Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

An Orbital Data Routing Framework for Latency-Aware Edge Processing in CubeSat Networks

Debjit Naskar

debjitnaskar@icloud.com

RCC Institute of Information Technology

Ritabhasa Chowdhury

Institute Of Engineering and Management

Research Article

Keywords:

Posted Date: November 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11640535/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

An Orbital Data Routing Framework for Latency-Aware Edge Processing in CubeSat Networks

Debjit Naskar
Department of Electronics and Communication Engineering
RCC Institute of Information Technology
Email: debjitnaskar@icloud.com

Ritabhasa Chowdhury
Department of Electronics and Communication Engineering
Institute of Engineering and Management
Email: mail2ritabhasa@gmail.com

***Abstract*—Small satellite constellations are evolving from data collectors into distributed edge-computing networks that must meet real-time latency constraints. Conventional Delay-Tolerant Networking (DTN) falls short under rapidly changing orbital geometry, intermittent inter-satellite links (ISLs), and fluctuating compute loads. This paper presents the Latency-Aware Orbital Data Routing Framework (LA-ODRF), which integrates (i) a predictive orbital link-state estimator based on two-line element (TLE) propagation, (ii) a delay-sensitive routing layer with a weighted multi-metric cost function, and (iii) an edge-orchestration engine that balances compute versus forwarding against time budgets. The primary novelty is this unified closed-loop integration of routing, orchestration, and orbital prediction, rather than an isolated modification to a single module. Simulations of low Earth orbit (LEO) constellations with optical and RF ISLs show that LA-ODRF reduces end-to-end latency by up to 37%, increases task offload efficiency by 52%, and improves resilience to ISL disruptions compared to baseline DTN-BP routing. The framework provides a foundation for orbital cloud services and scalable CubeSat processing clusters.**

***Index Terms*—CubeSat, inter-satellite links, edge computing, low-latency routing, LEO constellations, predictive networking**

I. Introduction

Real-time space applications such as intelligence, surveillance, and reconnaissance (ISR), disaster response, and time-sensitive analytics are increasingly supported by CubeSat constellations that combine sensing, processing, and inter-satellite relaying [1]. Latency becomes a primary operational constraint because orbital motion reshapes contact opportunities and path lengths on the order of minutes, while on-board edge workloads introduce queueing and compute delays. Traditional DTN/BP forwarding tolerates disruption but is not optimized for latency-sensitive tasks or compute offloading decisions [2], [3].

The emergence of mega-constellations such as Starlink and OneWeb has demonstrated the feasibility of large-scale LEO networks [4], yet these systems primarily focus on ground-user connectivity rather than in-orbit processing. Scientific and defense applications increasingly require on-board analytics, federated learning, and distributed sensor fusion [5], [6], creating demand for routing protocols that jointly optimize communication latency and computational resource allocation.

This work introduces the Latency-Aware Orbital Data Routing Framework (LA-ODRF), a unified routing and edge orchestration stack that co-optimizes communication and computation under orbital dynamics. LA-ODRF predicts link availability windows, contact duration, and expected link quality using TLE-driven SGP4 propagation [7], computes routes via a weighted multi-metric cost, and schedules local processing versus offloading according to latency budgets. Unlike existing contact graph routing schemes [8], [9], LA-ODRF explicitly models compute queue depths and processing delays, enabling fine-grained trade-offs between forwarding and local execution.

Representative onboard workloads used throughout the paper, such as image processing and AI inference, are illustrative examples rather than a limitation of the framework; LA-ODRF is intended for any task class that can be profiled by compute cost, data size, and deadline.

Contributions:

- A unified orbital communication-computation control loop that integrates TLE/SGP4 contact prediction, queue-aware routing, compute offloading, and deadline-constrained arbitration at each routing epoch.
- A latency-aware routing layer with a tunable cost function combining propagation delay, queueing, hop availability probability, and compute load, supporting heterogeneous traffic classes.
- An edge-orchestration decision engine that jointly arbitrates among local execution, forwarding, and offloading with budget-aware policies informed by task deadlines.
- A comprehensive simulation study on LEO constellations (6–36 CubeSats, optical and RF ISLs) showing substantial latency and offload gains over DTN-BP baseline [2].

