Robust Downlink Scheduling for Low Latency Data Delivery in LEO Satellite Networks

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

[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*—Low Earth Orbit satellites generate terabytes of Earth observation data daily, which must be downlinked within limited contact windows. However, orbital geometry and sparse ground stations limit satellite-ground contact to short, intermittent windows, making large-volume data delivery challenging. Existing methods treat predicted link rates as deterministic inputs for downlink scheduling, ignoring skewed heavy-tailed errors that cause backlog accumulation and data loss under capacity overestimation. To address these challenges, we first propose a conditional error distribution model for downlink scheduling that characterizes prediction uncertainty and derives risk-adjusted link capacities with probabilistic guarantees. Based on this, we propose a rolling-horizon robust scheduling algorithm that approximates the stochastic dynamic program with finite-window earliest-deadline-first greedy optimization, improving completion rate, latency, and throughput without prior knowledge of link rates. Experiments show that our algorithm achieves a 94.3% task completion rate, outperforming the state-of-the-art by 22.1%, with 73.4% higher throughput, 57.1% lower latency, and 75.3% less data loss.**

***Index Terms*—LEO satellite networks, Integrated satellite and terrestrial networks, Downlink scheduling, Stochastic optimization**

I. Introduction

Low Earth orbit (LEO) satellite networks have become critical infrastructure for global Earth observation (EO), with over 45% of LEO satellites collecting tens of terabytes of remote sensing data daily [1], [2], which must be downlinked to ground-based EO data centers before distribution to over 4.53 billion users worldwide [3], as shown in Fig. 1. Many downstream applications impose strict timeliness requirements, where data delivered beyond task deadlines in scenarios such as disaster response, extreme weather forecasting, and maritime safety monitoring loses all operational value [4], [5]. However, LEO satellite-ground communication is intermittent, with contact windows severely limited by orbital geometry and ground station deployment, making it challenging to transmit large volumes of data within time constraints [6]. How to rationally schedule downlink priority and bandwidth allocation within these short contact windows is therefore a core problem.

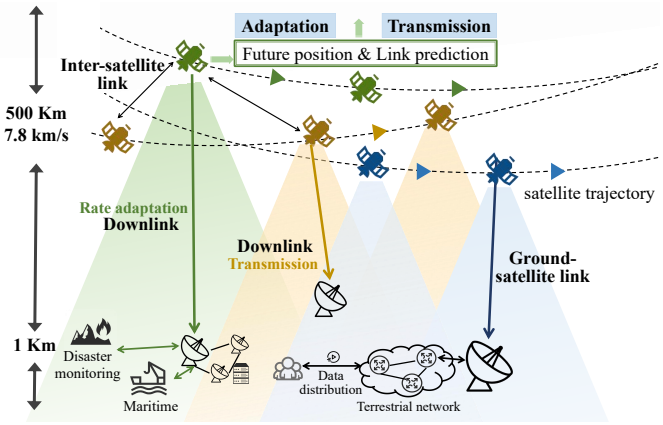

Fig. 1: Data delivery process in typical EO systems.

To alleviate downlink bottlenecks, existing methods mainly follow two approaches. The first leverages on-board edge computing (OEC) to preprocess and compress data before transmission, yet increasing task volume and data demands leave the bottleneck unresolved [2], [7]. The second approach performs downlink scheduling before communication windows using link prediction models based on elevation angles, historical signal-to-noise ratio (SNR), and weather forecasts [8]. However, elevation variations and rain attenuation [9], [10] cause prediction errors, with measured rates falling below predictions in over 40% of windows. When scheduling relies on predicted rates, insufficient

actual throughput causes onboard data accumulation, increasing latency and violating task deadlines. Mismatch between predicted and actual link rates is the core challenge for downlink scheduling in LEO constellations.

Accordingly, we conduct an in-depth analysis of prediction errors and identify two overlooked properties. First, the error distribution exhibits skewness and heavy tails. Our measurements show that prediction errors remain near zero in most windows but exhibit heavy tails under low elevation angles, rain attenuation, and multipath interference, dominating data loss and latency. Second, adjacent window prediction errors exhibit significant temporal correlation, driven by the continuity of orbital geometry and the persistence of weather conditions. Existing scheduling methods treat predicted link rates as exact quantities, implicitly assuming zero prediction error. Under capacity overestimation, insufficient actual throughput leaves data undelivered, causing either direct loss or backlog in subsequent windows. Consecutive overestimations from temporal correlation then let the backlog grow, causing multiple tasks to miss deadlines and triggering large-scale delivery failures.

To mitigate scheduling errors caused by link rate prediction uncertainty, we propose a rolling horizon robust scheduling algorithm (RHRS) that integrate link prediction uncertainty modeling with rolling downlink scheduling. Specifically, RHRS characterizes temporal variations of link rate prediction errors through a conditional error distribution model and derives risk-adjusted capacities for future windows. Based on these, RHRS transforms the uncertainty-aware stochastic scheduling problem into a finite window earliest deadline first (EDF) optimization problem to determine future data transmission plans. Then, only the current-window decision is executed, and prediction errors are updated from actual outcomes to adapt future downlink capacity and task allocation. Extensive experiments demonstrate that RHRS achieves an average task completion rate of 94.3%, improving by 22.1% over the state-of-the-art baseline, with 73.4% higher throughput, 57.1% lower average latency, and 75.3% reduction in data loss rate.

