address this issue, a loss-pressure-guided dual-layer collaborative fault-tolerant scheduling algorithm, termed LP-DCFS, is developed under an event-triggered replanning framework.

At each replanning instant t_0 , LP-DCFS constructs a local rescheduling problem for the UAV u_j that triggers replanning. Given the current fragment, buffer, and energy states, LP-DCFS first screens candidate terminals according to loss pressure and flight feasibility. The upper layer searches for an access sequence z_j , and the lower layer evaluates each decoded access plan by optimizing coded-fragment allocation under the corresponding resource budgets. The access sequence with the largest evaluated recovery gain is selected for execution, as illustrated in Figure 4.

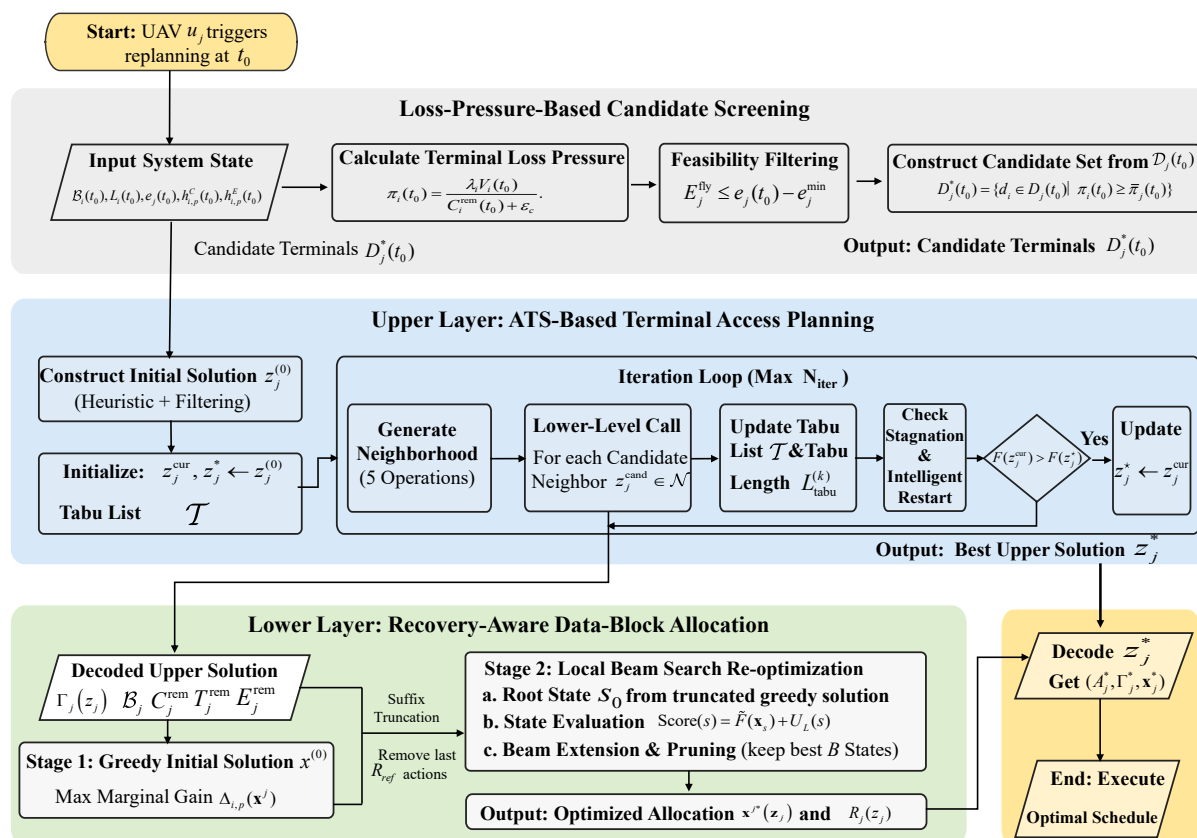


Figure 4. Flowchart of the LP-DCFS algorithm.

4.1. Loss-Pressure-Based Candidate Screening

At replanning instant t_0 , a candidate terminal set is constructed for UAV u_j and used as the search space of the upper-layer access planner.

4.1.1. Terminal Loss Pressure

To prioritize terminals with large unrecovered information and high buffer-overflow risk, a loss-pressure metric is defined. Let $\mathcal{B}_i(t_0)$ denote the set of sealed but unrecovered data blocks at terminal d_i . For any $\mathcal{W}_{i,p} \in \mathcal{B}_i(t_0)$, let $k_{i,p}$ be its recovery threshold, and let $h_{i,p}^C(t_0)$ and $h_{i,p}^E(t_0)$ be the numbers of fragments already confirmed at the BS and currently in transit, respectively. The remaining recovery demand is

$$\Delta_{i,p}(t_0) \triangleq [k_{i,p} - (h_{i,p}^C(t_0) + h_{i,p}^E(t_0))]_{+}. \quad (34)$$

The unrecovered information at terminal d_i is

$$V_i(t_0) = \sum_{\mathcal{W}_{i,p} \in \mathcal{B}_i(t_0)} S_{i,p} \mathbb{I}\{\Delta_{i,p}(t_0) > 0\}, \quad (35)$$

where $S_{i,p}$ is the block size and $\mathbb{I}\{\cdot\}$ is the indicator function. Let $L_i(t_0) = L_i^{\text{raw}}(t_0) + L_i^{\text{frag}}(t_0)$ be the current buffer occupancy and $C_i^{\text{rem}}(t_0) = L_i^{\text{max}} - L_i(t_0)$ be the remaining buffer space. To avoid singularity when the buffer is saturated, define the effective time to buffer saturation as $T_i^{\text{full}}(t_0) = (C_i^{\text{rem}}(t_0) + \varepsilon_c) / \lambda_i$, where λ_i is the data generation rate and $\varepsilon_c > 0$ is a small constant. The loss pressure of terminal d_i is

$$\pi_i(t_0) \triangleq \frac{V_i(t_0)}{T_i^{\text{full}}(t_0)} = \frac{\lambda_i V_i(t_0)}{C_i^{\text{rem}}(t_0) + \varepsilon_c}. \quad (36)$$

4.1.2. Feasible Candidate Construction

A terminal is feasible for UAV u_j if the round-trip flight energy between the BS and d_i does not exceed the available energy. The serviceable terminal set is

$$\begin{aligned} \mathcal{D}_j(t_0) \triangleq \{d_i \in \mathcal{D} \mid E_j^{\text{fly}}(\text{BS} \rightarrow d_i \rightarrow \text{BS}) \\ \leq e_j(t_0) - e_j^{\text{min}}\}, \end{aligned} \quad (37)$$

where $e_j(t_0)$ is the available energy of u_j at time t_0 , and e_j^{min} is the safe-return threshold.

If $\mathcal{D}_j(t_0) = \emptyset$, no terminal is assigned to UAV u_j at this replanning instant, and the candidate terminal set is empty. Otherwise, the average loss pressure on $\mathcal{D}_j(t_0)$ is

$$\bar{\pi}_j(t_0) = \frac{1}{|\mathcal{D}_j(t_0)|} \sum_{d_i \in \mathcal{D}_j(t_0)} \pi_i(t_0). \quad (38)$$

The candidate terminal set is then

$$\mathcal{D}_j^*(t_0) \triangleq \{d_i \in \mathcal{D}_j(t_0) \mid \pi_i(t_0) \geq \bar{\pi}_j(t_0)\}. \quad (39)$$

To justify this mean-based pruning rule, for any nonempty set $\mathcal{S} \subseteq \mathcal{D}_j(t_0)$, define $\mu(\mathcal{S}; t_0) = \frac{1}{|\mathcal{S}|} \sum_{d_i \in \mathcal{S}} \pi_i(t_0)$ as its average loss pressure.

Theorem 3 (Mean-pruning property). *Given t_0 , for any nonempty set $\mathcal{S} \subseteq \mathcal{D}_j(t_0)$, if there exists $d_a \in \mathcal{S}$ such that*

$$\pi_a(t_0) < \mu(\mathcal{S}; t_0), \quad (40)$$

then

$$\mu(\mathcal{S} \setminus \{d_a\}; t_0) > \mu(\mathcal{S}; t_0). \quad (41)$$

Proof. Let $m = |\mathcal{S}|$ and $\sum_{d_i \in \mathcal{S}} \pi_i(t_0) = \Sigma$. Then $\mu(\mathcal{S}; t_0) = \Sigma / m$ and

$$\mu(\mathcal{S} \setminus \{d_a\}; t_0) = \frac{\Sigma - \pi_a(t_0)}{m - 1}.$$

By (40), $\Sigma - m\pi_a(t_0) > 0$. Hence,

$$\mu(\mathcal{S} \setminus \{d_a\}; t_0) - \mu(\mathcal{S}; t_0) = \frac{\Sigma - m\pi_a(t_0)}{m(m-1)} > 0,$$

which proves (41). $\square$

4.2. ATS-Based Terminal Access Planning

Given $\mathcal{D}_j^*(t_0)$, the upper layer determines the service set and visiting order of the replanning UAV u_j . This joint subset-selection and route-ordering problem is solved by an adaptive tabu search (ATS) method [30,31]. Each candidate access sequence is decoded

and evaluated using the recoverable-information gain returned by the lower-layer coded-fragment allocation. The procedure is summarized in Algorithm 1.

