SatBaco: Load-balanced and Latency-optimized Cooperative Service Migration for Satellite Computing

Chuanxiu Chi[1,2], Yiran Zhang[1], Yingjie Wang[1], Ao Zhou[1], Qibo Sun[1], Rajkumar Buyya[2] and Shangguang Wang[1]

[1] *State Key Laboratory of Networking and Switching Technology*
Beijing University of Posts and Telecommunications, Beijing, China

[2] *Quantum Cloud Computing and Distributed Systems (qCLOUDS) Lab*
School of Computing and Information Systems, The University of Melbourne, Australia

***Abstract*—Satellite computing reduces transmission overhead through in-orbit data processing, enhancing service performance in Earth observation, disaster monitoring, and weather forecasting. However, the harsh space environment exposes satellite nodes to stochastic failure, and limit onboard resources renders traditional fault-tolerance strategies largely inapplicable, making service migration critical for service continuity. To address dual challenges of timeliness and stability in satellite service migration, this paper proposes SatBaco, an adaptive framework that formulates migration node selection as an optimization problem minimizing both service migration latency and average satellite storage load. Accordingly, coalition game theory is employed to characterize the interactions among satellites during the migration process, recasting the optimization as a coalition equilibrium problem. Migration node coalition formation algorithm is then proposed to resolve conflicting interests between at-risk satellite and candidate nodes, deriving an initial migration decision within acceptable complexity. To adapt to dynamically satellite-ground networks, a dynamic coalition adjustment algorithm guided by risk-aware neighborhood selection is introduced. Extensive evaluations driven by realistic constellation information have demonstrated excellent performance of SatBaco, achieving 39.05% migration latency reduction and 26.34% satellite system average load pressure decrease compared to existing methods.**

***Index Terms*—Satellite computing, Resource allocation, Load balancing, Coalition game theory, Service migration.**

I. Introduction

As Low Earth Orbit (LEO) mega-constellations proliferate, satellite computing has evolved from passive communication relay into a cornerstone of space-based service infrastructure [1], [2]. Commercial deployments such as Starlink and OneWeb, spanning tens of thousands of satellites, now deliver global low-latency connectivity, remote sensing, and navigation enhancement to underserved regions [3], [4]. Yet harsh orbital environment subjects satellite nodes to stochastic failure, driven by high-energy particle radiation, extreme thermal cycling, and microgravity [5], [6], while severe constraints on onboard compute, storage, and power render terrestrial fault-tolerance strategies largely inapplicable [7]–[9]. Any loss of service state is irreversible, posing a fundamental threat to continuity. Against this backdrop, satellite service migration has emerged as a critical enabler, transferring service state from an at-risk satellite to a healthy node upon fault prediction to avert service disruption [10]–[12].

Existing service migration methods fall broadly into two categories [13]–[15]. Centralized methods depend on a central controller and timely state aggregation, such as Mapsm, ApMove, and path selection [16]–[18], which are impractical for satellite networks due to rapidly changing topologies and intermittent inter-satellite links. Distributed methods (e.g., BAACO and ADSA [1], [19]) mitigate single-point bottleneck of centralized architectures, but they lack dual-objective modeling and fine-grained awareness of satellite health indicators (temperature, radiation, storage queue), making them ill-suited for dynamic scenarios with node failures and topology changes [6], [20]. Overall, existing service migration methods suffer from dual limitations in timeliness and stability in large-scale LEO constellations, narrow communication windows (3-5 minutes [1]) impose extreme latency constraints, while imbalanced storage load risks cascading service degradation.

Leveraging the high-speed inter-satellite links and centralized control afforded by large-scale LEO constellations [21], [22], this paper employs multi-satellite collaborative service migration that dynamically allocates migration tasks to minimize latency while balancing satellite storage load. However, resource-constrained satellite nodes make multi-satellite collaboration face critical challenges [23], [24]. On one hand, warning satellites tend to select neighboring satellites with the lowest communication delay to shorten migration time. On the other hand, candidate satellites, aiming to minimize their own load, treat task rejection as a rational choice. This fundamental conflict of interest, coupled with dynamic variations in on-orbit load and failure risk causing continuous network topology evolution, renders pre-computed optimal solutions prone to failure during execution. The computational complexity of global optimization grows exponentially, failing to meet the real-time requirements of emergency migration.

Therefore, to address above challenges and achieve low-latency and load-balanced service migration, this paper proposes SatBaco, an adaptive satellite service migration framework that integrates both regular and emergency migration strategies, facilitating rapid response to warning satellites through multi-satellite collaboration The core challenge lies in determining target migration nodes and load allocation to implement migration strategy in SatBaco, t, i.e., the migration nodes selection (MNS) problem. To this end, this paper models satellite node risk states and dynamic satellite-terrestrial network topology, formulating MNS as a multi-objective optimization problem to simultaneously minimize service migration latency and average storage load. Furthermore, to address conflicts between warning satellite and candidate nodes, we reformulate MNS problem based on coalition game, strategic interactions among multiple nodes to derive coalition equilibrium solution. Specifically, we design migration node coalition formation (MNCF) algorithm via a distributed convergence mechanism guided by preference factors to obtain the optimal migration strategy, reducing computational complexity from $O(2^n)$ to $O(n \log n)$. We further propose a risk-aware coalition adjustment (RACA), which leverages node risk factors to guide neighborhood search, enhancing system adaptability to dynamic topology changes in satellite-ground networks.

Contributions of this paper are summarized as follows:

1) We propose SatBaco, an adaptive satellite service migration framework that formally models in-orbit risks and dynamic topology impacts on migration, employing coalition game theory to characterize inter-node strategic conflicts and jointly optimize migration latency and storage load balance.
2) Existence and convergence of the migration coalition game equilibrium are formally proved. Building on this, the MNCF algorithm achieves distributed optimal strategy search, RACA further enhances system resilience through risk-guided neighborhood search to accommodate dynamic topology changes.
3) Simulation experiments based on Starlink and OneWeb constellations validate the effectiveness of SatBaco. Compared to the baseline, SatBaco reduces average service migration latency by 39.05% and average storage load by 26.34%, achieving an effective tradeoff between migration latency and load balance.

II. Related Work

As satellite computing becomes key infrastructure for mission-critical applications such as Earth observation, disaster response, and weather forecasting [1], [6], service migration has emerged as a core technology for ensuring service continuity [5], [15]. However, existing research exhibits significant limitations in both timeliness and load balancing. This section reviews relevant work and analyzes limitations.

A. Centralized Service Migration and Load Balancing

Most mainstream service migration solutions adopt centralized control architectures in MEC and cloud-edge collaborative scenarios, where a central controller or orchestrator makes unified decisions based on a global view [13]–[15]. MAPSM [16] integrates RNN with geo-embedding Markov chains to predict user locations, executing service migration before handover to reduce latency. ApMove [17] provides a centralized orchestration framework for stateful edge services. yang et.al [18] a centralized migration strategy based on path selection to minimize migration latency. For load balancing, QEdgeProxy [25] uses a centralized proxy to distribute workloads based on global QoS metrics. These approaches substantially improve migration latency in terrestrial environments. However, such centralized solutions universally assume sufficient control-plane bandwidth and timely state aggregation. In satellite networks, orbital dynamics cause rapid topology changes and intermittent ground links, making timely global state acquisition impractical. Moreover, their $O(n^2)$ state-collection and decision complexity cannot be completed within the minute-level warning window.

B. Distributed and Cooperative Service Migration

To overcome the scalability and single-point bottleneck of centralized control, some works adopt distributed approaches [13]–[15], [26], [27]. BAACO [27] uses capacity-aware ant colony optimization to distributedly search migration targets across candidate nodes, reducing fault recovery time. ADSA [26] combine multi-objective optimization with deep reinforcement learning to collaboratively allocate migration or compute tasks across nodes, balancing latency and energy consumption. However, these distributed works share two common limitations. First, most optimize only one or a few network-layer metrics (e.g., latency, energy), lacking joint modeling of migration latency and storage load. Second, most rely on static weights or offline-trained policies, with insufficient fine-grained awareness of satellite health states (temperature, radiation, storage queues, etc.), making them ill-suited for dynamic adaptation to node failures and abrupt topology changes.