In summary, the main contributions of our paper are:

- We identify two overlooked properties of link rate prediction errors, heavy-tailed skewness and inter-window temporal correlation, as the primary source of downlink scheduling uncertainty in satellite network.
- We propose a rolling horizon robust scheduling algorithm that models conditional error distributions, introduces risk-adjusted capacity, and decomposes the problem into tractable finite-window EDF subproblems.
- Extensive experiments show that RHRS outperforms state-of-the-art baselines in task completion rate, throughput, completion delay, and data loss rate.

II. RELATED WORK

Downlink bottlenecks in emerging LEO constellations have been widely studied [7], [9], [11]. We review prior work on downlink scheduling and selective downlinking, and analyze their limitations.

A. Downlink scheduling

Existing downlink scheduling methods ignore time-varying link quality [5], [12]–[16]. AEROPATH [14] jointly optimizes satellite selection and inter-satellite routing to maximize throughput with low latency, yet assumes precise downlink capacity is known, disregarding fading and weather effects. OrbitCast [15] leverages constellations for low-latency data distribution via location-aware routing, but relies on exact capacity predictions without modeling prediction errors. Tang et al. [16] propose multi-path cooperation to improve reliability, yet still lack capacity uncertainty modeling at the scheduling level. In summary, existing scheduling studies treat downlink capacity as a deterministic input and fail to adequately capture its time-varying nature.

B. Opportunistic communication and selective downlink

Opportunistic communication and selective downlink are key paradigms for coping with intermittent LEO connectivity. Denby et al. [7] proposed offloading computation to satellites to reduce downlink load, e.g., by pre-filtering building imagery. However, this conflicts with the Earth observation business model, where satellites deliver raw data to users. Without knowledge of the end application, onboard pre-filtering risks discarding user-critical information. Vasisht et al. [8] introduced L2D2, a distributed hybrid ground station architecture using commodity hardware to achieve low-latency and robust downlink. L2D2 improves link quality prediction via a data-driven model accounting for multipath and occlusion [9]. Nevertheless, it ignores downlink prediction uncertainty and absorbs errors with a fixed buffer, which cannot handle bias across consecutive windows or capture how prediction error affects capacity.

C. Summary

As summarized in Table 1, existing methods achieve efficient transmission with low latency and high throughput, yet their assumptions deviate from reality.

TABLE I: Related works

	Downlink scheduling	Capacity uncertainty	Multi-window cumulative	Safety margin
[14]	✓	×	×	×
[15]	✓	×	×	×
[7]	✓	×	×	×
[8]	✓	×	×	✓
[16]	✓	×	×	×
Ours	✓	✓	✓	✓

L2D2 addresses both downlink scheduling and fixed safety margin, but it relies solely on link prediction without modeling the uncertainty and the multi-window cumulative effects of prediction errors on scheduling.

III. System Model and Problem Formulation

A. Network model

Let the constellation consist of M LEO satellites, denoted by the set $\mathscr{S} = \{s_1, s_2, \ldots, s_M\}$. The set of ground stations is denoted by $\mathscr{G} = \{g_1, g_2, \ldots, g_N\}$, which represent distributed ground station nodes. Each satellite is equipped with an onboard buffer of finite capacity, with a maximum size denoted by $B^{\max}$, used for temporary storage of downlink datasets. The buffer state at time t is denoted by $B(t) \in [0, B^{\max}]$.

B. Intermittent communication model

For each satellite–ground station pair, contact windows form a non-overlapping sequence. The i-th window is defined as $w_i = (t_i^{\text{start}}, t_i^{\text{end}}, s, g)$, with duration $L_i = t_i^{\text{end}} - t_i^{\text{start}}$. The link rate within a window varies dynamically with elevation angle $\theta(t)$ and real time channel conditions, denoted as $r_i(t)$. The actual capacity of window i is,

$$C_i = \int_{t_i^{\text{start}}}^{t_i^{\text{end}}} r_i(t)\, dt \approx \bar{r}_i \times L_i, \tag{1}$$

and the predicted capacity, derived from elevation angle, weather, and historical SNR, is $\hat{C}_i = \hat{r}_i \times L_i$. The deviation between $\hat{C}_i$ and C_i constitutes the root cause of scheduling uncertainty.

C. Link rate prediction error model

We define the prediction error ε_i as the relative deviation between the actual rate and the predicted rate, $r_i = \hat{r}_i \cdot (1 - \varepsilon_i)$, $\varepsilon_i > 0$ indicates overestimation that actual rate lower than predicted, $\varepsilon_i < 0$ indicates underestimation that actual rate higher than predicted. ε_i is not symmetric, independent, and identically distributed noise, but it exhibits two characteristics.