4.2.1. Solution Encoding and Decoding

Let $\mathcal{D}_j^*(t_0) = \{\tilde{d}_1, \tilde{d}_2, \dots, \tilde{d}_{M_j}\}$, where $M_j = |\mathcal{D}_j^*(t_0)|$. An upper-layer solution is represented by

$$z_j = [\hat{d}_1, \hat{d}_2, \dots, \hat{d}_{L_j}], \quad (42)$$

where $\hat{d}_\ell \in \mathcal{D}_j^*(t_0)$, $L_j \leq M_j$, and all elements are distinct.

The initial solution is generated by sorting candidate terminals in descending order of loss pressure and inserting them from the BS while preserving flight and safe-return feasibility. For a given z_j , terminals are checked in order and retained only if feasibility is preserved, yielding the realized service set $A_j(z_j)$ and visiting sequence $\Gamma_j(z_j)$. The decoded access plan determines the serviceable block set and residual resource budgets for lower-layer allocation. Let $R_j(z_j)$ denote the allocation value returned by the lower layer. The upper-layer fitness is

$$F(z_j) \triangleq R_j(z_j) - \Phi_j(z_j), \quad (43)$$

where $\Phi_j(z_j)$ is the penalty for infeasible or high-cost solutions.

4.2.2. Neighborhood Design

Five neighborhood operators are used in ATS [32], as summarized in Table 1.

Table 1. Neighborhood operators used in ATS.

Type	Symbol	Move	Role
Insertion	$\mathcal{N}^{\text{ins}}$	$\hat{d} \notin z_j \rightarrow z_j$	Expand the service set
Deletion	$\mathcal{N}^{\text{del}}$	$\hat{d} \in z_j \rightarrow \emptyset$	Reduce the route size
Swap	$\mathcal{N}^{\text{swap}}$	$(\hat{d}_p, \hat{d}_q) \mapsto (\hat{d}_q, \hat{d}_p)$	Adjust the visiting order
Replacement	$\mathcal{N}^{\text{rep}}$	$\hat{d} \in z_j \leftarrow \hat{d}' \notin z_j$	Reconfigure the service set
Reordering	$\mathcal{N}^{\text{reo}}$	$z_j[a:b] \rightarrow \pi(z_j[a:b])$	Optimize a local subroute

In Table 1, $\hat{d}$ and $\hat{d}'$ denote candidate terminals, $z_j[a:b]$ denotes a contiguous subsequence, and $\pi(\cdot)$ denotes a reordering operator.

4.2.3. Adaptive Tabu and Restart

ATS maintains a tabu list of recently executed move attributes to avoid cycling. Let $L_{\text{tabu}}^{(k)}$ denote the tabu tenure at iteration k . It is updated as

$$L_{\text{tabu}}^{(k+1)} = \begin{cases} \min\{L_{\text{max}}, L_{\text{tabu}}^{(k)} + \Delta^+\}, & \text{Stagnation,} \\ \max\{L_{\text{min}}, L_{\text{tabu}}^{(k)} - \Delta^-\}, & \text{Improvement,} \end{cases} \quad (44)$$

where L_{min} and L_{max} are the lower and upper bounds, and Δ^+ and Δ^- are adjustment steps. A tabu move is accepted if it improves the historical best solution.

If no improvement is observed for N_{stall} consecutive iterations, an intelligent restart reconstructs a feasible solution from high-value terminals in the historical best solution. The search terminates when N_{iter} is reached or repeated restarts fail to improve the historical best solution.

Algorithm 1: ATS-based terminal access optimization

Input: Candidate terminal set $\mathcal{D}_j^*(t_0)$, N_{iter} , $L_{\text{tabu}}^{(0)}$
Output: Optimal service set A_j^* and visiting sequence Γ_j^*

- 1 Construct an initial feasible solution $z_j^{(0)}$;
- 2 Initialize $z_j^{\text{cur}} \leftarrow z_j^{(0)}$, $z_j^* \leftarrow z_j^{(0)}$, and tabu list $\mathcal{T}$;
- 3 **for** $k = 0$ **to** $N_{\text{iter}} - 1$ **do**
- 4 Generate the neighborhood of z_j^{cur} according to Table 1;
- 5 **foreach** *admissible neighbor* z_j' **do**
- 6 Decode z_j' to obtain $A_j(z_j')$ and $\Gamma_j(z_j')$;
- 7 Invoke lower-layer allocation to obtain $R_j(z_j')$;
- 8 Compute $F(z_j')$ according to (43);
- 9 **end**
- 10 Select the neighbor with the largest $F(z_j')$ as z_j^{cur} ;
- 11 Update $\mathcal{T}$ and $L_{\text{tabu}}^{(k)}$ according to (44);
- 12 Update z_j^* if improved;
- 13 **if** *no improvement is observed for* N_{stall} *consecutive iterations* **then**
- 14 Restart from a reconstructed feasible solution;
- 15 **end**
- 16 **end**
- 17 Decode z_j^* and return A_j^*, Γ_j^* ;

4.3. Recovery-Aware Data-Block Allocation

For a candidate access sequence z_j , the decoded access plan determines the serviceable block set $\mathcal{B}_j$ and remaining resource budgets ($C_j^{\text{rem}}, T_j^{\text{rem}}, E_j^{\text{rem}}$). The lower layer allocates coded fragments over $\mathcal{B}_j$ and returns $R_j(z_j)$ for upper-layer evaluation.

4.3.1. Decision Variables and Constraints

For each $\mathcal{W}_{i,p} \in \mathcal{B}_j$, define the integer variable $x_{i,p}^j \in \mathbb{Z}_+$ as the number of coded fragments carried by UAV u_j . Let $S_{i,p}$ denote the information size of $\mathcal{W}_{i,p}$, and let $(n_{i,p}, k_{i,p})$ denote its coding parameters. The coded-fragment size is $\omega_{i,p} = S_{i,p}/k_{i,p}$. If $\bar{x}_{i,p}^j$ is the maximum number of remaining fragments still assignable to UAV u_j , then

$$0 \leq x_{i,p}^j \leq \bar{x}_{i,p}^j, \quad \forall (i,p) \in \mathcal{B}_j. \quad (45)$$

Let $R_{i,j}$ denote the transmission rate between terminal d_i and UAV u_j , and let $\epsilon_{i,p,j}$ denote the energy required to download one coded fragment of $\mathcal{W}_{i,p}$. The storage, communication-time, and energy constraints are

$$\sum_{(i,p) \in \mathcal{B}_j} \omega_{i,p} x_{i,p}^j \leq C_j^{\text{rem}}, \quad (46)$$

$$\sum_{(i,p) \in \mathcal{B}_j} \frac{\omega_{i,p}}{R_{i,j}} x_{i,p}^j \leq T_j^{\text{rem}}, \quad (47)$$

$$\sum_{(i,p) \in \mathcal{B}_j} \epsilon_{i,p,j} x_{i,p}^j \leq E_j^{\text{rem}}. \quad (48)$$

The feasible region is denoted by $\mathcal{X}_j$.

4.3.2. Recovery-Probability Approximation

The original system objective is

$$F_1 = \sum_{i,p} S_{i,p} \Pr(H_{i,p} \geq k_{i,p}), \quad (49)$$

where $H_{i,p}$ denotes the total number of valid coded fragments of $\mathcal{W}_{i,p}$ that eventually reach the BS.

To reduce repeated lower-layer computation during upper-layer search, a moment-based approximation is adopted. Let $h_{i,p}^C(t_0)$ denote the number of valid fragments already confirmed at the BS. For any UAV u_m carrying fragments of $\mathcal{W}_{i,p}$, let q_m denote its successful return probability and $x_{i,p}^m$ the number of carried fragments. For the current allocation vector $\mathbf{x}^j$, the mean and variance of $H_{i,p}$ are approximated by

$$\mu_{i,p}(\mathbf{x}^j) = h_{i,p}^C(t_0) + \sum_{m \neq j} q_m x_{i,p}^m + q_j(\Gamma_j) x_{i,p}^j, \quad (50)$$

$$\begin{aligned} \sigma_{i,p}^2(\mathbf{x}^j) = & \sum_{m \neq j} q_m(1 - q_m)(x_{i,p}^m)^2 \\ & + q_j(\Gamma_j)(1 - q_j(\Gamma_j))(x_{i,p}^j)^2, \end{aligned} \quad (51)$$

where $q_j(\Gamma_j)$ is the successful return probability of UAV u_j under visiting sequence Γ_j .