C. Satellite Fault Tolerance and Service Migration.

Inter-satellite link management and data migration strategies is mainly research. For link management, StarCure [6] and LSREF [10] focus on quickly restoring services through adaptive network topology adjustments and balancing computational resources with fault response time. Data migration are further divided into two types. First, periodic ground synchronization method synchronizes satellite-generated data to ground data centers on an hourly or daily basis, offering a simple mechanism with reliable ground datacenter [28]–[30]. For instance, L2D2 adopts a hybrid ground station model to reduce downlink latency, serving as a representative implementation [28]. However, its core limitation lies in the inability to complete migration within the short warning window of sudden satellite failures. On the one hand, satellite–ground communication windows are brief (typically 5–8 minutes) and constrained by limited bandwidth, making large data transfers time-consuming. On the other hand, when failure

TABLE I: A Summary of Contributions of Related Papers

Category	Works	Contribution	Limitation
Centralized service migration	[16], [17], [18]	Mobility-aware and proactive migration scheduling	Neglects orbital-window and inter-satellite coordination
Centralized load balancing	[25]	Dynamic adaptive resource allocation	Disregards orbital topology dynamics
Distributed migration	[27], [26]	Capacity-aware migration optimization	Lacks joint optimization of migration latency and load
Inter-satellite link management	[6], [10]	Adaptive topology reconfiguration under link dynamics	Overlooks service instance migration and storage load distribution
Periodic ground synchronization	[28], [29]	Scheduled data synchronization via ground contact	Constrained by ground contact windows and limited downlink bandwidth

warnings occur unexpectedly, satellites may not be within range of any available ground station, even with distributed coverage [28]. Second, multi-satellite cooperative migration utilizes inter-satellite links to perform data redundancy among pre-selected satellites, aiming to reduce dependence on ground links [26], [27], [31]–[33]. It draws from established migration mechanisms in terrestrial networks [27], [34]–[36] and is supported by extensive research on satellite resource allocation and task offloading [2], [7], [29], [37], [38], like above mature distributed migration methods.

However, existing methods face two fundamental limitations in dynamic satellite environments. First, lack joint modeling of migration latency and post-migration storage load, often resulting in data concentration on a few satellites and increasing risk of overload and failure. Second, above approaches rely on static replication strategies, overlooking dynamic evolution of storage resource states and network topology, making it difficult to handle link fluctuations and unexpected events, ultimately compromising system stability and robustness.

Overall, existing works fail to simultaneously satisfy both orbital-window timing constraints and multi-node load balancing (see Tab. 1). First, centralized schemes overlook inter-satellite collaboration potential and cannot handle hard orbital-window constraints. Second, existing distributed schemes lack load awareness, causing secondary overload on target nodes. Third, existing satellite service migration research neglects dynamic evolution of storage resource states and network topology.

To address the above limitations, this paper aims to develop a low-latency and load-balanced satellite migration.

III. System Model

We first model satellite network topology and node resources, then define failure triggers and migration constraints, and finally formulate problem as a multi-objective optimization minimizing migration latency and storage load variance.

A. Satellite Computing Network Model

Dynamic network topology. Considering dynamism and predictability, SatBaco adopts a virtual topology strategy to address dynamic of satellite network. The topology strategy discretizes dynamic topology of satellite network by dividing the orbit period into multiple time slots, *i.e.*, $\Gamma = 1, 2, \cdots, t$. Links and network topology status remain unchanged during time slot t. Network topology within a time slot t is represented as a graph $\mathcal{G}_t\,(V_t, E_t)$. Each snapshot of the satellite in time slot t is recorded as $V_t = \{\mathcal{K}_t, \mathcal{M}_t\}$, where, satellite set is denoted as $\mathcal{K}^t = \{\mathcal{K}_1^t, \mathcal{K}_2^t, ..., \mathcal{K}_k^t, \}$, ground station set is denoted as $\mathcal{M}_t = \{\mathcal{M}_1^t, \mathcal{M}_2^t, ..., \mathcal{M}_m^t\}$. An edge connecting two nodes indicates the existence of an effective data transmission path between them, including satellite-ground links (GSL) and inter-satellite links (ISL).

Satellite node resource model. Two dimensional resource model of a satellite service node is defined as $\mathcal{K}_i = \langle C_i, B_i \rangle$, where C_i denotes computational and storage capacity, and B_i is communication bandwidth.

Inter-satellite link state. It varies dynamically with orbital position, denoted as $L(s_i, s_j, t)$. The communication window function is defined as $W(\mathcal{K}_i, \mathcal{K}_j, t) \rightarrow [t_{start}, t_{end}]$, representing time window during which satellites $\mathcal{K}_i$ and $\mathcal{K}_j$ can communicate at time t.

B. Satellite Failure and Early-Warning Model

Satellite node failure probability. The failure probability $P_{fail}(\mathcal{K}_i, t)$ is jointly determined by cumulative radiation dose effects and thermal failure probability distribution:

$$P_{fail}(\mathcal{K}_i, t) = f(R_{rad}(\mathcal{K}_i, t), T_{chip}(\mathcal{K}_i), Risk(\mathcal{K}_i, C)), \quad (1)$$

where $R_{rad}(\mathcal{K}_i, t)$ is cumulative radiation dose and $T_{chip}(\mathcal{K}_i)$ is chip temperature, $Risk(\mathcal{K}_i, C))$ is storage overload risk.

Early-warning trigger conditions. Service migration is triggered when any of following conditions holds. Storage utilization exceeds the threshold, i.e., $Load(\mathcal{K}_i) > \alpha_{threshold}$. Node temperature surpasses the thermal limit, i.e., $T_{chip}(\mathcal{K}_i) > T_{threshold}$. Accumulated radiation dose exceeds the warning level, i.e., $R_{rad}(\mathcal{K}_i, t) > R_{threshold}$.

C. Communication Link Latency Model

Satellite-ground link latency. Transmission conditions of the satellite-ground link need to be clarified. Satellites must be within the communication range of the ground station to transmit data. We use a visibility model to represent when satellites can see ground stations and when they can communicate. There are two main conditions for communication. Firstly, satellite sub-point enters a designated circular region centered around the ground station, enabling communication between satellite and ground station. Secondly, the elevation of the satellite in the sky must be higher than the minimum angle of elevation from a ground station, for connectivity to be

possible. If the above two conditions are met, data transmission can be carried out. $Vis\left(\mathcal{K}_k, \mathcal{M}_m\right)^t = \{0, 1\}$ indicates the feasibility of communication between satellites and ground.

After meeting the communication conditions, we Migration important data to the ground data center through the satellite-ground link. Hence the latency is calculated as:

$$L_{latency}^{GSL} = \frac{len\left(d\right)}{R\left(\mathcal{K}_k, \mathcal{M}_m\right)} Vis\left(\mathcal{K}_k, \mathcal{M}_m\right)^t, \tag{2}$$

where $len\left(d\right)$ is size of Migration data, $R\left(\mathcal{K}_k, \mathcal{M}_m\right)$ is the date rate of the satellite to ground link.