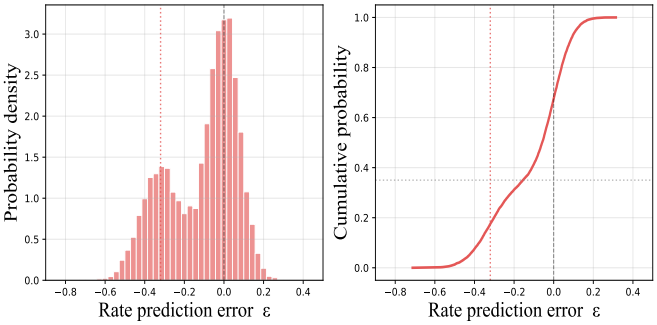

Fig. 2: Structural characteristics of link rate prediction.

Observation 1. Skewed heavy-tailed errors. Based on real-world satellite data collected from HEDE Aerospace, spanning 2023.01 to 2025.01 (63,072,000 records covering channel quality, satellite position), the distribution of ε_i exhibits a bimodal, skewed structure, with the main peak concentrated near $\varepsilon_i \approx 0$, indicating that the link quality of most windows matches the prediction. The heavy tail on the overestimation side ($\varepsilon_i > 0$) corresponds to low-elevation windows, precipitation attenuation (2–3 dB for X/Ka-band), and multipath obstruction. This tail risk, not Gaussian noise, dominates deadline violations, and the mean $\mathbb{E}[\varepsilon_i]$ fails to represent typical window behavior.

Observation 2. Temporal correlation of errors. Link rate errors of adjacent windows over the same ground station exhibit significant positive correlation, due to orbital geometry continuity (similar elevation profiles, low-elevation windows in clusters) and weather persistence (hourly-scale changes slower than window intervals). No such correlation exists for different ground stations. For each ground station g, the error sequence follows an autoregressive process,

$$\varepsilon_{g,j} = \mu_{\varepsilon,g} + \rho_g(\varepsilon_{g,j-1} - \mu_{\varepsilon,g}) + \eta_{g,j}. \tag{2}$$

where $\rho_g > 0$ captures the persistence of overestimation across consecutive windows. Consecutive overestimation is far more likely than under independence, amplifying systematic deadline violations.

D. Task-level queue and buffer model

Satellite s generates EO tasks $\mathscr{D}_s$, each described by a triplet $(t_d, size_d, \tau_d)$ denoting generation time, data size, and deadline. The on-board buffer maintains a task queue Q_i of incomplete tasks before window w_i, with scalar backlog $B_i = \sum_{d \in Q_i} \text{remain}_d$. At window w_i, newly

generated data D_i is merged into the queue, and the transmission allocation x_i must satisfy

$$0 \le x_i \le \min\big(C_i,\, B_i + D_i\big). \tag{3}$$

The actual transmitted bytes are $\mathbb{T}_i = \min(x_i, C_i) = \min(x_i, \hat{C}_i(1-\varepsilon_i))$. The backlog evolves as

$$B_{i+1} = \min\big(B^{\max},\, \max(0,\, B_i + D_i - \mathbb{T}_i)\big), \tag{4}$$

where $\min(\cdot, B^{\max})$ ensures that the buffer does not exceed the on-board storage limit, with any excess treated as data loss, and $\max(0,\cdot)$ guarantees non-negativity.

E. Multi-window error accumulation model

When $x_i > C_i$ (i.e., the scheduler overestimates the capacity), we have $\mathbb{T}_i = C_i < x_i$, and the difference $x_i - C_i$ remains as backlog carried over to B_{i+1}, exerting pressure on subsequent windows. This recursive relationship reveals rollback accumulation. B_i does not reset automatically. When the scheduler overestimates a window's capacity ($\varepsilon_i > 0$), the undelivered data $x_i - \mathbb{T}_i > 0$ is carried over as backlog to the next window. If the scheduler remains unaware of this backlog when planning the next window (since it relies solely on the predicted value $\hat{C}_{i+1}$), the backlog continues to accumulate. When overestimation errors occur consecutively (Observation 2), the backlog grows beyond the compensation capacity of subsequent windows, causing tasks to miss deadlines. Furthermore, once the backlog crosses a critical threshold, it causes multiple tasks to miss deadlines simultaneously, rather than deteriorating gradually on a per-task basis.

F. Problem formulation

We formally cast the scheduling problem under rate prediction uncertainty. Let $B_i \in [0, B^{\max}]$ denote the buffer backlog (state) and $x_i \ge 0$ the transmission allocation (decision) at window w_i. The true capacity is $C_i = \hat{C}_i \cdot (1-\varepsilon_i)$, where $\hat{C}_i$ is the predicted capacity and $\varepsilon_i \sim F_\varepsilon$ is the prediction error with a skewed, temporally correlated distribution as characterized in §III-C.