Using a Gaussian tail approximation with continuity correction, the recovery probability of $\mathcal{W}_{i,p}$ is approximated as

$$\psi_{i,p}(\mathbf{x}^j) = \Phi\left(\frac{\mu_{i,p}(\mathbf{x}^j) - k_{i,p} + 0.5}{\sigma_{i,p}(\mathbf{x}^j) + \varepsilon}\right), \quad (52)$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function and $\varepsilon > 0$ is a small constant. The approximate objective is

$$\tilde{F}(\mathbf{x}^j) = \sum_{(i,p) \in \mathcal{B}_j} S_{i,p} \psi_{i,p}(\mathbf{x}^j). \quad (53)$$

Thus, the lower-layer problem associated with z_j is

$$\max_{\mathbf{x}^j \in \mathcal{X}_j} \tilde{F}(\mathbf{x}^j). \quad (54)$$

4.3.3. Greedy Initialization

A feasible initial solution is constructed by a marginal-gain greedy strategy. For the current state $\mathbf{x}^j$, the marginal gain of adding one fragment to $\mathcal{W}_{i,p}$ is

$$\Delta_{i,p}(\mathbf{x}^j) = \tilde{F}(\mathbf{x}^j + \mathbf{e}_{i,p}) - \tilde{F}(\mathbf{x}^j), \quad (55)$$

where $\mathbf{e}_{i,p}$ is the unit vector associated with $x_{i,p}^j$.

The feasible action set is

$$\mathcal{A}(\mathbf{x}^j) = \left\{ (i,p) \in \mathcal{B}_j \mid x_{i,p}^j < \bar{x}_{i,p}^j, \mathbf{x}^j + \mathbf{e}_{i,p} \in \mathcal{X}_j \right\}. \quad (56)$$

At each step, the feasible action with the largest positive marginal gain is selected until no such action exists, yielding $\mathbf{x}^{(0)}$.

4.3.4. Local Beam-Search Refinement

Since the greedy strategy uses one-step gains, it may miss the cumulative benefit of assigning multiple fragments to the same block. To refine $\mathbf{x}^{(0)}$, a local beam search is applied in its induced neighborhood [33].

Let M be the length of the action sequence associated with $\mathbf{x}^{(0)}$. Removing the last R_{ref} actions gives a partial solution $\bar{\mathbf{x}}^{(0)}$, which defines the root state. Each beam state s consists of an allocation vector and the remaining storage, time, and energy budgets.

To avoid ranking states only by the current objective, a finite-horizon rollout estimate is used. The score of state s is

$$\text{Score}(s) = \tilde{F}(\mathbf{x}_s) + U_L(s), \quad (57)$$

where $U_L(s)$ is the cumulative gain of a length- L greedy rollout:

$$U_L(s) = \sum_{\ell=1}^L \max_{(i,p) \in \mathcal{A}_\ell(s)} \Delta_{i,p}^{(\ell)}. \quad (58)$$

In (58), $\mathcal{A}_\ell(s)$ denotes the feasible action set at rollout step ℓ , and $\Delta_{i,p}^{(\ell)}$ denotes the corresponding marginal gain [34].

After each expansion layer, candidate states are sorted by $\text{Score}(s)$, duplicate states are removed, and the top B states are retained for the next layer [35].

The procedure is summarized in Algorithm 2.

Algorithm 2: Recovery-probability-based data-block allocation

Input: Decoded access plan of z_j , candidate block set $\mathcal{B}_j$, remaining resources $(C_j^{\text{rem}}, T_j^{\text{rem}}, E_j^{\text{rem}})$, parameters (B, R_{ref}, L)

Output: Allocation solution $\mathbf{x}^{j*}$ and allocation value $R_j(z_j)$

- 1 Construct a greedy feasible solution $\mathbf{x}^{(0)}$;
 - 2 Remove the last R_{ref} greedy actions to obtain the root state s_0 ;
 - 3 Initialize the beam set $\mathcal{S} \leftarrow \{s_0\}$;
 - 4 **for** $r = 1, \dots, R_{\text{ref}}$ **do**
 - 5 Expand each state in $\mathcal{S}$ by feasible actions;
 - 6 Evaluate child states by (57)–(58);
 - 7 Remove duplicate states and retain the top- B states as the new beam set $\mathcal{S}$;
 - 8 **end**
 - 9 Output the allocation of the highest-scoring state in $\mathcal{S}$ as $\mathbf{x}^{j*}$;
 - 10 Set $R_j(z_j) \leftarrow \tilde{F}(\mathbf{x}^{j*})$;
 - 11 **return** $\mathbf{x}^{j*}, R_j(z_j)$;
-

Although the bilevel structure makes LP-DCFS more computationally demanding than a single-layer greedy method, its computational burden is bounded by the prescribed search parameters. Let N_{iter} denote the maximum number of ATS iterations, N_{nei} the maximum number of admissible neighbors evaluated in each iteration, C_{dec} the cost of decoding and checking one candidate, and C_{low} the cost of one lower-layer allocation call. Since each admissible neighbor is decoded and evaluated by the lower layer, the dominant complexity of the upper-layer ATS is $O(N_{\text{iter}}N_{\text{nei}}(C_{\text{dec}} + C_{\text{low}}))$. The tabu-list update, neighbor selection, and restart operations have lower-order costs compared with repeated candidate evaluations.

For the lower layer, let A_{max} denote the maximum number of feasible allocation actions considered at one search state, and let C_Δ denote the cost of evaluating one marginal gain. The greedy initialization has a dominant complexity of $O(A_{\text{max}}|\mathcal{B}_j|C_\Delta)$, while the local

beam-search refinement has a dominant complexity of $O(R_{\text{ref}}BA_{\text{max}}LC_{\Delta})$. Therefore, B , R_{ref} , and L control the beam width, local refinement range, and look-ahead depth, respectively, whereas N_{iter} and N_{nei} determine the computational burden of the upper-layer search. Under the adopted settings, the search parameters provide a controllable balance between solution refinement and computational cost.

5. Performance Evaluation

This section evaluates the proposed fault-tolerance mechanism and scheduling algorithm through simulation. MFTB is first assessed across different terminal scales with varying numbers of failed UAVs in terms of data recoverability and recovery efficiency, using recovered data volume, recoverable data volume, lost data volume, and average recovery time as performance metrics. Under a unified fault-tolerance mechanism, LP-DCFS is then compared with several baseline scheduling algorithms in terms of recovery gain, resource consumption, and scheduling efficiency.

5.1. Simulation Setup

A disaster-oriented data collection scenario is constructed, consisting of one BS, multiple heterogeneous UAVs, and IoTs distributed over the mission area. The BS is located at the origin $[0,0]$ and is responsible for scheduling and centralized data aggregation, while UAVs depart from the BS to perform data collection and fragment return tasks. The simulation focuses on the scheduling and data-recovery effects of UAV random crash/non-return faults under finite energy, storage, and communication resources. IoT locations are assumed to be fixed and known during the scheduling horizon, the BS performs centralized scheduling, and UAV-IoT communication follows the LoS channel model described in Section 3. The simulated failure process represents independent random UAV failures caused by energy depletion or device abnormality, whereas correlated disaster-induced failures, adversarial attacks, and dynamic obstacle effects are not explicitly modeled in the current simulation setting. All simulations are implemented in Python 3.10.13 and conducted on a platform with a 12th Gen Intel(R) Core(TM) i9-12900H CPU and 32 GB DDR5 4800 MHz memory.

Three types of heterogeneous UAVs are considered, with different flight speeds, power consumption, battery capacities, bandwidths, storage capacities, and MTTF values [19]. The corresponding parameters are listed in Table 2. The three UAV types are used to represent heterogeneous platforms rather than a monotonic performance ranking. Their different speeds, energy capacities, flight powers, hovering powers, storage capacities, and MTTF values reflect trade-offs among UAV size, battery capacity, payload, propulsion demand, and onboard communication/storage resources. Therefore, a UAV with a larger battery or stronger onboard resources may have a lower cruising speed and higher power consumption because of its heavier platform or payload.

Table 2. UAV Parameters Used in Simulation.