What is novel relative to prior work:

- Prior DTN/CGR approaches optimize forwarding over predicted contacts, but do not jointly optimize routing with on-board compute queue contention and task-placement decisions.
- Prior CubeSat edge-computing studies focus on local inference/compression pipelines, but do not provide a

network-layer mechanism to couple deadline-aware routing with execution offloading.
- LA-ODRF contributes a unified control loop where orbital prediction, queue-aware routing, and compute-forwarding arbitration are solved together at each epoch, which is the key technical distinction of this work.

The remainder of this paper is organized as follows: Section II surveys related work in LEO networking, DTN routing, and space-based edge computing. Section III formalizes the problem statement. Section IV describes the LA-ODRF architecture in detail. Section V presents routing and orchestration algorithms. Section VI details simulation methodology. Section VII analyzes results, and Section VIII concludes with future directions.

II. Background and Related Work

A. LEO Constellation Dynamics

LEO CubeSats operating at altitudes between 500–650 km complete an orbit in approximately 95 minutes [10]. Relative motion induces fast-changing geometry, Doppler shifts exceeding 40 kHz at UHF frequencies [11], and intermittent line-of-sight (LOS), directly impacting ISL availability and link budget. The dynamic topology requires routing protocols that adapt to predictable yet rapidly evolving contact opportunities [12].

Walker constellation patterns [13] provide structured coverage but introduce periodic gaps. Recent work on mega-constellations demonstrates that dense deployments improve path diversity and reduce end-to-end latency through multi-hop ISLs [14], [15]. These gaps contribute directly to latency, because data must often be stored and deferred until the next contact opportunity. However, these studies primarily address ground connectivity rather than in-orbit workload distribution, which is the setting addressed by LA-ODRF. For a circular orbit, the orbital period is given by:

$$T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{1}$$

where a is the semi-major axis and μ is Earth's gravitational parameter (3.986×10^{14} m^3/s^2). For a 550 km altitude orbit, $a = 6371 + 550 = 6921$ km, yielding $T \approx 95.8$ minutes and orbital velocity $v = \sqrt{\mu/a} \approx 7.6$ km/s. This rapid motion means that satellites separated by 30° in the same orbital plane can open or close contact windows in under 3 minutes.

Coverage Dynamics: A single LEO satellite at 550 km altitude with a minimum elevation angle of 10° provides ground coverage over a circular footprint with radius $r_c \approx 2300$ km. Contact duration with a fixed ground station ranges from 4–10 minutes depending on pass geometry. For ISL establishment between co-planar satellites with maximum range $d_{\max} = 5000$ km (corresponding to roughly 60° separation), visibility windows last 8–15 minutes before exceeding range or angular constraints.

B. Inter-Satellite Communication

Optical ISLs offer multi-Gbps throughput and low latency but require tight pointing with sub-microradian accuracy [16]. Acquisition and tracking loops introduce setup delays of 10–50 ms [17]. RF ISLs provide robustness at lower capacity (100s of Mbps) and relaxed pointing requirements [18]. Hybrid optical/RF meshes can reduce outage probability but complicate routing due to modality switching and heterogeneous link characteristics [19].

Free-space optical communication (FSOC) systems such as the Laser Communications Relay Demonstration (LCRD) [20] have validated relay architectures, yet integration with adaptive routing remains an open challenge. Recent advances in software-defined radios enable reconfigurable RF topologies [21], but latency-aware scheduling across mixed optical/RF paths is underexplored.

Link Budget Analysis: For an optical ISL at wavelength $\lambda = 1550$ nm over distance $d = 2000$ km, the free-space path loss (FSPL) is:

$$L_{\text{FSPL}} = 20\log_{10}\left(\frac{4\pi d}{\lambda}\right) \approx 285.2 \text{ dB} \tag{2}$$

With transmit power $P_t = 1$ W (30 dBm), telescope aperture diameter $D_t = 10$ cm providing gain $G_t \approx 120$ dB, and receiver aperture $D_r = 10$ cm with $G_r \approx 120$ dB, the received power is:

$$P_r = P_t + G_t + G_r - L_{\text{FSPL}} - L_{\text{atm}} \approx -15.2 \text{ dBm} \tag{3}$$

where $L_{\text{atm}} \approx 0$ dB for vacuum propagation. Assuming a receiver sensitivity of -40 dBm for 1 Gbps OOK modulation, this provides $\sim$ 25 dB link margin, sufficient to tolerate pointing jitter and scintillation.