Therefore, the scheduler aims to find an allocation strategy $\{x_i\}$ that simultaneously minimizes average delay, maximizes completion rate, and maximizes throughput under the link rate prediction error distribution F_ε. We formulate the optimization objective in the form of a normalized weighted sum,

$$\begin{aligned}
\min_{\{x_i\}} \quad & \mathbb{E}_{F_\varepsilon}\left[w_1 \cdot \frac{\overline{\ell_d}}{\ell_{\text{ref}}} - w_2 \cdot \text{CR} - w_3 \cdot \frac{\text{TP}}{\text{TP}_{\max}} \right] \\
& \ell_d = t_d^{\text{finish}} - t_d,\ \forall d \\
& \text{CR} = \frac{1}{|\mathscr{D}_s|} \sum_{d \in \mathscr{D}_s} \mathbf{1}\big[t_d^{\text{finish}} \le \tau_d\big] \\
& \text{TP} = \frac{\sum_{i=1}^{K} \mathbb{T}_i}{\mathbb{T}_{\text{total}}} \\
\text{s.t.} \quad & C1:\ 0 \le x_i \le \min(C_i, B_i + D_i) \\
& C2:\ B_{i+1} = \min\Big(B^{\max}, \\
& \qquad \max\big(0, B_i + D_i - \min(x_i, C_i)\big)\Big) \\
& C3:\ 0 \le B_i \le B^{\max}, \quad \forall i
\end{aligned} \tag{5}$$

where $\mathbb{T}_{\text{total}}$ is the total duration covered by all windows. ℓ_{ref} is the median of task deadlines, and $\text{TP}_{\max}$ is the theoretical maximum throughput, defined as the sum of true capacities of all windows divided by the total duration. w_1, w_2, and w_3 are weight coefficients, set equal by default. For each window w_i ($i = 1, 2, \ldots, K$), the following constraints must hold. C1 is data availability constraint. The allocated transmission volume cannot exceed predicted capacity. C2 refers to buffer dynamics. Backlog in each window equals remaining data from previous window plus newly arrived data minus the actual transmitted data. Backlog follows discrete-time queueing recurrence from §III-E with $B_1 = 0$, coupling scheduling decisions x_i, channel capacity C_i, and new arrivals D_i. C3 is the physical limit. The backlog must not exceed the onboard memory capacity.

IV. Algorithm Design

To solve problem Eq. 5, three challenges must be addressed. (i) Constraint $x_i \le C_i$ cannot be enforced directly, as C_i is unknown at scheduling time. (ii) The objective is in expectation form and nonlinear, making an analytical solution infeasible. (iii) The state is coupled across windows since B_{i+1} depends on ε_i, preventing per-window independent optimization. We transform the stochastic problem into a tractable deterministic equivalent and combine it with rolling replanning.

A. Constraint transformation

Both constraints C1 and C2 involve the unknowable capacity C_i, and the objective contains expectations over ε_i, rendering a direct solution intractable. We therefore

relax the hard constraint into a chance constraint, replacing the exact capacity uncertainty with a probabilistic guarantee that capacity overflow occurs with probability at most δ. In constraint C1, $B_i + D_i$ is data availability known at scheduling time, so it can be retained directly. The truly unknowable is C_i. We replace the corresponding sub-constraint $x_i \le C_i$ with a chance constraint,

$$P\big(x_i \le \hat{C}_i(1-\varepsilon_i)\big) \ge 1-\delta. \tag{6}$$

Solving this chance constraint is equivalent to:

$$x_i \le \hat{C}_i \cdot \big(1 - F_{\varepsilon_i}^{-1}(1-\delta)\big) := \hat{C}_i^{\delta}, \tag{7}$$

where $F_{\varepsilon_i}^{-1}(1-\delta)$ is the $(1-\delta)$-quantile of the ε_i distribution. Accordingly, C1 as a whole is transformed into a deterministic constraint,

$$C1': \quad 0 \le x_i \le \min\big(\hat{C}_i^{\delta},\, B_i + D_i\big) \tag{8}$$

The setting of δ reflects the asymmetric cost of overestimation versus underestimation (Observation 1), with δ binned by window risk level $\delta = \{\delta_{\text{tight}}, \delta_{\text{loose}}\}$. δ_{tight} is high risk windows with low elevation angle, precipitation, and high historical error. And δ_{loose} is a low-risk window with a high elevation angle and clear weather. After this, the unknowable capacity constraint is replaced by a deterministic conservative capacity upper bound $\hat{C}_i^{\delta}$, and the problem becomes a solvable deterministic optimization. Next, we address the determinization of the state transition. The actual transmission amount $\mathbb{T}_i = \min(x_i, C_i)$ in C2 similarly contains the random C_i. Since C1 guarantees $x_i \le \hat{C}_i^{\delta}$, and $\hat{C}_i^{\delta}$ is a conservative value set at confidence level $1-\delta$, we have $x_i \le \hat{C}_i^{\delta} \le C_i$ with probability at least $1-\delta$, so $\mathbb{T}_i \approx x_i$ holds at the same confidence level. Accordingly, C2 is simplified to the deterministic recurrence,

$$C2': B_{i+1} = \max\big(0,\, B_i + D_i - \min\big(\hat{C}_i^{\delta}, B_i + D_i\big)\big). \tag{9}$$

Unlike the physical buffer model, we no longer truncate and discard excess data at $B^{\max}$. Instead, overflow is treated as a penalty in the objective. This avoids corrupting task completion determination, as physical data loss would invalidate the integrity check of each task.