Inter-satellite link latency. Establishing an inter-satellite link requires three conditions. First, there must be no obstruction between the satellites, and they cannot be in opposite directions. Second, the straight-line distance between the satellites must be greater than the sum of the Earth's ionospheric height and radius. Third, the satellites' antennas must be within the specified pointing angle range for communication to occur. Inter-satellite links enable communication between satellites, with the length of link corresponding to the linear distance between satellites. Inter-satellite transmission time in Migration path is the sum of following two parts. 1) Transmission delay is the time it takes to transmit all the data from first bit, which is determined by the size $len\left(d\right)$ of Migration data divided by data rate $R(\mathcal{K}, \mathcal{K}')$ of inter-satellite link. 2) Propagation delay, which is the time it takes for the electromagnetic radiation to travel the distance $q_{kk'}$, *i.e.*, the quotient of distance and the propagation rate of the electrical signal (approximately equal to the speed of light c). Here, the latency of inter-satellite links is calculated as,

$$L_{latency}^{ISL} = \frac{len(d)}{R(\mathcal{K}, \mathcal{K}')} + \frac{q_{kk'}}{c}, \tag{3}$$

where $R(\mathcal{K}, \mathcal{K}')$ is date rate for communication between $\mathcal{K}$ and $\mathcal{K}'$. This rate is defined by,

$$R(\mathcal{K}, \mathcal{K}') = B \log_2 \left(1 + \frac{P_r(\mathcal{K}, \mathcal{K}')}{k_B T_S B S_n}\right), \tag{4}$$

where $P_r(\mathcal{K}, \mathcal{K}')$ is received power, B is the bandwidth, K_B is Boltzmann's constant, T_s is the system noise temperature, and S_n is the signal-to-noise ratio margin.

D. Satellite Storage Queue.

Each satellite manages its regular tasks through a dedicated processing queue. When handling migration task from a warning satellite, incoming requests are queued and processed sequentially by storage device. As task volume increases, migration requests experience growing queuing delays, resulting in higher storage latency and prolonged completion times. Queue length directly impacts waiting time: longer queues lead to extended migration durations. If storage task queue becomes congested, processing delays increase, lengthening completion times. Under high load, migration tasks are delayed by other tasks, affecting system's overall performance.

To accurately characterize the satellite storage queue, and the waiting time for migration data storage and the time required for storage completion, we first provide an explanation of the relevant parameters:

- Arrival rate (λ). it represents arrival rate of satellite storage requests, measured in tasks per second. It quantifies the number of storage tasks arriving within a unit of time, assuming that these incoming requests are independent and follow a poisson distribution.
- Device throughput (γ_k). it denotes throughput of storage device, indicating the amount of data that device can process per second, measured in bytes per second.
- Task size (D). it refers to data volume of each storage task, which can be measured in KB, MB, GB.
- Processing rate (μ). it indicates processing rate of storage tasks, represents the number of tasks that storage device can handle per unit slot. The processing rate μ is directly related to task size D of storage tasks. Under the constraints of device throughput, as task size increases, the number of tasks that device can process per unit slot decreases. We express processing rate as $\mu = \frac{\gamma}{D}$.
- Remaining storage capacity (C_k^t). it refers to remaining storage capacity of the storage device in satellite k responsible for migration tasks.

To calculate waiting time and storage time required by storage device when processing migration requests, we employ classic M/M/1 [39]–[41] queuing model for storage queue $Q^t = \{Q_1^t, Q_2^t, ..., Q_k^t\}$. We define system utilization as $\rho = \frac{\lambda}{\mu}$. When ρ approaches 1, it indicates that device is increasingly busy, leading to longer queue waiting times. We denote the average queue length as Ł$_q$ and the average waiting time of requests in queue as W_q,

$$W_q = \frac{\rho}{\mu(1-\rho)} = \frac{\lambda}{\mu(\mu-\lambda)}, \tag{5}$$

$$L_q = \lambda W_q = \frac{\lambda^2}{\mu(\mu-\lambda)}, \tag{6}$$

where both queue length L_q and waiting time W_q are determined by the relationship between arrival rate λ and processing rate μ, with performance degrading as λ approaches μ.

The total storage time for migration data in satellite storage queue consists of two components: queue waiting time W_q and storage processing time W_{proc}. Among these, storage processing time W_{proc} represents total time taken by storage device to process all migration data requests. Given that processing rate is μ (the number of requests processed per second), processing time can be expressed as follows,

$$W_{proc} = \frac{S}{D \times \mu}, \tag{7}$$

where W_{proc} depends on total data size S and effective processing capacity $D \times \mu$, which combines per-task data volume D with the system's task processing rate μ.

Taking into account both queuing delay and processing delay, total time required to complete storage of migration can be expressed as Eq. (8), that is migration latency,

$$T_{total} = W_q + W_{proc} = \frac{\lambda}{\mu(\mu - \lambda)} + \frac{S}{D \times \mu}, \tag{8}$$

where total migration latency T_{total} combines queue delay W_q and processing time W_{proc}, where the first term captures congestion effects and the second term depends on the total data volume S and system throughput.

E. Satellite storage load.

The storage load $\hbar$ for a satellite k is quantified by the sum of the stored data, queued data, and assigned migration data.

$$\hbar_k = (C - C_k^t) + Q_k^t + S_k, \tag{9}$$

where storage load $\hbar_k$ aggregates occupied capacity $C - C_k^t$, queued data Q_k^t, and newly assigned migration data S_k, providing a combined measure of satellite's storage burden.

F. Problem Formulation

Migration Nodes Selection Problem (MNS). To efficiently back up task data within a limited time, we model selection of migration nodes for warning satellites as MNS optimization problem in SatBaco. The core optimization objectives include minimizing total migration latency and average storage load of satellites. By optimizing both objectives, system meets task deadlines and prevents performance bottlenecks from overloaded satellites. First, objective of minimizing total migration latency is to quickly transmit data to secure nodes within specified warning time limit, including both migration data transmission time and the total storage completion time. migration data transmission time depends on path latency, while storage completion time is determined by waiting and processing time for each data unit. Second, minimizing average storage load migration ensures tasks are well-distributed, preventing satellite overload. To optimize performance, SatBaco minimizes total migration latency and average satellite storage load for better system efficiency and load balancing. Given set of satellites and ground stations in every slot, connectivity and capacity of each link in every slot, and storage capabilities of each satellite and ground station in every slot, MNS problem can be formulated as:

$$\begin{aligned}
&\min \sum_{k \in \mathcal{K}^*} (L^{ISL} + T_{t,k}^{total}) + \frac{\sum_{k \in \mathcal{K}} \hbar_k}{|K|} \\
&\textbf{subject to,} \\
&\textbf{C1:}\ \forall k \in \mathcal{K}^*, C_k^t > S_k^t, t \in \Gamma \\
&\textbf{C2:}\ \forall e \in E, \sum_{s \in S} B_s < B_e \\
&\textbf{C3:}\ \sum_{k \in \mathcal{K}^*} S_k = \sum_{j \in \mathcal{K}'} \mathcal{K}_j \\
&\textbf{C4:}\ 0 < \mu, \lambda, \rho < 1
\end{aligned} \tag{10}$$

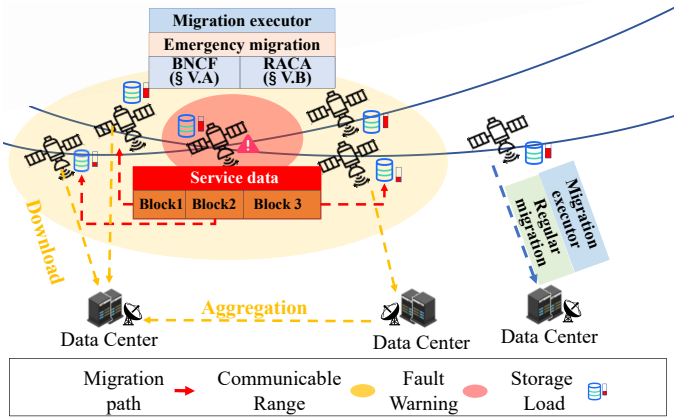


Fig. 1: **The overview of SatBaco architecture.**

The first part represents total time to complete data migration for failed satellites. When nodes in migration set are satellites, L^{ISL} is used to denote data transmission time and $T_{t,k}^{total}$ is used to denote data storage completion time. **The second part** describes storage load situation of satellites in system. When a fault alert occurs during a dynamic emergency process, it is necessary to evaluate load conditions of migration satellites after they have received desired migration data. By minimizing both parts, we obtain the optimal migration set $\mathcal{K}^*$ and the optimal allocation of migration quantities $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$. Constraint conditions C_1: remaining capacity of migration node is sufficient. C_2: ensure availability of bandwidth resources during transmission. C_3: ensure total data carried by migration satellites equals total data of warning satellites.