For UHF RF links at 450 MHz with isotropic antennas ($G_t = G_r = 0$ dB) and $P_t = 2$ W (33 dBm), FSPL over 2000 km is:

$$L_{\text{FSPL}} = 20\log_{10}\left(\frac{4\pi \times 2 \times 10^6}{c/450 \times 10^6}\right) \approx 146.5 \text{ dB} \tag{4}$$

Resulting in $P_r \approx -113.5$ dBm, which requires coding gain and narrowband filtering to achieve acceptable SNR for modest data rates (100s of kbps to low Mbps).

C. Delay/Disruption-Tolerant Networking

DTN/BP [2], [3] and contact-graph routing (CGR) [8] address intermittency via store-carry-forward and predictive contact schedules. In this setting, the contact graph is a directed acyclic graph whose edges represent scheduled communication opportunities while the satellites remain within a usable ISL range, which in our model is about 5000 km, and satisfy pointing constraints. CGR computes earliest-delivery-time paths, but its optimization targets delivery probability and deadline feasibility rather than minimum latency [9]. Variants such as Enhanced CGR [9] and probabilistic CGR [22] incorporate link uncertainty, yet the cited methods do not explicitly model on-board compute contention or queue-aware path costs.

Opportunistic routing schemes [23], [24] exploit unplanned encounters but sacrifice predictability. Epidemic and spray-and-wait forwarding [25], [26] improve delivery ratio at the cost of bandwidth overhead and unbounded latency variance. Our work extends CGR by incorporating compute load and queueing delay for onboard tasks such as image processing, anomaly detection, sensor fusion, and federated learning updates into cost metrics, enabling joint communication-computation optimization.

D. Edge Computing in Space

Recent work explores anomaly detection [27], image compression [28], and federated learning [29] on CubeSats to minimize downlink volume. While these studies demonstrate the value of onboard processing, they also motivate the need for system-level orchestration policies that decide where and when tasks should execute across multiple satellites. Edge orchestration frameworks for terrestrial IoT [30], [31] prioritize tasks by deadline and resource availability, yet orbital mobility and ISL dynamics introduce unique constraints.

Projects such as CloudSat [6] and SpaceCloud [5] propose distributed computing abstractions for satellite constellations, but routing and compute scheduling are treated separately. Recent surveys [32], [33] identify the need for co-designed routing and edge-orchestration layers—a gap LA-ODRF addresses directly.

E. Identified Gaps

Based on the cited literature, existing approaches only partially address the following requirements in combination:

- Time-varying propagation delay from orbital geometry
- Compute queue depths and processing latency
- Heterogeneous ISL modalities (optical vs. RF)
- Real-time task deadlines and offloading trade-offs

LA-ODRF addresses this gap by unifying predictive link estimation, latency-aware multi-metric routing, and deadline-driven compute orchestration in a single framework. The primary novelty is this system-level integration and co-optimization workflow, rather than an isolated new algorithm within only one module.

Table I explicitly positions LA-ODRF against representative classes of prior work.

TABLE I: Positioning of LA-ODRF Relative to Prior Work Classes

Approach class	Orbital pred.	Queue-aware routing	Compute offload	Joint opt.
CGR / Enhanced CGR [8], [9]	Yes	Partial	No	No
Opportunistic DTN [23], [24]	Limited	No	No	No
CubeSat edge studies [27], [28]	No	No	Partial	No
Terrestrial MEC methods [30], [31]	No	Yes	Yes	Partial
LA-ODRF (this work)	**Yes**	**Yes**	**Yes**	**Yes**

III. Problem Statement

We target a constellation represented as a time-varying graph $G(t) = (V, E(t))$ where nodes V correspond to CubeSats and the edge set $E(t)$ evolves with orbital motion [34]. Ground stations are modeled as external destination sinks connected by time-varying satellite-to-ground contact opportunities, and are not part of V in this formulation.