Parameter	UAV-I	UAV-II	UAV-III
Flight speed (m/s)	20	18	16
Energy capacity (Wh)	75	90	135
Flight power (W)	90	90	125
Hover power (W)	100	108	135
Storage capacity (GB)	4	6	5
Bandwidth (MHz)	10	12	8
MTTF (s)	8000	9000	10,000

Three terminal scales, including 20, 30, and 40 IoTs, are considered with a fixed number of UAVs, as shown in Figure 5. Figures 5a–c show the inter-terminal distance matrices, which exhibit no evident clustering, indicating random terminal deployment. Figures 5d–f report the distributions of terminal–BS distance, storage capacity, and data arrival rate, mainly falling within 1.0–3.0 km, 12–22 GB, and 4–12 MB/s, respectively. These results confirm the spatial randomness and parameter heterogeneity of the simulated scenarios.

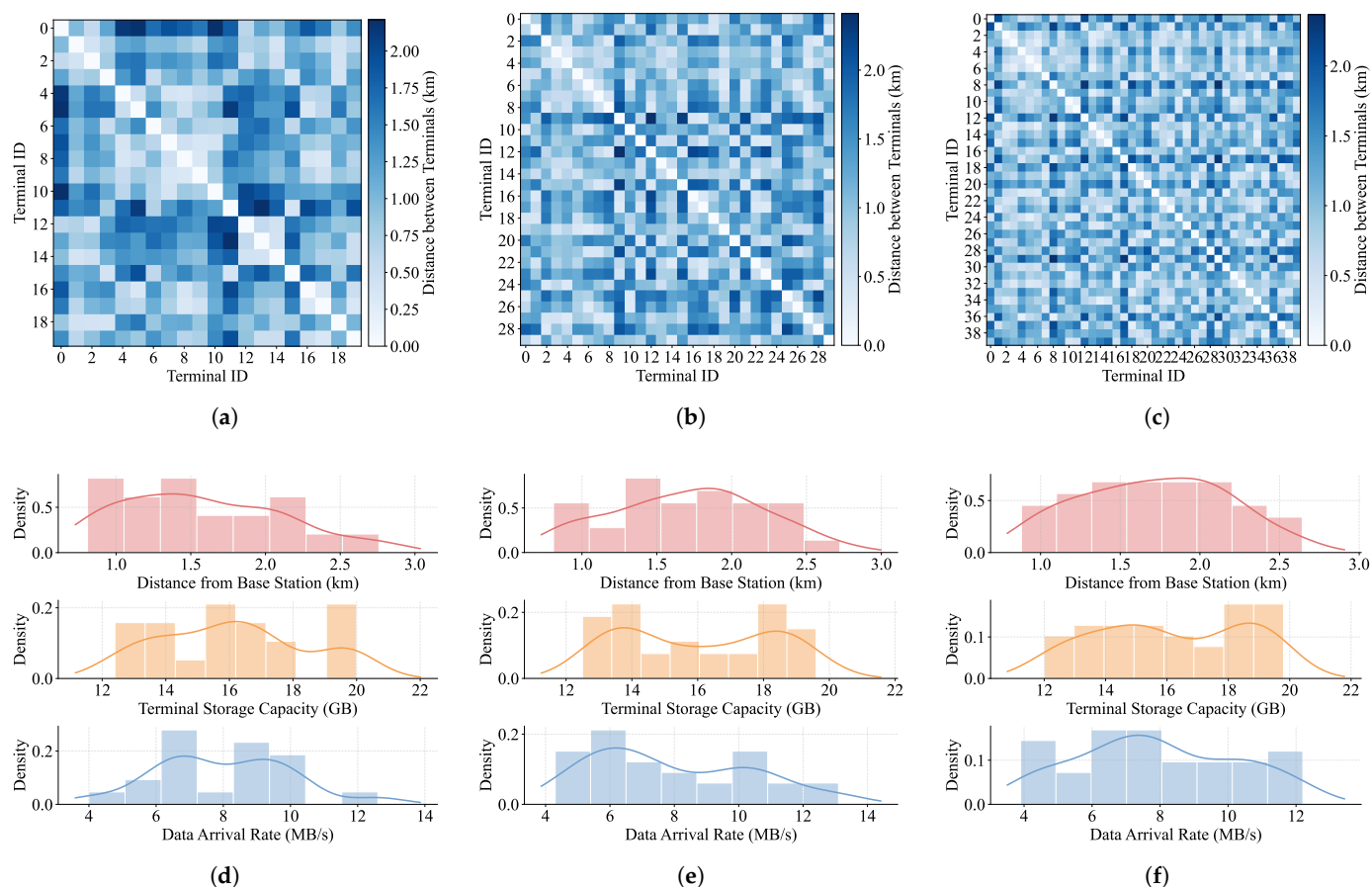


Figure 5. Simulation Scenarios and Statistical Characteristics under Different Terminal Scales. (a–c) Heatmaps of the inter-terminal distance matrices for scenarios with 20, 30, and 40 terminals, respectively. (d–f) Statistical distributions of terminal–BS distance, terminal storage capacity, and data arrival rate for the corresponding scenarios.

To assess the consistency of the Gaussian-tail approximation used in MFTB, we conducted an additional numerical comparison based on the recorded MFTB simulation data. The evaluation covers the simulation settings with 20, 30, and 40 IoTs and with 0, 2, and 4 failed UAVs. The recorded MFTB simulations used the coding configurations $(n, k) = (1, 1)$, $(6, 4)$, and $(8, 5)$, while the exact-versus-approximate comparison in Table 3 covers the configurations for which the Gaussian approximation is used. The corresponding coding parameters and scenario-level average UAV reliability $\bar{q}$ were obtained from the simulation records. The recovery probabilities calculated from the exact combinatorial expression and the Gaussian-tail approximation are reported in Table 3. The maximum absolute difference is 0.0092, indicating close agreement between the two calculations under the MFTB simulation conditions considered in this study.

Table 3. Comparison between exact and Gaussian-approximated recovery probabilities for the recorded MFTB simulation cases.

Scenario	(n, k)	$\bar{q}$	Exact	Approx.	Absolute Error
0 failures, 20 IoTDs	(6,4)	0.7153	0.7721	0.7631	0.0090
0 failures, 20 IoTDs	(8,5)	0.7153	0.8327	0.8309	0.0018
0 failures, 30 IoTDs	(6,4)	0.7194	0.7794	0.7709	0.0085
0 failures, 40 IoTDs	(6,4)	0.7141	0.7700	0.7608	0.0092
2 failures, 20 IoTDs	(6,4)	0.7209	0.7820	0.7737	0.0083
2 failures, 20 IoTDs	(8,5)	0.7209	0.8421	0.8411	0.0010
2 failures, 30 IoTDs	(6,4)	0.7512	0.8325	0.8292	0.0032
2 failures, 40 IoTDs	(6,4)	0.7437	0.8205	0.8159	0.0046
4 failures, 20 IoTDs	(6,4)	0.7449	0.8224	0.8180	0.0044
4 failures, 20 IoTDs	(8,5)	0.7449	0.8790	0.8817	0.0027
4 failures, 30 IoTDs	(6,4)	0.7624	0.8497	0.8486	0.0011
4 failures, 40 IoTDs	(6,4)	0.7657	0.8546	0.8542	0.0004

5.2. Comparison of Fault-Tolerance Models

For each combination of terminal scale, number of failed UAVs, and fault-tolerance mechanism, the simulations were independently repeated ten times, and the reported results are presented as the mean $\pm$ standard deviation over these runs. Under the above simulation setting, two representative fault-tolerance mechanisms are adopted as baselines to evaluate the effectiveness of the proposed model.

- **Buffer-Limited Retransmission (BLR):** BLR is a retransmission-based mechanism under finite-buffer constraints [36]. Each IoTD keeps an unacknowledged local copy, while a UAV carries another copy for return transmission. The local copy shares the limited buffer with newly generated data and may be evicted under overflow. If the UAV fails, retransmission is possible only if the local copy remains available. Other settings are identical to MFTB.
- **Replication:** Replication [37] maintains multiple full copies of the same data object to improve recoverability through spatial redundancy. In this work, each data block has two full replicas, and receiving either replica at the BS is sufficient for recovery. Other settings are identical to MFTB.

Throughout the simulations, the centralized scheduler is assumed to have accurate UAV state information and reliable model parameters. Specifically, the residual-energy and position information is assumed to be accurate, and the adopted MTTF values are treated as reliable input parameters. UAV-return and data-delivery outcomes are directly recorded by the simulator and treated as correctly observed; a separate return-status or acknowledgment signaling protocol is not modeled. Malicious or inaccurate telemetry, GNSS spoofing, and falsified or corrupted feedback information are not considered in the present simulation setting.