IV. SatBaco: Adaptive Satellite Service Migration Mechanism

This section presents SatBaco's design, covering migration strategy, and adaptive switching mechanism.

A. Framework Overview

SatBaco is an adaptive satellite service migration mechanism that ensures timely migration of service state data from failure-prone satellites through dynamic optimization of target node selection and data block allocation. The core idea is to make hierarchical decisions on migration methods based on satellite status, leveraging multi-satellite collaboration for rapid large-scale migration while reducing storage pressure through distributed migration. Fig. 1 illustrates SatBaco system architecture, consists of a satellite segment and a ground segment. Satellite segment is constructed from large constellations managed by companies such as SpaceX or OneWeb, and ground segment comprises geographically distributed data centers. Satellites store their own data and also undertake migration tasks from neighboring satellites. SatBaco ultimately migrates onboard mission data to ground data centers to ensure reliability of subsequent computations.

Specifically, migration executor is deployed on each satellite and adaptively selects migration strategies based on satellite

status. Two strategy types are supported. During normal operation, executor performs standard incremental migration of task and instruction data while maintaining ground station communication. During warning, executor invokes migration target node selection algorithm (MNCF, RACA) based on coalition game to form migration node coalition for emergency.

B. Dual-Mode Migration Strategy

Regular Cooperative Migration (RCM) performs incremental migration, synchronizing only changed task data and instruction states within scheduled communication windows. This keeps ground snapshot up to date and reduces ERM pressure, as a full TB-scale transfer during failure is nearly impossible within the 3–5 minute warning window. This design minimizes disruption to normal operations while accelerating fault-time migration.

Emergency Migration Strategy (ERM). The core challenge of ERM is that warning window is extremely short (typically 3–5 minutes), yet service state data can reach GB scale, making single-link ground transmission impossible within the deadline. To address this, ERM adopts a two-phase cooperative transfer. In the first phase, data is distributedly offloaded to multiple neighboring satellites via parallel inter-satellite links, forming a node coalition. In the second phase, coalition nodes relay data to ground through their respective communication windows. ERM transforms single-link serial transfer into multi-link parallel transfer.

In SatBaco, RCM and ERM are not simply sequential but are dynamically selected based on satellite health indicators including storage utilization, temperature, and radiation dose. While all indicators remain within normal thresholds, the system maintains RCM mode. Once any indicator exceeds its warning threshold, ERM is triggered immediately.

V. Coalition Game-based Migration Target Selection

In ERM, migration node selection faces interest conflicts. Warning satellite prefers neighbors with lowest communication latency to minimize migration time, while candidate satellites rationally choose to reject migration tasks in order to maximize their own utility. This conflict renders traditional optimization methods ineffective. Centralized methods rely on global state, but satellite load and failure risk change continuously, causing precomputed optimal solutions to become invalid at execution time. Distributed global search carries a computational complexity of $O(2^n)$, which cannot meet requirements of 3–5 minute emergency window.

Therefore, SatBaco models MNS as a coalition game. Each candidate node acts as an independent player, autonomously deciding whether to join the migration coalition based on local information, with coalition stability guaranteed by game-theoretic equilibrium. SatBaco requires no central coordinator, naturally aligning with the decentralized nature of satellite networks, while reducing computational complexity from $O(2^n)$ to $O(n \log n)$ to satisfy real-time emergency migration requirements. The definition of coalition game is as follows.

Definition 3.1: Coalition Game for Migration Node Selection (CGMS). The process of identifying various coalitions of migration nodes for warning satellites' data is defined as a quintuple $CG =< \mathcal{P}, C_p, \mathcal{F}, U_p, U(\mathcal{F}) >$, and formulation of CGMS is as follows,

- **Players.** $\mathcal{P}$ is the set of nodes that could establish a communication connection with warning satellites, including satellites that can establish ISL connections. Each player is independent and responsible for handling data migration tasks of warning satellites.
- **Strategy.** The strategy space for players includes the maximum storage capacity resources C_p they can provide, and decision on which coalition to form, that is, whether to join or leave a coalition.
- **Coalition.** The coalition partition is defined as the set of $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_j, \ldots, \mathcal{F}_f\}$, where it partitions player set of workers $\mathcal{P}$, *i.e.*, $\forall j, F_j \subseteq \mathcal{P}$. Each coalition is independent. A coalition $F_j \subseteq \mathcal{P}$ is formed by multiple players with goal of achieving rapid data migration tasks from failed satellites. The members of each coalition coordinate together to execute migration tasks.
- **Player Utility Function.** The utility of each player in migration data coalition depends on their storage capacity, transmission time, and satellite storage load. The utility function for player $p \in \mathcal{P}$ can be represented as U_p, where indicates contribution degree of node p within coalition. Utility is directly proportional to storage capacity of satellite C_p^t and inversely proportional to transmission and storage time $L_{t,p} + W_{t,p}^{total}$ and storage load $(Q_p^t + S_p^t)$. Nodes with greater storage capacity, shorter transmission times, and lower loads will be tasked with undertaking more migration responsibilities.

$$U_p = \frac{\alpha_1 C_p^t}{\alpha_2 (L_{t,p} + W_{t,p}^{total}) + \alpha_3 (Q_p^t + S_p^t)} \tag{11}$$

- **Total Utility of Coalition.** The total utility of coalition is constituted by the sum of utilities of its member nodes. This reflects overall migration capability and resource utilization level of coalition.

where $\alpha_1 + \alpha_2 + \alpha_3 = 1, 0 < \alpha_1, \alpha_2, \alpha_3 < 1$. Then, we provide definition of coalition Nash equilibrium.

Definition 3.2: Nash Equilibrium of CGMS. Nash equilibrium is achieved when following two conditions are satisfied in coalition games:

1. Individual Rationality. Each player's utility in the coalition is no less than their utility from acting alone. That is, for each player p and coalition F, $U_p(\mathcal{F}) \geq U_p(\{p\}), \forall p \in \mathcal{F}$.

2. Coalition Stability. No player can increase their utility by acting alone, leaving coalition, or forming a new coalition with other players. That is, $U_p(\mathcal{F}) \geq U_p(\mathcal{F}'), \forall \mathcal{F}' \subseteq \mathcal{P}, \forall i \in \mathcal{F}'$. No player can gain higher utility by leaving the current coalition F, acting alone, or forming a new coalition F' with other players.

Furthermore, we prove that equilibrium solution of CGMS corresponds to the optimal solution of MNS problem, supported by formal theorem 1 and proofs.

Algorithm 1 MNCF: Migration Node Coalition Formation Algorithm.