After the above substitutions, B_i becomes deterministic given x_i, independent of ε_i. All three performance metrics are computed from deterministic queue simulation, so the expectation operator is eliminated, and the problem reduces to a deterministic optimization.

$$\min_{x_i} \quad w_1 \cdot \frac{\overline{\ell_d}}{\ell_{\text{ref}}} - w_2 \cdot \text{CR} - w_3 \cdot \frac{\text{TP}}{\text{TP}_{\max}} \tag{10}$$

$$\text{s.t.} \quad C1',\ C2', \quad 0 \le B_i \le B^{\max}\ \forall i, \quad B_1 = 0 \tag{11}$$

B. Link rate prediction error distribution estimation

Section IV-A introduces the capacity reduction formula $\hat{C}_i^{\delta} = \hat{C}_i \cdot \big(1 - F_{\varepsilon_i}^{-1}(1-\delta)\big)$, where the key term is the quantile of ε_i. Since the temporal correlation of errors is station-specific rather than global, we perform estimation per ground station. For each link of ground station g, the error sequence follows an autoregressive process in Eq. (2). We estimate the $\mu_{\varepsilon,g}$ and ρ_g from the historical error sequence using ordinary least squares. We model $\eta_{g,j}$ as a skewed-t distribution $\eta_{g,j} \sim \sigma_{\eta,g} \cdot \text{ST}(\nu_g, \lambda_g)$, where $\sigma_{\eta,g}$ is the sample standard deviation of the ordinary least squares residuals $\hat{\eta}_{g,j}$, ν_g controls tail heaviness and λ_g controls asymmetry, capturing the heavy-tailed skewed behavior. The shape parameters (ν_g, λ_g) are fitted by maximum likelihood estimation on $\hat{\eta}_{g,j}$. Given the observed error $\varepsilon_{g,\text{prev}}^{\text{obs}}$, the conditional $(1-\delta)$-quantile of $\varepsilon_{g,j}$ is:

$$F^{-1}_{\varepsilon|\varepsilon^{\text{obs}}_{g,\text{prev}}}(1-\delta) = \hat{\mu}_g + \hat{\sigma}_{\eta,g} \cdot F^{-1}_{\text{ST}(\hat{\nu}_g,\hat{\lambda}_g)}(1-\delta), \tag{12}$$

where $\hat{\mu}_g = \hat{\mu}_{\varepsilon,g} + \hat{\rho}_g(\varepsilon^{\text{obs}}_{g,\text{prev}} - \hat{\mu}_{\varepsilon,g})$ is the AR(1) conditional mean and F^{-1}_{ST} is the closed-form quantile function of the fitted skewed distribution. The risk-adjusted capacity is $\hat{C}_i^{\delta} = \hat{C}_i \cdot \Big(1 - F^{-1}_{\varepsilon|\varepsilon^{\text{obs}}_{g,\text{prev}}}(1-\delta)\Big)$, making $\hat{C}_i^{\delta}$ computable from historical observations and the current error state. During online operation, each window's observed error $\varepsilon_i^{\text{obs}} = 1 - C_i^{\text{obs}}/\hat{C}_i$ updates the station's conditional state $\varepsilon_{g,\text{prev}}^{\text{obs}}$ for future visits.

C. Rolling horizon robust scheduling algorithm

After constraint transformation, each satellite faces a finite-horizon deterministic task scheduling. By the single-link access constraint, the scheduling of M satellites is independent and can be solved separately. Algorithm 1 outlines the overall rolling horizon scheduling framework. At each window, it updates link prediction error state using telemetry from the most recent pass, computes capacities $\hat{C}_k^{\delta}$ for the next H windows via $F^{-1}(1-\delta)$, and invokes Algorithm 2 to solve a deterministic H-window subproblem. Only the first window's decision is executed, while remaining windows serve as a lookahead buffer. After execution, actual transmission is capped by realized capacity C_i, and the queue is updated with any unfinished data carried forward.

RHRS innovates in three aspects. First, it maintains per-station conditional error distributions that exploit temporal correlation across windows rather than treating errors independently. Second, risk-adjusted capacity $\hat{C}_i^{\delta}$ accounts for heavy-tailed skewness, bounding overflow