The simulation runs for 4000 s based on the UAV endurance estimated from the energy-consumption model and parameter settings. At the end of the simulation, *Recovered* denotes data reconstructed at the BS, *Recoverable* denotes unrecovered but still restorable data, and *Lost* denotes irrecoverable data caused by UAV failures, buffer overflow, or fragment loss.

Table 4 reports the mean $\pm$ standard deviation of the fault-tolerance performance under different terminal scales and UAV-failure levels. The mean values show that MFTB generally preserves more data in recovered or recoverable states while limiting irrecoverable loss. The standard deviations provide the variation of these results across the ten repeated runs. Although Replication achieves slightly higher recovered-data volume in the

20-IoTD cases, its recoverable-data volume remains substantially lower and its lost-data volume is consistently higher than those of MFTB and BLR. This indicates that full-copy redundancy can support early recovery under a light workload, but its higher storage and transmission overhead can weaken the preservation of subsequently generated data.

Table 4. Comparison of fault-tolerance performance under different terminal scales. Values are reported as mean $\pm$ standard deviation over ten repeated runs. All data volumes are measured in GB; $\uparrow$ and $\downarrow$ indicate higher and lower values are preferred, respectively.

IoTDs	Metric Failed UAVs	MFTB			BLR			Replication		
		0	2	4	0	2	4	0	2	4
20	Recovered ($\uparrow$)	411.6 $\pm$ 4.2	410.2 $\pm$ 4.5	388.0 $\pm$ 5.8	411.3 $\pm$ 4.1	405.4 $\pm$ 4.6	387.0 $\pm$ 5.9	426.5 $\pm$ 5.3	420.6 $\pm$ 5.8	390.2 $\pm$ 7.2
	Recoverable ($\uparrow$)	253.4 $\pm$ 1.8	253.6 $\pm$ 1.9	261.3 $\pm$ 2.1	255.8 $\pm$ 1.9	257.4 $\pm$ 2.0	260.6 $\pm$ 2.2	142.8 $\pm$ 0.3	142.8 $\pm$ 0.3	142.8 $\pm$ 0.3
	Lost ($\downarrow$)	24.6 $\pm$ 0.9	25.8 $\pm$ 1.1	40.2 $\pm$ 1.6	22.5 $\pm$ 0.8	26.8 $\pm$ 1.2	42.0 $\pm$ 1.7	120.2 $\pm$ 3.8	126.1 $\pm$ 4.5	156.5 $\pm$ 6.1
30	Recovered ($\uparrow$)	422.9 $\pm$ 5.1	396.6 $\pm$ 5.6	379.4 $\pm$ 6.8	424.2 $\pm$ 5.3	401.4 $\pm$ 5.8	377.9 $\pm$ 6.9	422.0 $\pm$ 5.9	389.3 $\pm$ 6.4	375.0 $\pm$ 7.8
	Recoverable ($\uparrow$)	398.4 $\pm$ 2.9	403.5 $\pm$ 3.1	399.9 $\pm$ 3.3	390.5 $\pm$ 2.8	396.6 $\pm$ 3.0	400.7 $\pm$ 3.2	211.8 $\pm$ 0.3	211.8 $\pm$ 0.3	211.8 $\pm$ 0.3
	Lost ($\downarrow$)	204.1 $\pm$ 3.8	225.4 $\pm$ 4.6	246.1 $\pm$ 5.9	210.8 $\pm$ 4.1	227.4 $\pm$ 4.9	246.9 $\pm$ 6.2	391.6 $\pm$ 10.8	424.3 $\pm$ 12.3	438.7 $\pm$ 14.1
40	Recovered ($\uparrow$)	413.6 $\pm$ 5.8	388.6 $\pm$ 6.3	360.3 $\pm$ 7.5	401.1 $\pm$ 6.0	383.2 $\pm$ 6.5	339.5 $\pm$ 8.1	396.9 $\pm$ 6.8	385.0 $\pm$ 7.3	349.5 $\pm$ 9.2
	Recoverable ($\uparrow$)	537.0 $\pm$ 3.9	539.0 $\pm$ 4.1	540.9 $\pm$ 4.4	536.9 $\pm$ 3.8	539.3 $\pm$ 4.0	539.1 $\pm$ 4.3	279.0 $\pm$ 0.3	279.0 $\pm$ 0.3	279.0 $\pm$ 0.3
	Lost ($\downarrow$)	405.2 $\pm$ 7.2	428.1 $\pm$ 8.5	454.6 $\pm$ 10.1	417.9 $\pm$ 7.8	433.3 $\pm$ 9.2	477.3 $\pm$ 12.4	680.0 $\pm$ 18.5	691.9 $\pm$ 20.1	727.4 $\pm$ 23.8

As terminal scale increases to 30 and 40 IoTDs, the advantage of MFTB becomes more evident, particularly under higher UAV-failure levels. MFTB generally maintains more data in recovered or recoverable states and limits the growth of irrecoverable loss. In contrast, Replication exhibits a rapid increase in lost-data volume under larger terminal scales and more severe failure conditions. BLR performs better than Replication in several cases because its local copies enable possible retransmission. However, its finite-buffer constraint still limits long-term data preservation when retained copies compete with newly generated data.

Figure 6 shows the mean relative proportions of recovered, recoverable, and lost data under different terminal scales and UAV-failure levels. For each experimental condition, the plotted proportions are obtained by averaging the results over ten repeated runs. As the terminal scale and the number of failed UAVs increase, the lost-data proportion generally rises under all mechanisms, but its increase is slower under MFTB. Meanwhile, MFTB preserves a larger proportion of data in recovered or recoverable states in most conditions, indicating that fewer data blocks enter an irrecoverable state. Replication exhibits a consistently smaller recoverable-data share and a larger lost-data share, especially under larger terminal scales and more severe failure conditions. BLR performs better than Replication in several cases, but its retained local copies still occupy finite IoTD buffers and may be evicted under overflow, which limits its data-preservation capability compared with MFTB.

Figure 7 presents the mean recovery time of successfully reconstructed data blocks under different terminal scales and UAV-failure levels. For each experimental condition, the reported values are aggregated over ten repeated runs. MFTB achieves the shortest mean recovery time and exhibits relatively narrow error bars across the considered conditions. This indicates not only more timely data reconstruction, but also limited inter-block dispersion in recovery time, suggesting that the data blocks tend to reach their recovery thresholds within a relatively consistent time interval without pronounced block-level delays. Data blocks that remain unrecovered or become irrecoverably lost are excluded from this metric and are instead represented by the recoverable-data and lost-data volumes. By comparison, BLR and Replication generally exhibit longer recovery times and larger error bars, indicating slower and more heterogeneous recovery progress.

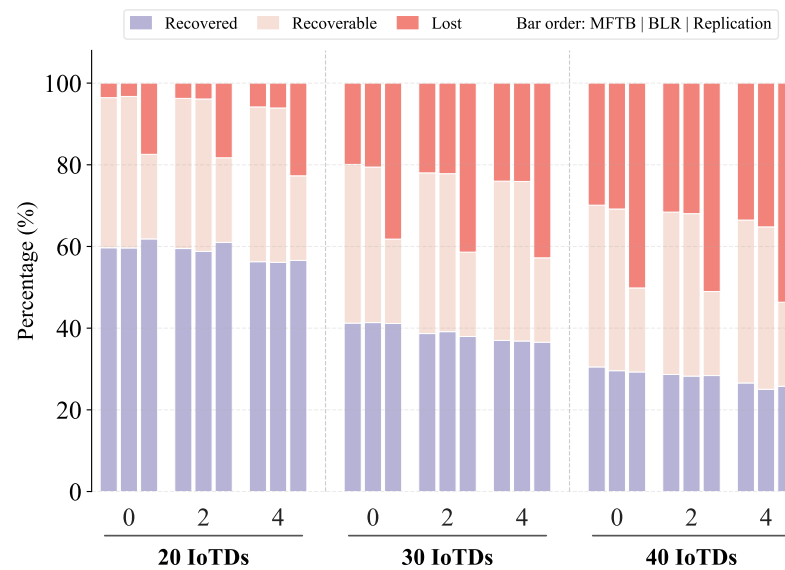


Figure 6. Data Distribution under Different Terminal Scales.