Input: $\mathcal{P}, S, Vis(\mathcal{K}, \mathcal{K}'), C_p, \{\alpha_1, \alpha_2, \alpha_3\}, \{\theta_1, \theta_2, \theta_3\}$
Output: $\mathcal{F}, \Psi = \{\varphi_1, \varphi_2, \ldots, \varphi_p\}$
Initialize $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_j, \ldots, \mathcal{F}_f\}$
Initialize $\mathcal{F}_{cur} \leftarrow \mathcal{F}, n \leftarrow 0$
repeat
  $p \in \mathcal{P}$ chooses two coalitions, $\mathcal{F}_j \in \mathcal{F}_{cur}, \mathcal{F}_{j'} \in \mathcal{F}_{cur}$
  $\mathcal{F}_j \neq \mathcal{F}_{j'}, n \leftarrow n + 1$
  if $\mathcal{F}_j$ transitions to $\mathcal{F}_{j'}$ & $\mathcal{F}_{j'} \succ_p \mathcal{F}_{j'}$ **then**
    $\mathcal{F}_{j'}.add(p)$
    $\mathcal{F}_{cur} \leftarrow (\mathcal{F}_{cur} \setminus \{\mathcal{F}_j, \mathcal{F}_{j'}\}) \cup (\mathcal{F}_j \setminus \{p\}, \mathcal{F}_{j'} \cup \{p\})$
  else
    $\mathcal{F}_j.add(p)$
  end if
until $\max U(\mathcal{F})$
Get $\Psi = \{\varphi_1, \varphi_2, \ldots, \varphi_p\}$ by Eq. (14)

Theorem 1: Under the condition that U_p is concave and monotonically increasing in C_p^t and decreasing in latency and storage load, maximizing the coalition utility

$$U(\mathcal{F}) = \sum_{p \in \mathcal{F}} U_p$$

is equivalent to minimizing the MNS objective (Eq. 10) up to a constant factor determined by C_p^t.

Proof: Since $U_p > 0$ and is monotonically decreasing in both latency and storage load, maximizing $U(\mathcal{F}) = \sum_{p \in \mathcal{F}} U_p$ is equivalent to minimizing $\sum_{p \in \mathcal{F}} U_p^{-1}$. Taking the reciprocal of U_p and multiplying by a positive constant $\beta > 0$ gives:

$$\frac{\beta}{U_p} = \frac{\alpha_2 \beta}{\alpha_1 C_p^t} \left(L^{\mathrm{ISL}} + T_{t,p}^{\mathrm{total}} \right) + \frac{\alpha_3 \beta}{\alpha_1 C_p^t} \hbar_p.$$

Setting coefficient of latency term to unity and that of storage load term to $\frac{1}{|\mathcal{K}|}$, $\frac{\alpha_2 \beta}{\alpha_1 C_p^t} = 1, \frac{\alpha_3 \beta}{\alpha_1 C_p^t} = \frac{1}{|\mathcal{K}|}$, the summation $\sum_{p \in \mathcal{F}} \frac{\beta}{U_p}$ reduces to $\sum_{k \in \mathcal{K}^*} \left(L^{\mathrm{ISL}} + T_{t,k}^{\mathrm{total}} \right) + \frac{\sum_{k \in \mathcal{K}} \hbar_k}{|\mathcal{K}|} = \mathcal{J}$, which is precisely MNS objective. Since β and C_p^t are positive constants with respect to the optimization, minimizing $\sum_p \frac{\beta}{U_p}$ is equivalent to minimizing $\mathcal{J}$. Therefore, maximizing $U(\mathcal{F})$ is equivalent to minimizing the MNS objective up to the constant factor.

A. Migration Node Coalition Formation Algorithm

Preference factor-guided distributed convergence. Traditional coalition formation algorithms typically prioritize maximizing coalition utility, neglecting the benefits of individual members. In satellite networks, this leads to overloaded core nodes while edge nodes remain idle. MNCF's core innovation lies in introducing a preference factor, allowing nodes to consider both coalition utility and size when making decisions,

$$\delta = \theta_1 \varphi_p - \theta_2 S + \theta_3 \cdot \mathrm{Num}(\mathcal{F}_j), \tag{12}$$

where φ_p is node utility, S is total amount of migrated data within coalition, and $\mathrm{Num}(\mathcal{F}_j)$ is coalition size, all normalized. The key insight is that introduction of third term encourages nodes to form moderately sized coalition, avoiding overload of core nodes.

Convergence and complexity analysis. Iterative process of MNCF is essentially a potential function descent. Define potential function of a coalition structure as sum of all coalition utilities:

$$\Phi(\mathcal{F}) = \sum_{\mathcal{F}j \in \mathcal{F}} U(\mathcal{F}j). \tag{13}$$

here, each coalition switch (a node transferring from $\mathcal{F}j$ to $\mathcal{F}j'$) necessarily satisfies $\delta \mathcal{F}j'^p > \delta^p_{\mathcal{F}j}$, meaning node achieves a higher preference factor in new coalition. Since preference factor is positively correlated with coalition utility, it follows that $\Phi(\mathcal{F}new) \geq \Phi(\mathcal{F}_{old})$. As number of coalition structures is finite, algorithm converges to a local optimum in a finite number of steps.

Computational complexity of MNCF is $O(n \log n)$, significantly better than $O(2^n)$ of traditional exhaustive search. Specifically, each node compares $O(\log n)$ candidate coalitions on average, across n nodes, converging within a constant number of iterations.

Finally, Shapley values are used to fairly distribute coalition payoffs,

$$\varphi_p = \sum_{\mathcal{F} \subseteq \mathcal{P} \setminus p} \frac{|\mathcal{F}|!(|\mathcal{P}| - |\mathcal{F}| - 1)!}{|\mathcal{P}|!} [U(\mathcal{F} \cup p) - U(\mathcal{F})], \tag{14}$$

where migration task assigned to node p is $\varphi_p S$.

B. Dynamic Risk-Aware Coalition Adjustment Algorithm

MNCF operates under a static network assumption, whereas satellite networks exhibit dynamic evolution in link states, node loads, and failure probabilities. RACA addresses this through risk-guided neighborhood search, which drives directional convergence across the state space rather than relying on the random perturbation of standard simulated annealing.

Risk-guided neighborhood selection. Standard simulated annealing generates neighborhoods randomly, exploring all neighboring solutions with uniform probability. This is inefficient in large state spaces due to excessive invalid exploration. RACA uses a risk factor to guide the search direction. The node risk is defined as:

$$\mathrm{Risk}_p = \hbar_p + \frac{\eta_p}{\eta_{\max}}, \tag{15}$$

where $\hbar_p$ is normalized storage load and η_p is failure rate. The risk-guided neighborhood generation rules are:

- High-risk nodes ($\mathrm{Risk}_p > \ell$) are removed from current coalition with probability 1.
- Low-risk nodes are included as candidates with high probability.

Satellites with high storage load are already near capacity, and accepting additional migration task greatly increase their failure risk. Nodes with high failure rates should not be assigned extra tasks.

Algorithm 2 RACA: Risk-Aware Coalition Adjustment Algorithm

Input: $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_j, \ldots, \mathcal{F}_f\}, P(C_p^{max}, \eta_p, \hbar_p), \epsilon, \ell, Vis(\mathcal{P}, \mathcal{P}'), \alpha_{1,2,3}, \theta_{1,2,3}, \Psi_{\mathcal{F}_f} = \{\varphi_1, \varphi_2, \ldots, \varphi_p\}, S$
Output: $\mathcal{F}^*, \Psi^*_{\mathcal{F}_f} = \varphi_1^*, \varphi_2^*, ..., \varphi_p^*$
Compute Risk_p^t for all $p \in \mathcal{F}$.
$\mathcal{F}^* \leftarrow \{p \in \mathcal{F}_0 \mid \text{Risk}_p \leq \ell\}$.
if $\mathcal{F}^* = \mathcal{F}_0$ **then**
  return $\mathcal{F}_0$ {No adjustment needed.}
else
  $Tem \leftarrow Tem_0$.
  while $T > T_{\min}$ **do**
    Randomly select index $p \in \{p \in P | vis(p, p') = 1\}$.
    if $\text{Risk}_p > \ell$ **then**
      $c'[p] \leftarrow 0$ {Remove high-risk node.}
    else
      Propose new $c'[p] = \varphi_p + \mathcal{U}(-(s/|\mathcal{F}|), -(s/|\mathcal{F}|))$.
      Clip $c'[p]$ to $[0, C_p^{max} - \hbar_p]$.
    end if
    Compute $\Phi(c'[p])$ and $\Delta = \Phi(c'[p]) - \Phi(\varphi_p)$.
    if $\Delta \leq 0$ **then**
      Accept $c'[p]$.
    else
      Accept $c'[p]$ with probability $\exp(-\Delta/Tem)$.
    end if
    $Tem \leftarrow \zeta \cdot Tem$.
  end while
end if
return $\Psi^*_{\mathcal{F}_f} \leftarrow \{c'[p] \mid p \in \mathcal{F}^*\}$.