Algorithm 1 Rolling horizon scheduling algorithm

Input: $\mathscr{S}$, G, task sets $\mathscr{D}_s$, horizon length H, risk tolerance δ, error distribution parameters $\{\mu_{\varepsilon,g}, \sigma_{\varepsilon,g}, \rho_g\}$
Output: Transmission schedule and completion times

for each satellite $s \in \{1,\dots,M\}$ in parallel **do**
  Initialize $Q_{s,1} \leftarrow \emptyset$, $B_{s,1} \leftarrow 0$
  for $i = 1,2,\dots,K_s$ **do**
    $g_i \leftarrow$ ground station for window $w_{s,i}$
    if g_i has been visited before **then**
      $\varepsilon^{\text{obs}}_{g_i,\text{prev}} \leftarrow 1 - C^{\text{obs}}_{g_i,\text{prev}} / \hat{C}_{g_i,\text{prev}}$
    end if
    for $k = i$ **to** $\min(i+H-1, K_s)$ **do**
      $\hat{C}^{\delta}_k \leftarrow \hat{C}_k \cdot (1 - q^{\delta}_k)$ where $q^{\delta}_k = F^{-1}_{\varepsilon|o}(1-\delta_k)$
    end for
    Add arriving tasks $d \in \mathscr{D}_s$ with $t_d \le w_{s,i}$ start to $Q_{s,i}$
    $(x^*_i, \dots) \leftarrow \text{TQA}(Q_{s,i}, B_{s,i}, \{\hat{C}^{\delta}_k\}, H)$
    Transmission: $x^{\text{act}}_i \leftarrow \min(x^*_i, C_{g_i}, B_{s,i} + D_i)$
    Update $Q_{s,i+1}$, record t^{finish}_d for completed tasks
  end for
end for

Algorithm 2 EDF-based subproblem greedy algorithm

Input: $Q_{s,i}$, $\hat{C}^{\delta}_k$, H
Output: Optimal allocation $\{x^*_k\}$

Sort tasks by τ_d ascending
for each task d in sorted order **do**
  for $k = i$ **to** $\min(i+H-1, K_s)$ **do**
    if $w_{s,k}.\text{start} \ge t_d$ **then**
      Allocate $\min(\text{remain}_d, \hat{C}^{\delta,\text{rem}}_k)$ to window k
      if task d complete **then**
        Record t^{finish}_d; **break**
      end if
    end if
  end for
end for

probability to at most δ. Third, the rolling horizon with lookahead couples short-term execution with future planning, bridging stochastic uncertainty and deterministic optimization without sacrificing statistical fidelity.

D. Subproblem solving: task-level queue allocation

The subproblem solved by task-level queue allocation (TQA) at w_i is defined as follows. Given the queue Q (containing backlogged tasks and newly arrived ones up to w_i.start), the set of risk-adjusted capacities $\{\hat{C}^{\delta}_k\}$ for the next H windows $k = i,\dots,i+H-1$, and the deadlines τ_d of all tasks, the solver determines the allocation x^*_k for each window k within the horizon. The allocation follows an earliest deadline first greedy policy. All tasks are first sorted in ascending order of τ_d, with ties broken by remaining bytes in ascending order. For each task, the algorithm greedily assigns data to the earliest feasible window after t_d with remaining capacity, marking the task complete once fully allocated or pending otherwise. Output is per-window allocation vector $\{x^*_k\}$, from which only x^*_i is executed.

V. Performance Evaluation

A. Experimental setting

We use SpaceX Starlink Phase I (1584 satellites, 72 orbits, 550 km) as the primary LEO constellation, with remote sensing satellite orbits [15] and 173 ground stations [17]. EO traffic is generated from NASA LANCE datasets [18], [19] for emergency response and disaster management. Task sizes range from 50 to 500 MB, with deadlines split into urgent (30%, 1–3h), normal (50%, 4–8h), and loose (20%, 12–24h). For comparison, we implement three benchmark algorithms. The distributed ground station (DGS) [8] employs a hybrid ground station model integrated with a link prediction mechanism to reduce downlink latency. OrbitCast [15] leverages inter-satellite links to forward data to other satellites that have upcoming downlink windows when predicted transmission rate is insufficient to complete data delivery within current window. Oracle uses actual capacity C_i with the same EDF-based greedy allocation, serving as the upper bound for completion rate and throughput and the lower bound for delay. We evaluate three metrics: completion rate (CR), system throughput (TP), and latency, as defined in Eq. (5).

Link rate estimation uses a link budget model. The predicted rate $\hat{r}_i$ is derived from ground station parameters and window elevation angle, which determine slant range d_i, yielding SNR via free-space path loss and then $\hat{r}_i$ via the Shannon formula. The actual rate superimposes structured errors onto $\hat{r}_i$: a base prediction error following a skewed Beta distribution (mean 0.06, 90th percentile 0.41), additional attenuation from Uniform(0.1, 0.3) at low elevation angles ($<15°$), and weather attenuation from Uniform(0.05, 0.15) triggered with probability 0.15. All parameters are calibrated from the L2D2 measurement report [8].

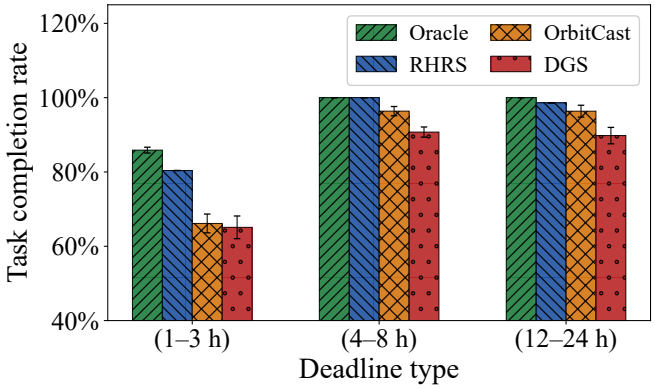

Fig. 3: Task completion rate under different deadline tiers.