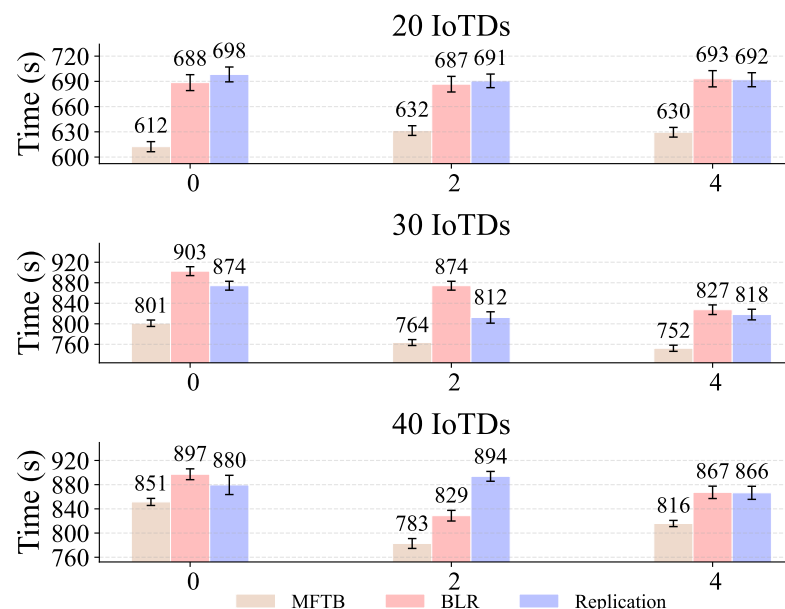


Figure 7. Average Recovery Time under Different Terminal Scales.

Overall, MFTB preserves more data in recovered or recoverable states while reducing irrecoverable loss and recovery delay, with a clearer advantage in medium- and large-scale scenarios and under severe UAV failures.

5.3. Performance Analysis of Scheduling Algorithms

Three baseline algorithms are considered for comparison with the proposed optimization method. To ensure a fair comparison, all scheduling algorithms are evaluated under the same fault-tolerance mechanism, namely MFTB. For each evaluated combination of terminal scale, failed-UAV number, and scheduling method, the simulations are independently repeated ten times. The reported performance results are summarized over these runs to characterize both their average behavior and run-to-run variation. The compared methods are summarized as follows:

- **LP-DCFS:** The proposed LP-DCFS adopts a bilevel optimization framework. In the upper-layer ATS, the maximum number of iterations is set to 10, the stagnation thresh-

old is 10, and the tabu tenure is initialized at 7 and adaptively adjusted within [3, 15]. The neighborhood-size limit is set to 50. These settings limit the number of candidate evaluations while allowing sufficient neighborhood exploration and preventing premature cycling. In the lower layer, beam-search-based fragment allocation uses $B = 3$, $R_{\text{ref}} = 2$, and $L = 2$. These values retain several candidate allocation states, reconsider the last two greedy allocation actions, and provide limited look-ahead, respectively. Larger values would increase the number of candidate states and marginal-gain evaluations; therefore, the adopted settings provide a controlled balance between solution refinement and computational cost. The same parameter values are used in all simulation cases to ensure a consistent computational budget and fair comparison.

- **ACEPSO**: ACEPSO is an improved particle swarm optimization method [38]. In this work, it is used to jointly optimize the visiting sequence and coded-fragment allocation. The swarm size is set to 20, the maximum number of iterations is 30, the inertia weight is $w = 0.5$, and the cognitive and social coefficients are $c_1 = c_2 = 1.5$.
- **ADE-DMRM**: ADE-DMRM is an improved differential evolution method [39]. It is used for joint optimization of the visiting sequence and coded-fragment allocation. The population size is set to 20, the maximum number of iterations is 30, the scaling factor is $F = 0.5$, and the crossover rate is $CR = 0.7$. A random-difference mutation strategy is adopted to generate trial solutions.
- **DQN**: DQN [40] is adopted to solve the access decision and fragment allocation problem. The learning rate is 10^{-4} , the replay-buffer size is 50,000, the batch size is 64, the discount factor is $\gamma = 0.95$, and the exploration rate decreases linearly from 0.3 to 0.05. The target network is updated every 1000 steps, and the total number of training steps is 200,000. The policy network is implemented as a multilayer perceptron (MLP), whose input state consists of node-level features and UAV-level resource features, with a total dimension of 123.

Figure 8 presents the distribution of data recovery delay under different terminal scales and UAV-failure levels. For each data block, the recovery delay is averaged over the ten repeated runs, and the violin plots summarize the resulting block-level distributions. LP-DCFS generally achieves lower mean and median recovery delays, together with more concentrated delay distributions, indicating more timely and more uniform recovery across data blocks. As the terminal scale and the number of failed UAVs increase, the recovery delays of all methods rise because of the heavier scheduling burden. Nevertheless, LP-DCFS exhibits a smaller increase in both the central tendency and the spread of recovery delay, indicating stronger scheduling robustness under larger workloads and more severe UAV failures.

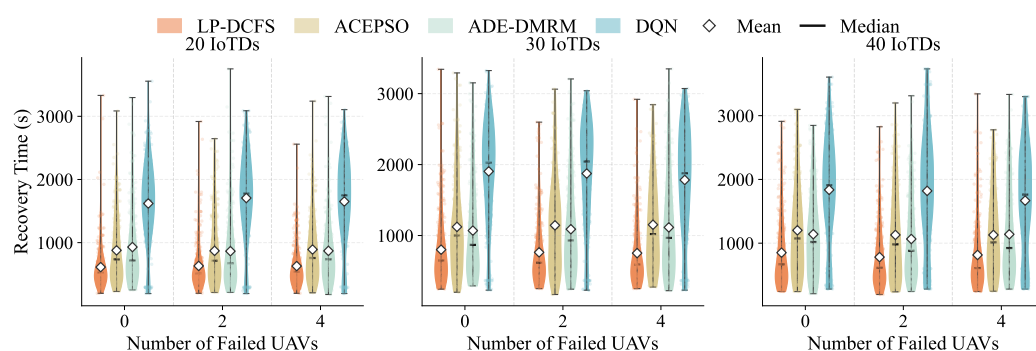


Figure 8. Distribution of Data Recovery Delay under Different Terminal Scales.

Figure 9 shows the mean per-second storage-utilization trajectories of the compared methods under different terminal scales. Each solid curve is obtained by averaging the

storage utilization at each simulation second over ten repeated runs, while the shaded band indicates the variation across the considered UAV-failure levels. Although storage utilization increases over time for all methods, the growth patterns differ markedly. DQN rises rapidly in the early stage but saturates sooner, whereas LP-DCFS increases more steadily and approaches a comparable final utilization level. This indicates that LP-DCFS maintains more balanced resource allocation throughout the recovery process and avoids excessive early-stage resource consumption.

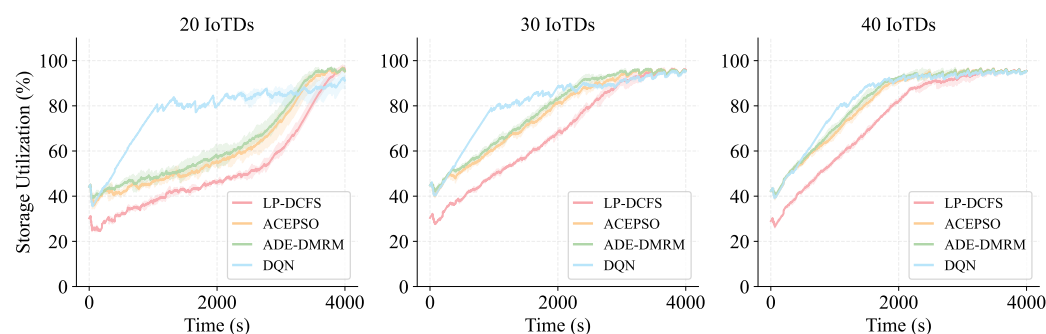


Figure 9. Storage Utilization over Time under Different Terminal Scales.

Figure 10 depicts the mean recovered data volume under different terminal scales and UAV-failure levels. Each bar represents the mean of ten repeated runs, and the error bars denote the corresponding standard deviations. LP-DCFS achieves the highest recovered data volume across all evaluated settings. Recovered data decrease for all methods as the number of failed UAVs increased, but the decline was less pronounced for LP-DCFS. Under the 40-IoTD and four-failed-UAV setting, LP-DCFS recovers a mean of 360.3 GB, compared with 269.1 GB for ACEPSO, 272.3 GB for ADE-DMRM, and 235.3 GB for DQN. The comparatively smaller standard deviations of LP-DCFS across most settings indicate lower between-run variation and support its stable recovery performance under high workload and severe UAV failures.