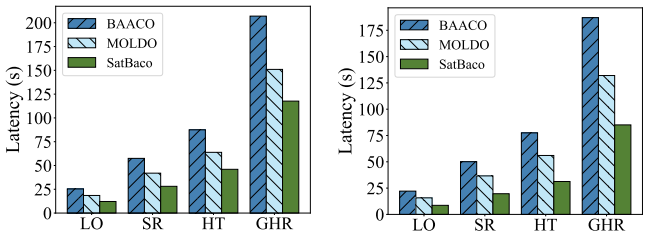
TABLE II: Parameters

Parameters	Value
Starlink (Phase I)	Inclination 53°, Altitude 550 km, Orbit 72, Satellite 22
OneWeb	Inclination 87.9°, Altitude 1200 km, Orbit 18, Satellite 40
ISL transmission rate $R(K, K')$	2 Gbps
GSL transmission rate $R(K, M)$	300 Mbps
Satellite storage capacity C	450 GB (Maximum)
Arrival rate γ	[3,6]/s
Task size D	1 GB
Throughput R	500 MB/s
Game rounds	150 (Maximum)
Risk thresholds ℓ	0.6, 0.8
Preference weight parameter θ	θ_1 = 0.6, θ_2 = 0.2, θ_3 = 0.2
Game utility weight parameter α	α_1 = 0.2, α_2 = 0.6, α_3 = 0.2

Convergence Analysis. Introducing risk guidance into simulated annealing adds a bias toward low-risk regions over Boltzmann distribution. Define modified objective function,

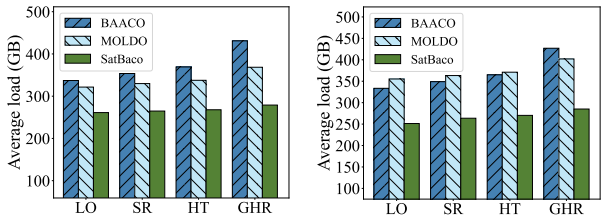
$$\Phi = \underbrace{\sum_{p \in \mathcal{F}^*} [\max u_p(c_{p'}, c_{-p}) - u_p(c_p)]}_{\text{Nash deviation}} + \underbrace{\epsilon \sum_{p \in \mathcal{F}^*} \text{Risk}_p}_{\text{coalition risk}}, \quad (16)$$

where first term ensures game-theoretic equilibrium and second penalizes high-risk members. Risk-guided neighborhood selection preserves Markov chain structure of simulated annealing, every solution retains a positive visitation probability, but frequency is biased toward low-risk regions. As $T \to 0$, algorithm converges to global (or near-global) optimum with probability 1.



(a) Service migration latency under OneWeb (b) Service migration latency under Starlink

Fig. 2: Service migration latency under different dataset scales



(a) Average storage load under OneWeb (b) Average storage load under Starlink

Fig. 3: Average storage load under different dataset scales

Complexity Analysis. For n satellites network, coalition search space is 2^n. Standard SA effectively explores only $1/2^n$ of this space. RACA's risk guidance compresses search space to $O(n)$, each node only needs to evaluate whether it satisfies the risk constraint, eliminating exhaustive enumeration. Empirically, RACA converges 1–2 orders of magnitude faster than standard SA due to this reduction in invalid exploration.

In summary, MNCF and RACA form complementary structure of static optimality and dynamic stability. MNCF finds optimal coalition for a given network state, while RACA adaptively adjusts coalition as network conditions evolve.

VI. Performance Evaluation

A. Experiment Setup

Based on the STK (System Tool Kit) [42] orbit analysis tool, we simulate the movement of satellites over time to generate a satellite network diagram. The constellation configuration parameters are set according to the state-of-the-art commercial mega-constellations—Starlink [43] and OneWeb [44]. Table II details the key parameters of the selected mega-constellations. Specifically, the Starlink constellation consists of 1,584 satellites distributed across 72 orbital planes, with 22 satellites per plane. The orbitals altitude are set to 550Km, and the orbitals inclination are set to 53°. The OneWeb constellation comprises 720 satellites distributed across 18 orbital planes, with 40 satellites per plane, the orbitals altitude are set to 1200Km and the orbitals inclination are set to 87.9 degrees. According to limitations such as distance and power, it is specified that each

satellite can establish connections with other satellites within its antenna scanning range. We set the GSL transmission rate at 300Mbps based on the actual measurement data from the Tiansuan constellation. Meanwhile, the ISL transmission rate is set at 2 Gbps according to [45]. The specific parameter settings are shown in the Table II.

Datasets. We reference the scale of four types of real-world Earth observation services for performance testing, Gaofen-1 high-resolution image datasets (GHR, 107.3GB) [46], Sentinel-2 radar image datasets (SR, 29.9GB) [47], Landsat optical image datasets (LO, 19.9GB) [48], HeDe-2 telemetry dataset (HT, 45.5GB). HeDe-2 is our private dataset and will be released.

Baseline Algorithms. Following algorithms are selected,

- BAACO [27]: A service migration node selection algorithm based on ant colony optimization, which only considers transmission distance.
- MOLDO: A multi-objective optimization algorithm based on deep reinforcement learning, derived from ADSA mechanism [26], which focuses on global optimality but employs static weights.

B. Experimental Results

1) Service migration latency over different constellations: Fig. 2 shows that SatBaco reduces service migration latency by 48.66% and 29.44% on average compared with BAACO and MOLDO. This improvement stems from RCM reducing transmission data volume, and ERM dynamically considering node storage load and transmission distance to optimize migration strategies in real time. In contrast, BAACO only considers transmission distance and cannot handle high-traffic scenarios, MOLDO uses static weights and cannot adjust dynamically.

2) Load balancing performance: Fig. 3 shows the average storage load of each migration node under different dataset scales. SatBaco consistently maintains the lowest average load, reducing it by 24.54% compared to BAACO and 28.13% compared to MOLDO. This benefit comes from ERM considering both node storage capacity and transmission distance during task allocation, effectively preventing overload.

3) Service completion time across different service configurations: Completion time is defined as maximum aggregation time for migration nodes to transfer all service data to ground station. As shown in Fig. 4, SatBaco achieves improvements of 32.72% compared to MOLDO and 58.22% compared to BAACO. Load balancing reduces the data volume assigned to each node, significantly shortening aggregation time.

4) Impact of coalition size on performance: Fig. 5 and Fig. 6 shows impact of player number on service migration latency and average load. This results reveal that both latency and average load decrease as number of players increases, because more satellites participate in service migration, thereby sharing computing and storage loads more effectively and achieving more uniform load distribution. However, more players are not always beneficial, as increasing the number of players inevitably elevates computational complexity, necessitating a trade-off between performance gains and overhead costs.

5) Impact of risk threshold of RACA on system resilience: RACA addresses a core challenge, satellite network topology changes dynamically, and pre-computed optimal combinations may become invalid during execution. RACA controls inclusion of high-risk nodes in coalition through risk threshold ℓ. Fig. 7 indicates that when threshold $\ell = 0.6$, the overall resource occupancy rate is lower than that at $\ell = 0.8$, because a lower threshold excludes more high-risk nodes, thereby reducing the number of coalition members and causing resource occupancy rate of remaining nodes to increase. Nevertheless, under both threshold settings, SatBaco's resource occupancy rate remains lower than those of BAACO and MOLDO, demonstrating superior performance. This verifies effectiveness of risk factor design in maintaining coalition stability by excluding high-risk nodes and enhancing system resilience.