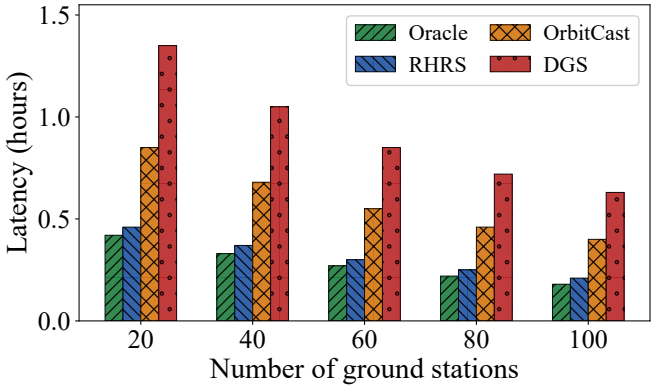

Fig. 5: Average completion delay under varying ground station densities.

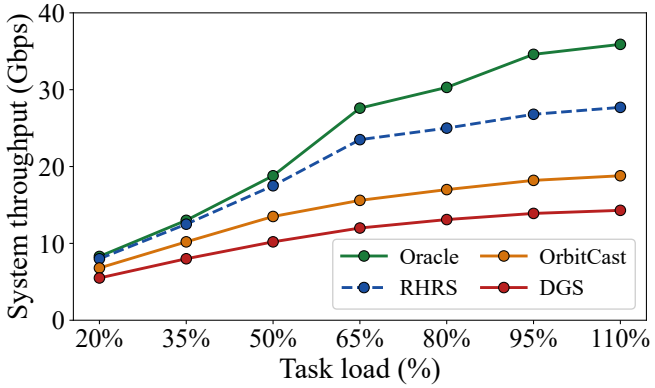

Fig. 4: System throughput under varying task loads.

B. Results and analysis

Analysis of task completion rate under different deadline tiers. Fig. 3 shows that RHRS consistently outperforms the baselines across all deadline tiers and closely tracks the Oracle. Under tight deadlines, where compensation windows are scarce, the proposed coordinated optimization mechanism, which combines outer-tier risk-adjusted capacity $\hat{C}_k^{\delta}$ to cap allocation and inner-tier EDF ordering with H-window compensation, enables data delivery within the shortest time. In addition, the probabilistic safety margin ensures 0 data loss.

System throughput analysis. This experiment sweeps task load from 20% to 110%. Under low-to-moderate loads, all algorithms differ by less than 1% as predicted capacity is not yet strained. Beyond 65%, prediction error skewness triggers capacity truncation: DGS and OrbitCast lack risk-adjusted capacity $\hat{C}_k^{\delta}$ and suffer exponentially increasing truncation events and throughput collapse, while RHRS keeps allocations below the quantile threshold, making truncation negligible. In the 65–80% range, RHRS delivers 95.8% more throughput than DGS and 47–51% more than OrbitCast. Under high load, EDF-ordered allocation directs available capacity to the tightest-deadline tasks, preventing persistent backlog accumulation.

Analysis of completion delay under varying ground station densities. Fig. 5 shows that RHRS achieves significantly lower delay than the baselines. This advantage stems from the error-state update in Algorithm 1, which identifies overestimation windows in advance, combined with EDF ordering and H-window compensation that prevent queuing delays caused by over-allocation in DGS and OrbitCast. As ground station density increases, visible windows become more frequent and all algorithms exhibit lower delays. However, RHRS prioritizes each new window capacity for the most urgent tasks, achieving the largest delay reduction of 54.3% as the number of ground stations increase.

Data loss rate under varying task loads. Fig. 6 shows that RHRS maintains data loss rate below 0.05 across all loads, while baselines exceed 0.10 beyond 30% load. Baselines allocate based on predicted capacity, causing backlog to accumulate under consecutive overestimation and eventually overflow the buffer. RHRS suppresses over-allocation at the source via risk-adjusted capacity $\hat{C}_k^{\delta}$, while EDF ordering prioritizes urgent tasks, keeping backlog below the buffer limit across all load conditions.