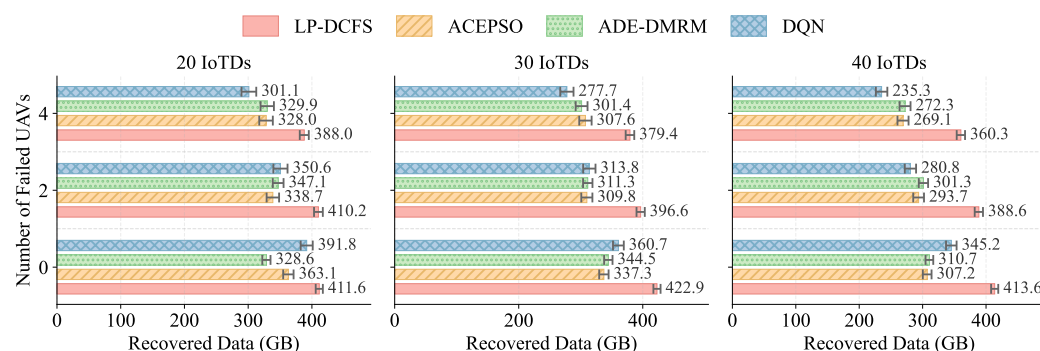


Figure 10. Comparison of Recovered Data Volume under Different Terminal Scales.

Figure 11 shows the mean total number of UAV departures under different terminal scales and UAV-failure levels. Each bar represents the mean of ten repeated runs, and the error bars denote the corresponding standard deviations. The number of departures generally increases as the terminal scale and the number of failed UAVs grow, since more disrupted data fragments need to be collected for recovery. LP-DCFS requires fewer or comparable departures than the baselines in most cases because it consolidates recovery-effective collection tasks more efficiently within each scheduling decision. This increases the marginal recovery contribution of each departure and reduces fragmented dispatches with limited recovery benefit. Together with Figure 10, the results indicate that LP-DCFS

recovers more data with lower or comparable scheduling overhead, demonstrating higher departure-level recovery efficiency.

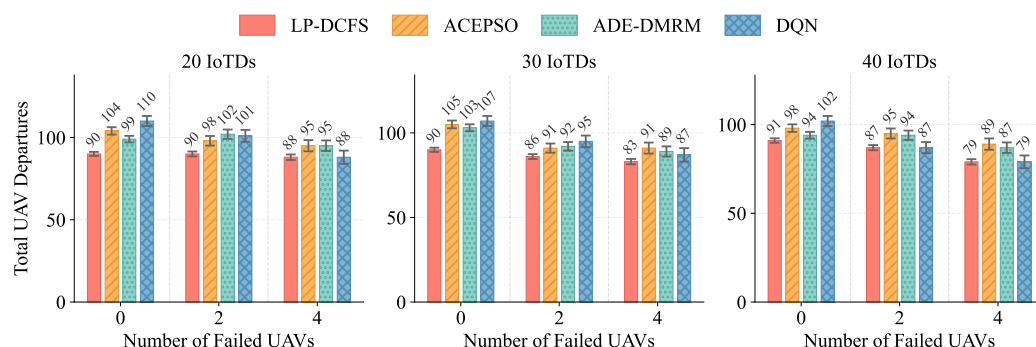


Figure 11. Cumulative Number of UAV Departures under Different Terminal Scales.

LP-DCFS therefore improves recovery timeliness and recovery yield without increasing scheduling effort, indicating a clear advantage in overall scheduling performance.

6. Conclusions

In this study, we investigated recoverability-aware fault-tolerant scheduling for UAV–IoT collaborative data collection under independent random UAV crash/non-return faults in a disaster-oriented setting. To address random UAV failures and buffer-constrained data protection, we developed a unified framework that integrates UAV reliability characterization, MDS-coded backup, and collaborative scheduling optimization. Specifically, we introduced an energy-aware reliability model to describe UAV failure risk over mission rounds, designed an MDS-coded fault-tolerant backup mechanism with adaptive protection-level selection to enhance data recoverability under limited buffer resources, and proposed a loss-pressure-guided bilevel scheduling algorithm, termed LP-DCFS, to jointly determine terminal access, visiting order, and block-level fragment allocation.

Under the modeled independent random crash/non-return faults and the evaluated simulation settings, the results indicate that the proposed framework improves data recovery performance and scheduling efficiency relative to the considered baseline methods. Within this modeled setting, the results support the joint design of failure-aware scheduling and data-layer protection for UAV-assisted data collection, rather than reliance on path planning or redundancy enhancement alone. These findings provide simulation-based evidence that the proposed framework can improve the resilience of disaster-oriented IoT data collection under the modeled fault and resource assumptions. Nevertheless, the present study is limited to simulation-based evaluation under independent random UAV crash/non-return faults, fixed IoT locations, centralized scheduling, and a LoS communication model. Correlated disaster-induced failures, adversarial interference, dynamic obstacles, and hardware-in-the-loop or field validation are not explicitly considered in the current framework. Future work will consider dynamic channels, environmental disturbances, correlated failure models, online replanning, security-aware scheduling, and distributed coordination among larger-scale heterogeneous UAV fleets.

Author Contributions: Conceptualization, H.X., W.B. and H.Y.; methodology, H.X. and W.B.; software, H.X.; validation, J.W. and X.L.; formal analysis, H.X. and J.W.; investigation, X.L. and Y.S.; resources, H.Y. and L.X.; data curation, H.X.; writing—original draft preparation, H.X.; writing—review and editing, W.B., H.Y., J.W., X.L., Y.S. and L.X.; visualization, H.X. and Y.S.; supervision, H.Y. and L.X.; project administration, H.Y.; funding acquisition, H.Y. and L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under Grants 72501290 and 72401287.

Data Availability Statement: The derived multi-UAV scenario configurations and data supporting the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors gratefully acknowledge the support provided by their institutions during this research.

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

References

1. Zheng, X.; Sun, G.; Li, J.; Wang, J.; Wu, Q.; Niyato, D.; Jamalipour, A. UAV swarm-enabled collaborative post-disaster communications in low altitude economy via a two-stage optimization approach. *IEEE Trans. Mob. Comput.* **2025**, *24*, 11833–11851. [\[CrossRef\]](#)
2. Wan, P.; Xu, G.; Chen, J.; Zhou, Y. Deep Reinforcement Learning Enabled Multi-UAV Scheduling for Disaster Data Collection with Time-Varying Value. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 6691–6702. [\[CrossRef\]](#)
3. He, H.; Peng, J.; Cai, L.; Liu, W.; Wang, C.; Gu, X.; Huang, Z. Energy-Efficient Multi-UAV Navigation for Cooperative Data Sensing and Transmission. *IEEE Trans. Mob. Comput.* **2025**, *25*, 3119–3136.
4. Pan, J.; Li, Y.; Chai, R.; Xia, S.; Zuo, L. Multiobjective trajectory planning for UAV-assisted IoT networks based on DRL approach. *IEEE Internet Things J.* **2025**, *12*, 15840–15852. [\[CrossRef\]](#)
5. Zhao, L.; Fei, Z.; Huang, J.; Wang, X.; Li, B.; Yuan, W. Deployment design for multi-UAV-assisted IoT networks: A digital twin-driven deep reinforcement learning approach. *IEEE Trans. Wirel. Commun.* **2025**, *25*, 2468–2483.
6. Bera, A.; Misra, S.; Chatterjee, C.; Mao, S. CEDAN: Cost-effective data aggregation for UAV-enabled IoT networks. *IEEE Trans. Mob. Comput.* **2022**, *22*, 5053–5063.
7. Liu, B.; Wan, Y.; Zhou, F.; Wu, Q.; Hu, R.Q. Resource allocation and trajectory design for MISO UAV-assisted MEC networks. *IEEE Trans. Veh. Technol.* **2022**, *71*, 4933–4948. [\[CrossRef\]](#)
8. Liu, W.; Wang, H.; Zhang, X.; Xing, H.; Ren, J.; Shen, Y.; Cui, S. Joint trajectory design and resource allocation in UAV-enabled heterogeneous MEC systems. *IEEE Internet Things J.* **2024**, *11*, 30817–30832. [\[CrossRef\]](#)
9. Zeng, Y.; Chen, S.; Cui, Y.; Yang, J.; Fu, Y. Joint resource allocation and trajectory optimization in UAV-enabled wirelessly powered MEC for large area. *IEEE Internet Things J.* **2023**, *10*, 15705–15722. [\[CrossRef\]](#)
10. Asim, M.; ELAffendi, M.; Abd El-Latif, A.A. Multi-IRS and multi-UAV-assisted MEC system for 5G/6G networks: Efficient joint trajectory optimization and passive beamforming framework. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 4553–4564.
11. Hao, C.; Chen, Y.; Mai, Z.; Chen, G.; Yang, M. Joint Optimization on Trajectory, Transmission and Time for Effective Data Acquisition in UAV-Enabled IoT. *IEEE Trans. Veh. Technol.* **2022**, *71*, 7371–7384. [\[CrossRef\]](#)
12. Liu, J.; Zhang, H. Height-Fixed UAV Enabled Energy-Efficient Data Collection in RIS-Aided Wireless Sensor Networks. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 7452–7463. [\[CrossRef\]](#)
13. Lin, N.; Tang, H.; Zhao, L.; Wan, S.; Hawbani, A.; Guizani, M. A PDDQNLP algorithm for energy efficient computation offloading in UAV-assisted MEC. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 8876–8890. [\[CrossRef\]](#)
14. Zhang, Y.; Mou, Z.; Gao, F.; Xing, L.; Jiang, J.; Han, Z. Hierarchical deep reinforcement learning for backscattering data collection with multiple UAVs. *IEEE Internet Things J.* **2020**, *8*, 3786–3800. [\[CrossRef\]](#)
15. Li, X.; Bao, W.; Yan, H.; Liu, Q.; Xu, Z.; Wang, J.; Zhu, X. Dynamic Demand-Aware UAV Scheduling for IoT Data Collection Using Deep Reinforcement Learning Approach. *Future Gener. Comput. Syst.* **2026**, *177*, 108248. [\[CrossRef\]](#)
16. Xin, H.; Bao, W.; Yan, H.; Wang, J.; Song, Y.; Xing, L.; Chen, H. Multi-UAV Joint Scheduling for Multimodal Data Sensing and Offloading in Post-Disaster Scenarios. *IEEE Trans. Veh. Technol.* **2026**, *75*, 14460–14474. [\[CrossRef\]](#)
17. Gong, H.; Huang, B.; Jia, B. Energy-Efficient 3-D UAV Ground Node Accessing Using the Minimum Number of UAVs. *IEEE Trans. Mob. Comput.* **2024**, *23*, 12046–12060. [\[CrossRef\]](#)
18. Wang, Y.; Huang, J.; Shan, F.; Gao, Y.; Xiong, R.; Luo, J. Optimizing Joint Speed and Altitude Schedule for UAV Data Collection in Low-Altitude Airspace. *IEEE Trans. Mob. Comput.* **2025**, *24*, 13600–13614. [\[CrossRef\]](#)
19. Yan, H.; Bao, W.; Li, X.; Zhu, X.; Zhang, Y.; Wang, J.; Liu, L. Fault-Tolerant Scheduling of Heterogeneous UAVs for Data Collection of IoT Applications. *IEEE Internet Things J.* **2024**, *11*, 26623–26644. [\[CrossRef\]](#)
20. Ramezani, M.; Amiri Atashgah, M.; Rezaee, A. A Fault-Tolerant Multi-Agent Reinforcement Learning Framework for Unmanned Aerial Vehicles–Unmanned Ground Vehicle Coverage Path Planning. *Drones* **2024**, *8*, 537. [\[CrossRef\]](#)
21. Li, M.; Richards, A.G.; Sooriyabandara, M. Experimental Validation of the Reliability-Aware Multi-UAV Coverage Path Planning Problem. In Proceedings of the AIAA SCITECH 2024 Forum, Orlando, FL, USA, 8–12 January 2024; p. 2879.