6) Convergence analysis of MNCF under different satellite network configurations: MNCF addresses first limitation of existing work, traditional centralized optimization algorithms have complexity $O(2^n)$ that cannot meet real-time requirements, and they fail to consider load balancing, leading to overload of popular nodes. MNCF iteration process is essentially a potential function descent. Fig. 8 shows that change in the sum of coalition utilities with number of iterations, MNCF converges to a stable solution within 60-100 iterations, verifying convergence analysis of potential function descent.

7) Verification of SatBaco allocation's individual rationality: We compare individual allocation and allocation in proposed CGMS. Fig. 9(a) shows that CGMS allocation reasonably distributes computing load, reduces maximum transmission latency, and ensures each node receives a fair allocation result based on its own contribution. Fig. 9(b) shows that CGMS allocation makes distribution of storage contributions more balanced, reflecting individual rationality, the utility of each node in coalition is no less than that when acting alone.

VII. Conclusions and Future Work

This paper proposed a solution for improving the reliability of satellite services. Service can be prevented by applying adaptive distributed SatBaco mechanism. Minimize the storage resources consumed and balance system storage load by service migration while ensuring service reliability. By applying MNCF and RACA algorithms, we achieve coalition game's dynamic equilibrium, providing the optimal strategy for migration node selection. Experiments have shown that SatBaco achieve a 39.05% reduction in migration latency, a 26.34% decrease in system average load, and superior performance in data aggregation response speed, system resilience compared to state-of-the-art methods.

As part of future research, we will explore integration time-series forecasting and anomaly detection to enable proactive backup scheduling that optimizes transmission paths, while also developing a multi-tiered backup architecture combining in-orbit, inter-satellite, and ground storage with lightweight protocols and resource-sharing strategies to enhance system robustness and scalability.

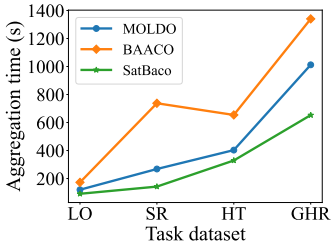


Fig. 4: Average aggregation time of service dataset

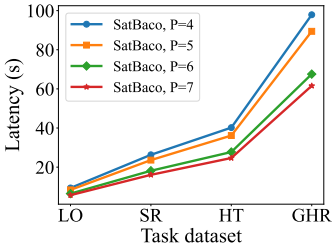


Fig. 5: Migration latency with varying player numbers

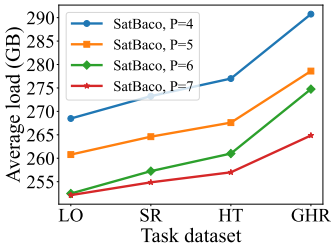


Fig. 6: Migration loads with varying player numbers

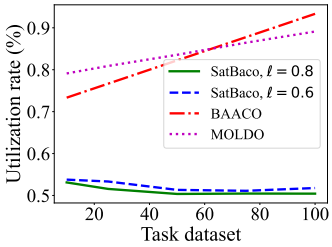
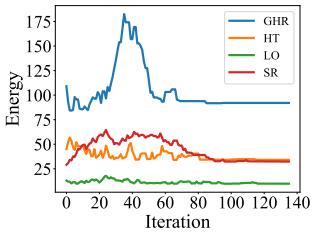
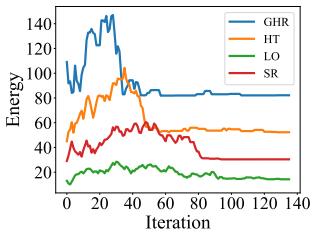


Fig. 7: Storage utilization rate of migration satellites

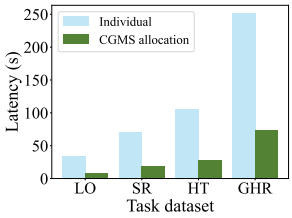


(a) Convergence under OneWeb

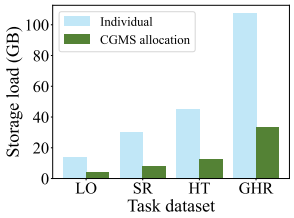


(b) Convergence under Starlink

Fig. 8: Convergence verification under different constellations



(a) Latency under different allocation methods



(b) Storage contribution under different allocation methods

Fig. 9: Individual rationality verification of players under different datasets

References

[1] Q. Zhang, S. Wang, J. Guan, P. K. Donta, X. Ma, R. V. Prasad, S. Dustdar, and X. Liu, "Satcooper: Enhancing cooperative inference analytics for satellite service via multi-exit dnns," *IEEE Transactions on Mobile Computing*, vol. 24, no. 9, pp. 8314–8328, 2025.

[2] J. Zhou, Q. Yang, L. Zhao, H. Dai, and F. Xiao, "Mobility-aware computation offloading in satellite edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 10, pp. 9135–9149, 2024.

[3] C. Yang, J. Yuan, Y. Wu, Q. Sun, A. Zhou, S. Wang, and M. Xu, "Communication-efficient satellite-ground federated learning through progressive weight quantization," *IEEE Transactions on Mobile Computing*, vol. 23, no. 9, pp. 8999–9011, 2024.

[4] C. Wang, Y. Zhang, Q. Li, A. Zhou, and S. Wang, "Satellite computing: A case study of cloud-native satellites," in *2023 IEEE International Conference on Edge Computing and Communications (EDGE)*, pp. 262–270, IEEE, 2023.

[5] P. Yue, J. An, J. Zhang, J. Ye, G. Pan, S. Wang, P. Xiao, and L. Hanzo, "Low earth orbit satellite security and reliability: Issues, solutions, and the road ahead," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1604–1652, 2023.

[6] Z. Lai, H. Li, Y. Wang, Q. Wu, Y. Deng, J. Liu, Y. Li, and J. Wu, "Achieving resilient and performance-guaranteed routing in space-terrestrial integrated networks," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pp. 1–10, IEEE, 2023.

[7] H. Hu, K. Song, C. Zhan, R. Fan, and J. Yang, "Joint service caching and resource allocation over different timescales in satellite edge computing networks," *IEEE Transactions on Mobile Computing*, 2025.

[8] R. Xing, M. Xu, A. Zhou, Q. Li, Y. Zhang, F. Qian, and S. Wang, "Deciphering the enigma of satellite computing with cots devices: Measurement and analysis," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pp. 420–435, 2024.

[9] Q. Li, S. Wang, X. Ma, A. Zhou, Y. Wang, G. Huang, and X. Liu, "Battery-aware energy optimization for satellite edge computing," *IEEE Transactions on Services Computing*, vol. 17, no. 2, pp. 437–451, 2024.

[10] L. Yang, A. Zhou, X. Ma, Y. Zhang, and S. Wang, "Optimizing space-borne computation: A reliability enhancement framework for leo constellation," in *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 2371–2378, IEEE, 2023.

[11] Y. Li, X. Sun, Y. Xia, P. Chen, Y. Li, and Q. Peng, "M-mnft: A novel modified (m, n)-fault tolerance approach for service migration in vehicular edge computing," in *2023 IEEE International Conference on Software Services Engineering (SSE)*, pp. 170–179, 2023.

[12] S. A. Khan, T. Mikkonen, N. Makitalo, and H. T. Heinonen, "Blockchain and relational databases: Causing a revolution in healthcare data management and data security," in *2025 IEEE International Conference on Software Services Engineering (SSE)*, pp. 12–23, IEEE, 2025.