Each application task τ may generate one or more packets p. Routing decisions are made per packet, while compute placement (local execution vs. offload) is made per task and then inherited by all packets associated with that task. Each task τ is characterized by:

- Deadline D: maximum tolerable end-to-end latency
- Compute requirement c: processing time on a reference platform
- Priority class: real-time (RT) or batch (BT), where RT tasks have tighter deadlines and higher scheduling priority than BT tasks

These constraints are meaningful because they define the coupled communication-computation operating regime that LA-ODRF must optimize: links are intermittent and time-varying, while compute and queue resources are limited and deadline-sensitive.

Key system constraints include:

1) **Time-varying propagation delay:** Slant range $d_{ij}(t)$ between satellites i, j changes with relative velocity up to 15 km/s, inducing propagation delays $\tau_{ij}(t) = d_{ij}(t)/c_0$ that vary by milliseconds per second [10].
2) **ISL intermittency:** Optical links require mutual visibility and pointing accuracy; contacts have finite duration (30 s to 10 min) with setup overheads [16].
3) **Queueing and compute delays:** Each satellite maintains transmit queue Q_i^{tx} and compute queue Q_i^{comp}. Queueing delay q_i and processing time compete for limited on-board resources.
4) **Energy and processor limits:** CubeSats have constrained power budgets (5–30 W) and modest processors (ARM Cortex-A series) [1], limiting concurrent task execution.
5) **Compute-forwarding trade-off:** For task τ, node i must decide whether to process locally, offload to neighbor j, or forward toward a ground station, subject to deadline D.

Objective: Minimize expected end-to-end latency $\mathbb{E}[T_{\text{e2e}}]$ for all tasks while maintaining packet delivery ratio (PDR) above threshold $\rho_{\min}$ and respecting energy budget $E_{\max}$. Here, the expectation is taken over stochastic link outages, queue arrivals, and contact-window uncertainty. Formally:

$$\min \mathbb{E}[T_{\text{e2e}}] \quad \text{s.t.} \quad \text{PDR} \geq \rho_{\min}, \sum_i E_i \leq E_{\max} \tag{5}$$

where $T_{\text{e2e}} = \sum_{(i,j)\in\pi}(\tau_{ij} + q_i) + c_\pi$ for path π and total compute time c_π.

The problem is NP-hard in general. A concise argument is that our formulation contains known hard subproblems as special cases: (i) if compute placement is fixed, the model reduces to deadline-constrained multi-commodity routing over capacitated time-expanded links; (ii) if routing is fixed, it reduces to deadline-constrained compute scheduling/offloading under capacity limits; and (iii) our full model couples both sets of binary decisions through Eqs. (7)–(9). Since each restricted case is already computationally hard, their joint optimization in a time-varying graph remains NP-hard [35]. We therefore employ a greedy heuristic framework that leverages orbital predictability to achieve strong empirical performance in practice.

A. Mathematical Formulation

We formulate the joint routing and compute offloading problem as a mixed-integer program. Let x_{ijt}^p be a binary variable indicating whether packet p is forwarded on link (i,j) at time t, and y_{it}^p indicate whether packet p is processed at node i at time t. The optimization becomes:

$$\min \quad \sum_p \sum_t t \cdot z_t^p \tag{6}$$

$$\text{s.t.} \quad \sum_{j:(i,j)\in E(t)} x_{ijt}^p + y_{it}^p \le 1,\ \forall i,t,p \tag{7}$$

$$\sum_p x_{ijt}^p \cdot s_p \le C_{ij}(t),\ \forall (i,j),t \tag{8}$$

$$\sum_p y_{it}^p \cdot c_p \le \mathcal{C}_i,\ \forall i,t \tag{9}$$

$$\sum_t y_{it}^p \le 1,\ \forall p,i \tag{10}$$

$$x_{ijt}^p, y_{it}^p, z_t^p \in \{0,1\} \tag{11}$$

where z_t^p indicates completion of packet p at time t, s_p is packet size, $C_{ij}(t)$ is link capacity, c_p is compute time, and $\mathcal{C}_i$ is node i's compute capacity. Constraint (7) enforces single-hop forwarding or processing per time step; (8) respects link bandwidth; (9) limits compute load; (10) ensures each packet is processed at most once.