22. Hernández-Pérez, M.A.; Delgado-Reyes, G.; Borja-Jaimes, V.; Valdez-Martínez, J.S.; Cervantes-Bobadilla, M. An Adaptation of a Sliding Mode Classical Observer to a Fractional-Order Observer for Disturbance Reconstruction of a UAV Model: A Riemann–Liouville Fractional Calculus Approach. *Mathematics* **2023**, *11*, 4876. [\[CrossRef\]](#)
23. Borja-Jaimes, V.; Coronel-Escamilla, A.; Escobar-Jiménez, R.F.; Adam-Medina, M.; Guerrero-Ramírez, G.V.; Sánchez-Coronado, E.M.; García-Morales, J. Fractional-order sliding mode observer for actuator fault estimation in a quadrotor UAV. *Mathematics* **2024**, *12*, 1247. [\[CrossRef\]](#)
24. Sistu, K.; Borkotoky, S.S. Time-Constrained Erasure Correction for Data Recovery in UAV-LoRa-WuR Networks. *arXiv* **2024**, arXiv:2403.09782.
25. He, H.; Ren, Y.; Li, J.; Jiang, X.; Hou, Y.; Chen, S.; Yang, J. Hop-by-hop Redundancy-guaranteed Adaptive Coding for Enhancing Transmission Reliability of UAV Networks. *IEEE Trans. Commun.* **2025**, *73*, 5523–5537. [\[CrossRef\]](#)
26. Liu, Y.; Fang, X.; Xiao, M.; Song, F.; Cui, Y.; Xue, Q.; Tang, C. Latency optimization for multi-UAV-assisted task offloading in air-ground integrated millimeter-wave networks. *IEEE Trans. Wirel. Commun.* **2024**, *23*, 13359–13376. [\[CrossRef\]](#)
27. Zeng, Y.; Xu, J.; Zhang, R. Energy minimization for wireless communication with rotary-wing UAV. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 2329–2345. [\[CrossRef\]](#)
28. Zhao, T.; Zhang, Y.; Wang, M.; Feng, W.; Cao, S.; Wang, G. A Critical Review on the Battery System Reliability of Drone Systems. *Drones* **2025**, *9*, 539. [\[CrossRef\]](#)
29. Bünz, B.; Mishra, P.; Nguyen, W.; Wang, W. Arc: Accumulation for reed–solomon codes. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–20 August 2026; Springer: Berlin/Heidelberg, Germany, 2025; pp. 128–160.
30. Li, Y.; Wang, S.; Zhou, S.; Wang, Z. A mathematical formulation and a tabu search heuristic for the joint vessel-UAV routing problem. *Comput. Oper. Res.* **2024**, *169*, 106723. [\[CrossRef\]](#)
31. Cong, J.; Han, W.; Guo, F.; Wan, B.; Han, X.; Li, C.; Su, X. Research on dynamic scheduling strategy of multi-platform unmanned helicopters based on improved TS algorithm. *Drones* **2025**, *9*, 646. [\[CrossRef\]](#)
32. Wan, P.; Wang, S.; Xu, G.; Long, Y.; Hu, R. Hybrid heuristic-based multi-UAV route planning for time-dependent data collection. *IEEE Internet Things J.* **2024**, *11*, 24134–24147. [\[CrossRef\]](#)
33. Kuroiwa, R.; Beck, J.C. Large Neighborhood Beam Search for Domain-Independent Dynamic Programming. In Proceedings of the 29th International Conference on Principles and Practice of Constraint Programming (CP 2023). Schloss Dagstuhl—Leibniz-Zentrum für Informatik, Toronto, Canada, 27–31 August 2023; Leibniz International Proceedings in Informatics (LIPIcs); Dagstuhl Publishing: Wadern, Germany, 2026; Volume 280, pp. 23:1–23:19. [\[CrossRef\]](#)
34. Verdù, F.J.C.; Castelli, L.; Bortolussi, L. Scaling Combinatorial Optimization Neural Improvement Heuristics with Online Search and Adaptation. In Proceedings of the AAAI Conference on Artificial Intelligence, Singapore, 20–27 January 2025; Volume 39, pp. 26273–26280. [\[CrossRef\]](#)
35. Pirnay, J.; Grimm, D.G. Self-Improvement for Neural Combinatorial Optimization: Sample without Replacement, but Improvement. *Trans. Mach. Learn. Res.* **2024**. Available online: <https://openreview.net/pdf?id=agT8ojoH0X> (accessed on 23 July 2026).
36. Yates, R.D.; Sun, Y.; Brown, D.R.; Kaul, S.K.; Modiano, E.; Ulukus, S. Age of information: An introduction and survey. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 1183–1210. [\[CrossRef\]](#)
37. Rambabu, D.; Govardhan, A. Survey on Data Replication in Cloud Systems. *Web Intell.* **2024**, *22*, 1–27. [\[CrossRef\]](#)
38. Hu, G.; Cheng, M.; Sheng, G. ACEPSO: A multiple adaptive co-evolved particle swarm optimization for solving engineering problems. *Adv. Eng. Inform.* **2024**, *61*, 102516. [\[CrossRef\]](#)
39. Lin, X.; Meng, Z. An adaptative differential evolution with enhanced diversity and restart mechanism. *Expert Syst. Appl.* **2024**, *249*, 123634. [\[CrossRef\]](#)
40. Wang, Y.; Li, H.; Shen, Q. A hierarchical multi-task and multi-agent assignment approach: Learning DQN strategy from execution. *IEEE Trans. Autom. Sci. Eng.* **2025**, *22*, 14712–14722. [\[CrossRef\]](#)

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