[13] J. Wang, J. Hu, G. Min, Q. Ni, and T. El-Ghazawi, "Online service migration in mobile edge with incomplete system information: A deep recurrent actor-critic learning approach," *IEEE Transactions on Mobile Computing*, vol. 22, no. 11, pp. 6663–6675, 2022.

[14] Z. Chen, S. Huang, G. Min, Z. Ning, J. Li, and Y. Zhang, "Mobility-aware seamless service migration and resource allocation in multi-edge iov systems," *IEEE Transactions on Mobile Computing*, vol. 24, no. 7, pp. 6315–6332, 2025.

[15] Y. Li, X. Sun, Y. Xia, P. Chen, Y. Li, and Q. Peng, "M-mnft: a novel modified (m, n)-fault tolerance approach for service migration in vehicular edge computing," in *2023 IEEE International Conference on Software Services Engineering (SSE)*, pp. 170–179, IEEE, 2023.

[16] X. Zhao, Y. Shi, S. Chen, J. Liu, B. Ji, and S. Mumtaz, "Mapsm: Mobility aware proactive service migration framework for mobile edge computing in consumer internet of vehicles," *IEEE Transactions on Consumer Electronics*, 2025.

[17] M. Zakarya, L. Gillam, A. A. Khan, O. Rana, and R. Buyya, "Apmove: A service migration technique for connected and autonomous vehicles," *IEEE Internet of Things Journal*, vol. 11, no. 17, pp. 28721–28733, 2024.

[18] J. Xu, X. Ma, A. Zhou, Q. Duan, and S. Wang, "Path selection for seamless service migration in vehicular edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 9040–9049, 2020.

[19] D. Bhattacherjee, W. Aqeel, I. N. Bozkurt, A. Aguirre, B. Chandrasekaran, P. B. Godfrey, G. Laughlin, B. Maggs, and A. Singla, "Gearing up for the 21st century space race," in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, pp. 113–119, 2018.

[20] F. Zhou, Q. Li, T. Xin, X. Wei, and H. Zhang, "Analyses and countermeasures of in-orbit satellite failures caused by space radiation environment," *Spacecraft Environment Engineering*, vol. 29, no. 4, pp. 392–396, 2018.

[21] W. X. Zheng, A. Taneja, M. Masood, A. Sabnis, R. K. Sitaraman, and D. Vasisht, "Starcdn: Moving content delivery networks to space," in *Proceedings of the ACM SIGCOMM 2025 Conference*, pp. 948–962, 2025.

[22] Y. Li, Y. Chen, J. Yang, J. Zhang, B. Sun, L. Liu, H. Li, J. Wu, Z. Lai, Q. Wu, *et al.*, "Small-scale leo satellite networking for global-scale demands," in *Proceedings of the ACM SIGCOMM 2025 Conference*, pp. 917–931, 2025.

[23] Y. Chen, Y. Yang, J. Hu, Y. Wu, and J. Huang, "A game-theoretical approach for distributed computation offloading in leo satellite-terrestrial edge computing systems," *IEEE Transactions on Mobile Computing*, vol. 24, no. 5, pp. 4389–4402, 2025.

[24] J. Huang, R. Xing, X. Ma, A. Zhou, and S. Wang, "Profit-aware task allocation in satellite computing," in *2024 IEEE International Conference on Web Services (ICWS)*, pp. 696–706, IEEE, 2024.

[25] I. Čilić, V. Jukanović, I. P. Žarko, P. Frangoudis, and S. Dustdar, "Qedge-proxy: Qos-aware load balancing for iot services in the computing continuum," in *2024 IEEE International Conference on Edge Computing and Communications (EDGE)*, pp. 67–73, IEEE, 2024.

[26] C. Yin, P. Dong, X. Du, Y. Zhang, and H. Zhang, "Adsa: A multi-path transmission scheduling algorithm based on deep reinforcement learning in vehicle networks," in *ICC 2022-IEEE International Conference on Communications*, pp. 5058–5063, IEEE, 2022.

[27] X. Li, H. Wang, S. Yi, and X. Yao, "Receiving-capacity-constrained rapid and fair disaster backup for multiple datacenters in sdn," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2017.

[28] J. S. Deepak Vasisht and R. Chandra, "L2d2: low latency distributed downlink for leo satellites," *In Proceedings of the 2021 ACM SIGCOMM 2021 Conference (SIGCOMM '21)*, p. 151–164, Aug. 2021.

[29] W. Liu, Q. Wu, Z. Lai, H. Li, Y. Li, and J. Liu, "Enabling ubiquitous and efficient data delivery by leo satellites and ground station networks," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pp. 687–692, IEEE, 2022.

[30] K. Devaraj, M. Ligon, E. Blossom, J. Breu, B. Klofas, K. Colton, and R. Kingsbury, "Planet high speed radio: Crossing gbps from a 3u cubesat," 2019.

[31] X. Zhang, Y. Wang, G. Geng, and J. Yu, "Delay-optimized multicast tree packing in software-defined networks," *IEEE Transactions on Services Computing*, vol. 16, no. 1, pp. 261–275, 2021.

[32] Z. Lai, H. Li, Q. Zhang, Q. Wu, and J. Wu, "Cooperatively constructing cost-effective content distribution networks upon emerging low earth orbit satellites and clouds," in *2021 IEEE 29th International Conference on Network Protocols (ICNP)*, pp. 1–12, IEEE, 2021.

[33] C. Chi, Y. Zhang, Q. Sun, and S. Wang, "Sabm: Adaptive backup mechanism for satellite services," in *Proceedings of the 1st ACM MobiCom Workshop on Satellite Networking and Computing*, pp. 1–6, 2023.

[34] L. Ma, W. Su, B. Wu, B. Yang, and X. Jiang, "Joint emergency data and service evacuation in cloud data centers against early warning disasters," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1306–1320, 2022.

[35] X. Li, H. Wang, S. Yi, and L. Zhai, "Cost-efficient disaster backup for multiple data centers using capacity-constrained multicast," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 17, p. e5266, 2019.

[36] Y. Wang, L. Zhang, P. Yu, K. Chen, X. Qiu, L. Meng, M. Kadoch, and M. Cheriet, "Reliability-oriented and resource-efficient service function chain construction and backup," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 240–257, 2021.

[37] Y. Zhang, Q. Wu, Z. Lai, and H. Li, "Enabling low-latency-capable satellite-ground topology for emerging leo satellite networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1329–1338, IEEE, 2022.

[38] X. Gao, J. Wang, X. Huang, Q. Leng, Z. Shao, and Y. Yang, "Energy-constrained online scheduling for satellite-terrestrial integrated networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 4, pp. 2163–2176, 2021.

[39] F. Habibi, T. Lorido-Botran, A. Showail, D. C. Sturman, and F. Nawab, "Msf-model: Queuing-based analysis and prediction of metastable failures in replicated storage systems," in *2024 43rd International Symposium on Reliable Distributed Systems (SRDS)*, pp. 12–22, IEEE, 2024.

[40] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 2076–2085, 2020.

[41] O. Kolosov, M. F. Aktaş, E. Soljanin, and G. Yadgar, "Theory vs. practice in modeling edge storage systems," in *2023 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pp. 313–322, IEEE, 2023.

[42] AGI, "Systems tool kit (stk)." https://www.agi.com/products/stk, 2021.

[43] Starlink. https://www.starlink.com/, 2025.

[44] OneWeb. https://oneweb.net/, 2025.

[45] D. Bhattacherjee and A. Singla, "Network topology design at 27,000 km/hour," *In Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies (CoNEXT '19)*, p. 341–354, Dec. 2019.

[46] high-resolution images from Gaofen-1. https://www.gscloud.cn/sources/accessdata/428?pid=471, 2025.

[47] S.-. radar imagery. https://www.gscloud.cn/sources/accessdata/448?pid=446, 2025.

[48] "Landsat optical imagery." https://www.gscloud.cn/sources/accessdata/241?pid=263, 2025.