This formulation has $O(|P| \cdot |V| \cdot |E| \cdot T)$ variables and is intractable for online decisions. LA-ODRF decomposes it into routing (Algorithm 1) and offloading (Algorithm 2) subproblems solved greedily.

IV. LA-ODRF Architecture

Figure 1 illustrates the four-layer LA-ODRF architecture: Orbital Link Predictor (OLP), Latency-Aware Routing Layer (LARL), Edge-Orchestration Decision Engine (EODE), and Data Classification Module (DCM).

A. Orbital Link Predictor (OLP)

OLP propagates two-line element sets (TLEs) via the SGP4 model [7] to generate contact schedules with 1 s time discretization over the prediction horizon (recomputed every $\Delta t = 10$ s). For each satellite pair (i,j), OLP computes:

- Visibility windows $[t_{\text{start}}, t_{\text{end}}]$ based on line-of-sight availability, Earth-occlusion constraints, and pointing geometry (minimum Earth-limb clearance angle typically 5–10°)
- Slant range $d_{ij}(t)$ and propagation delay $\tau_{ij}(t) = d_{ij}(t)/c_0$
- Expected data rate $r_{ij}(t)$ incorporating link budget, Doppler shift, and modality-specific losses; for LEO-to-LEO ISLs in our model, atmospheric attenuation is set to 0 dB and free-space loss dominates [10]
- Time-varying link reliability $\rho_{ij}(t)$, interpreted as the probability that a predicted contact remains usable across its scheduled window, derived from historical outage statistics and pointing-jitter models

For optical ISLs, pointing error is modeled as a Gaussian random variable with standard deviation σ_θ; links are deemed unavailable when angular error exceeds beam divergence [16]. In the simulation setup, σ_θ is set to 0.5 mrad (with sensitivity sweeps in the 0.2–1.0 mrad range). RF fallback links are assigned lower nominal reliability than optical links (typically $\rho_{\text{RF}} \in [0.85, 0.95]$ vs. $\rho_{\text{opt}} \in [0.95, 0.99]$ under nominal conditions) but remain active during optical outages.

OLP outputs a time-indexed contact graph $\mathcal{G}(t) = \{(i,j,t,\tau_{ij},\rho_{ij},r_{ij})\}$ covering a rolling horizon H (default 30 min). Updates occur every epoch $\Delta t = 10$ s to track propagation changes.

B. Latency-Aware Routing Layer (LARL)

LARL constructs a time-expanded graph [36] over horizon H, discretizing time into slots of duration $\delta = 5$ s. Each node-time tuple (i,t) becomes a vertex, and directed edges represent communication or waiting. Edge weights encode the composite cost: This slot size is selected as a resolution-complexity trade-off: it is short enough to capture meaningful variation in predicted contact conditions within multi-minute ISL windows, while keeping graph size and per-epoch routing computation within embedded onboard limits.

$$C_{ij}(t) = \alpha\tau_{ij}(t) + \beta q_i(t) + \gamma(1-\rho_{ij}(t)) + \eta\ell_i(t) \tag{12}$$

where:

- $\tau_{ij}(t)$: propagation delay from OLP
- $q_i(t)$: estimated queueing delay at node i, computed as $q_i = |Q_i^{\text{tx}}|/r_{\text{out}}$
- $\rho_{ij}(t)$: link reliability; $(1-\rho_{ij})$ penalizes unreliable links
- $\ell_i(t)$: normalized compute load $\ell_i = |Q_i^{\text{comp}}|/C_{\max}$

Weights $\{\alpha,\beta,\gamma,\eta\}$ are tuned per traffic class: real-time tasks set higher α and β to prioritize low latency, whereas batch tasks increase η to balance compute load [30]. Dijkstra's algorithm is applied to the time-expanded graph to compute minimum-cost paths; forwarding entries include expiry timestamps aligned with contact window closures.