VI. CONCLUSIONS AND FUTURE WORK

We propose a rolling horizon robust scheduling algorithm that addresses link rate prediction uncertainty in LEO downlink scheduling. By modeling conditional

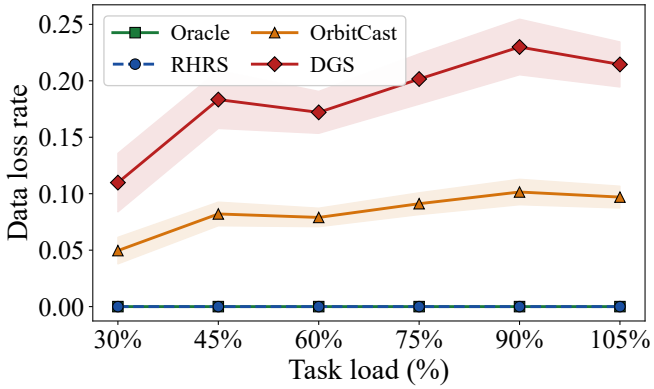

Fig. 6: Data loss rate under varying task loads.

error distributions and introducing risk-adjusted capacity, the stochastic scheduling problem is decomposed into tractable finite-window subproblems. Experiments show that RHRS improves task completion rate by 22.1% and reduces average delivery latency by 57.1% over baselines, demonstrating robust performance under uncertain link conditions. Future work will focus on jointly training the orbit prediction module and the rolling horizon scheduler, using scheduling feedback to guide prediction toward critical error regions.

Acknowledgment

This work was supported by the Science and Technology Innovation Program of Xiongan New Area under Grant 2024XAGG0025, the National Natural Science Foundation of China under Grants 62302055 and 62425203, and the BUPT Excellent Ph.D. Students Foundation under Grant CX2023130. The first author's research visit as a Visiting PhD student to Melbourne qCLOUDS Lab is supported by the University of Melbourne.

References

[1] C. Yang, Q. Zhang, Q. Sun, S. Ouyang, A. Zhou, S. Wang, and M. Xu, "Fedclr+: Tackling onboard label constraints for accurate federated satellite computing," *IEEE Transactions on Services Computing*, vol. 18, no. 4, pp. 2075–2088, 2025.

[2] X. Yang, J. Feng, L. Liu, Q. Pei, S. Mumtaz, K. Li, and S. Dustdar, "Irs-assisted hyperspectral image processing in satellite edge computing services," *IEEE Transactions on Services Computing*, vol. 18, no. 5, pp. 2946–2959, 2025.

[3] 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.

[4] P. Deng, X. Gong, and X. Que, "A mobility adaptive service placement scheme in satellite edge computing network," in *2022 IEEE 24th Int Conf on High Performance Computing Communications, (HPCC)*, pp. 1430–1435, 2022.

[5] C. Wang, Y. Peng, J. Wu, L. Liu, S. Mumtaz, M. Dong, and M. Guizani, "Deterministic scheduling and reliable routing for smart ocean services in maritime internet of things: A cross-layer approach," *IEEE Transactions on Services Computing*, vol. 17, no. 6, pp. 3387–3399, 2024.

[6] F. Tang, X. Li, L. Chen, J. Liu, M. Gao, Y. Yang, W. Xu, and H. Zhang, "Reinforcement learning based control domain division in leo satellite networks," in *2023 IEEE International Conference on High Performance Computing Communications (HPCC)*, pp. 303–310, 2023.

[7] B. Denby and B. Lucia, "Orbital edge computing: Nanosatellite constellations as a new class of computer system," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 939–954, 2020.

[8] D. Vasisht, J. Shenoy, and R. Chandra, "L2d2: Low latency distributed downlink for leo satellites," in *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pp. 151–164, 2021.

[9] K. Devaraj, M. Ligon, and E. Blossom, "Planet high speed radio: Crossing gbps from a 3u cubesat," in *Proceedings of the Small Satellite Conference*, 2019.

[10] Y. Li, H. Li, L. Liu, W. Zhao, Y. Chen, J. Wu, Q. Wu, J. Liu, Z. Lai, *et al.*, "A networking perspective on starlink's self-driving leo mega-constellation," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pp. 1–16, 2023.

[11] H. Li, Y. Li, Y. Liu, B. Deng, Y. Li, X. Li, and S. Zhao, "Earth observation satellite downlink scheduling with satellite-ground optical communication links," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 61, no. 2, pp. 2281–2294, 2025.

[12] R. A. d. Carvalho, "Optimizing the communication capacity of a ground station network," *Journal of Aerospace Technology and Management*, vol. 11, p. e2319, 2019.

[13] D. Karapetyan, S. M. Minic, K. T. Malladi, and A. P. Punnen, "Satellite downlink scheduling problem: A case study," *Omega*, vol. 53, pp. 115–123, 2015.

[14] 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.

[15] Z. Lai, Q. Wu, H. Li, M. Lv, and J. Wu, "Orbitcast: Exploiting mega-constellations for low-latency earth observation," in *2021 IEEE 29th International Conference on Network Protocols (ICNP)*, pp. 1–12, IEEE, 2021.

[16] F. Tang, H. Zhang, and L. T. Yang, "Multipath cooperative routing with efficient acknowledgement for leo satellite networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 1, pp. 179–192, 2018.

[17] "Satnogs: Open source global network of satellite ground-stations." https://satnogs.org/. Accessed: 2026-07-09.

[18] "Misr near real-time (nrt) data." https://earthdata.nasa.gov/earth-observation-data/near-realtime/download-nrt-data/misr-nrt. Accessed: 2026-07-09.

[19] "Modis near real-time (nrt) data." https://earthdata.nasa.gov/earth-observation-data/near-realtime/download-nrt-data/modis-nrt. Accessed: 2026-07-09.