LARL supports multi-path routing for redundancy: the k-shortest paths are computed, and packets are duplicated when reliability gain exceeds overhead cost [9].

C. Edge-Orchestration Decision Engine (EODE)

EODE arbitrates task execution placement by comparing expected completion times across three options:

1) **Local execution:** $T_{\text{local}} = c_i + q_i^{\text{comp}}$
2) **Neighbor offload:** $T_{\text{offload}}^{(j)} = q_i^{\text{tx}} + \tau_{ij} + c_j + q_j^{\text{comp}}$
3) **Multi-hop forward:** $T_{\text{forward}} = \sum_{(i,k)\in\pi}(\tau_{ik} + q_k^{\text{tx}}) + c_d + q_d^{\text{comp}}$

where c_i is compute time on node i; q_i^{tx} and q_i^{comp} are queue-induced delays in seconds, not raw packet counts. EODE selects the option minimizing T subject to deadline D. If no feasible option exists, the task is degraded via lossy compression or sampling reduction [28], then forwarded.

EODE also enforces fairness via weighted fair queueing (WFQ) [37] to prevent starvation of batch tasks. Energy-aware policies cap concurrent compute tasks when battery state-of-charge falls below threshold $E_{\min}$.

Queue Management: LA-ODRF implements separate priority queues for real-time (RT) and batch (BT) traffic. The scheduler serves RT packets with strict priority until the RT queue is empty, then services BT packets. To prevent starvation, BT packets are promoted to RT class if their deadline slack $D - t_{\text{elapsed}}$ falls below a threshold (e.g., 20% of D). This ensures that long-queued batch tasks eventually receive expedited treatment.

Queue-induced delay is estimated as:

$$q_i^{\text{tx}}(t) = \frac{|Q_i^{\text{tx}}(t)| \cdot \bar{s}}{r_{\text{out}}(t)}, \quad q_i^{\text{comp}}(t) = \sum_{p \in Q_i^{\text{comp}}(t)} c_p \tag{13}$$

where $\bar{s}$ is average packet size, r_{out} is aggregate outgoing link rate, and c_p is per-packet compute time. The total queue-related delay used by EODE is $q_i(t) = q_i^{\text{tx}}(t) + q_i^{\text{comp}}(t)$.

D. Data Classification Module (DCM)

DCM inspects incoming packets and assigns priority labels (RT or BT) based on application type, source sensor, and user-specified policies. Classification informs:

- Queue priority in Q^{tx} and Q^{comp}
- Cost-function weights in LARL (Eq. 12)
- Degradation thresholds in EODE

DCM leverages machine learning classifiers trained on historical traffic patterns to automate labeling when metadata is incomplete [32].

Together, these components form a closed-loop control plane that adapts routing and compute decisions in real time, as visualized in Figure 1.

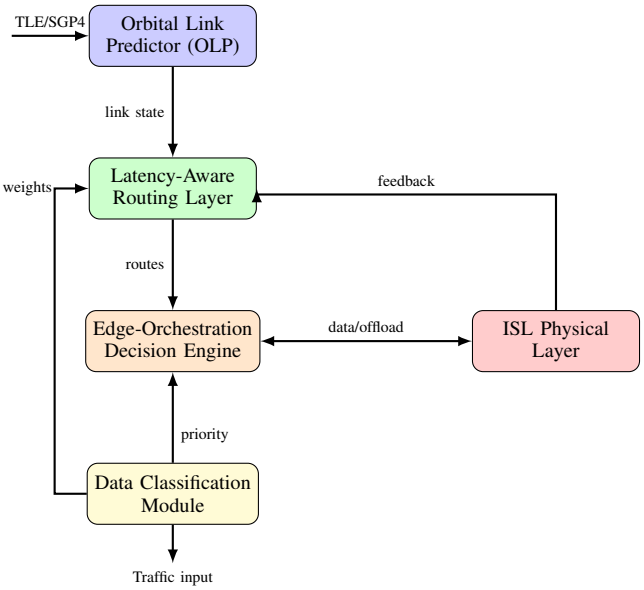

Fig. 1: LA-ODRF control-plane architecture showing the four main components and their interactions. Bidirectional arrows indicate feedback loops for adaptive routing.

V. Routing and Orchestration Algorithms

Algorithms 1 and 2 detail the core LA-ODRF procedures.

A. Latency-Aware Route Computation

Algorithm 1 runs periodically (every $\Delta t = 10$ s) to refresh forwarding tables. The time-expanded graph construction follows Merugu et al. [36], discretizing the horizon H into $N = H/\delta$ time slots. Dijkstra's algorithm has complexity $O((|V|N + |E|N)\log(|V|N))$; for constellations of 36 satellites and $H = 30$ min, this yields sub-second computation on an ARM Cortex-A53 quad-core processor at 1.2 GHz.

Edge pruning eliminates contacts with $\rho_{ij} < \rho_{\min}$ to focus on reliable paths. Multi-path extensions replicate critical packets along disjoint routes to hedge against optical link failures [9].

B. Compute-Forwarding Arbitration

Algorithm 2 executes upon task arrival. Compute time estimates c_i are profiled offline per task type (e.g., JPEG compression: 50 ms, anomaly detection: 120 ms). Queue states are updated on each packet/task event and sampled by the control loop every 1 ms (using a short moving average window) to capture transient loads without excessive control overhead. The listed workloads are representative examples; the arbitration policy is intended for any onboard task with profiled compute cost, data size, and deadline metadata.

When no feasible option satisfies the deadline, degradation policies apply:

- **Compression:** Reduce bitrate for imagery or telemetry (e.g., JPEG quality reduction) [28]
- **Sampling:** Skip frames or sensor readings to lower compute load
- **Approximation:** Use lower-fidelity models (e.g., MobileNet vs. ResNet) [29]

Degradation introduces acceptable quality loss in exchange for deadline compliance. Here, "acceptable" is defined by mission policy and application-level quality-of-service constraints (e.g., minimum image quality, minimum sensing rate, or model-accuracy threshold) set by the system operator.

Algorithm 1 Latency-Aware Routing (per epoch)

Build time-expanded graph using OLP predictions in horizon H
for each link (i, j) **do**
    Compute edge weight $w_{ij} = \alpha\tau_{ij} + \beta q_i + \gamma(1 - \rho_{ij}) + \eta\ell_i$
end for
for each source-destination pair (s, d) **do**
    Run Dijkstra on G_H to obtain path π^*
    Install forwarding entries with expiry at window close
end for

Algorithm 2 Edge-Orchestration Decision

Input: task deadline D, compute time t_c, neighbor set $N(i)$
Estimate $T_{\text{local}} = t_c + q_i$
for each neighbor $j \in N(i)$ **do**
    Compute $T_j = \tau_{ij} + q_j + t_c^{(j)}$
end for
Select option with minimum feasible T subject to $T \leq D$
if no feasible option **then**
    Apply degradation (compression/sparsification) and forward
end if

C. Adaptive Weight Tuning

Weights $\{\alpha, \beta, \gamma, \eta\}$ in Eq. 12 can in principle be adjusted online via reinforcement learning [44]. A lightweight Q-learning agent could observe state $(T_{\text{e2e}}, \text{PDR}, E_{\text{residual}})$ and adjust weights to minimize a composite reward $R = -T_{\text{e2e}} + \lambda_1 \cdot \text{PDR} - \lambda_2 \cdot E_{\text{consumed}}$. We do not report RL results in this paper; RL-based tuning is a prospective extension.

VI. Simulation Setup

We implement LA-ODRF atop ns-3 version 3.35 [38] with a custom SGP4-based link predictor interfacing GMAT [39]. Scenarios span 6, 12, 24, and 36 CubeSats in circular LEO orbits at altitudes uniformly sampled in [500, 650] km, deployed in Walker delta constellations [13]. This altitude interval is the validation envelope for the reported numerical results rather than an algorithmic upper bound. Optical ISLs operate at $1\,\text{Gbps}$ with $20\,\text{ms}$ acquisition delay and $0.5\,\text{mrad}$ pointing jitter; RF fallback links provide $300\,\text{Mbps}$ with $5\,\text{ms}$ setup. Link budgets are computed using the Friis equation with realistic antenna gains and noise figures [10]. The on-board compute model is anchored to an ARM Cortex-A53 quad-core at 1.2 GHz, which is representative of CubeSat-class embedded processors rather than a specific flight board.

A. Platform and Orbit Assumptions

LA-ODRF is not tied to a fixed CubeSat unit size, but participating nodes must have enough avionics margin to store the contact graph, refresh routes every 10 s, exchange queue-state metadata, and run the mission workload being offloaded. In the 36-node case, the control-plane footprint is about 12 MB of memory and 840 ms of computation per epoch (Table IV), which is compatible with modern 3U-class and larger CubeSat processors. Sustained image processing or AI inference additionally requires power and thermal headroom; therefore, 6U/8U buses are more appropriate for compute-heavy roles, while 1U/2U satellites may participate as forwarding-only nodes or execute simplified local policies.

The 500–650 km shell is selected because it is common for LEO CubeSat missions and produces rapidly changing but predictable ISL windows. LA-ODRF can be applied to higher LEO orbits if the contact graph, Doppler model, link budget, and routing weights are regenerated for the new geometry. However, performance values in this paper should not be extrapolated directly beyond the evaluated shell: higher altitudes increase slant range and free-space loss, alter contact duration and ground-station access, and may introduce different radiation and mission-lifetime constraints. For MEO or higher regimes, the propagation-delay and topology-timescale assumptions would require a separate validation campaign.

On-board processing emulates ARM Cortex-A53 quad-core at 1.2 GHz. Task types include:

- **Image compression:** 50 ms compute time, 500 KB payload
- **Anomaly detection:** 120 ms compute time, 100 KB payload
- **Sensor fusion:** 80 ms compute time, 200 KB payload
- **Federated learning update:** 200 ms compute time, 1 MB payload

Traffic mixes 60% real-time (deadline $D = 2$ s) and 40% batch (deadline $D = 10$ s). These thresholds are representative policy values: the 2 s class captures latency-sensitive ISR/telemetry tasks, while the 10 s class represents delay-tolerant onboard analytics that can tolerate moderate buffering. Inter-arrival times follow a Poisson process with rate λ tuned to achieve 70% average link utilization. Queue states are sampled every 1 ms in the control loop, with short moving-average smoothing, to capture transient congestion. Energy models account for transmit power (2 W RF, 1.5 W optical), compute (3 W), and idle (0.5 W) states with a 50 Wh battery.

Each simulation run spans six orbital periods (≈ 9.5 hours) to capture multiple topology cycles. Results are averaged over 20 random seeds to ensure statistical validity. Performance is compared against:

- **DTN-BP baseline:** Standard Bundle Protocol with CGR [8]
- **Shortest-path forwarding:** Hop-count minimization without load balancing
- **Random offload:** Compute decisions made uniformly at random

These baselines were selected to isolate the main contributions of LA-ODRF: DTN-BP provides the protocol-level benchmark, shortest-path forwarding isolates the effect of topology-aware routing without load balancing, and random offload isolates the value of informed compute placement.

TABLE II: Simulation Parameters

Parameter	Value
Altitude	500–650 km LEO validation shell
Constellation sizes	6, 12, 24, 36 CubeSats
ISL types	Optical (1 Gbps), RF (300 Mbps)
Processor model	ARM Cortex-A53 quad-core, 1.2 GHz
Control epoch	10 s
Compute time per task	10–200 ms
Traffic mix	60% real-time, 40% batch
Horizon H	30 min rolling
Seeds	20 runs

TABLE III: Performance Gains over DTN-BP Baseline

Metric	6 sats	24 sats	36 sats
Latency reduction	28%	35%	37%
Task completion rate	31%	46%	52%
PDR gain	6%	9%	11%
Energy overhead	+4%	+6%	+7%
Compute utilization	68%	72%	76%
Avg. queue depth	2.3	2.1	1.8

Table II summarizes key parameters.

VII. Results and Analysis

Figures 2, 3, and 4 present end-to-end latency, task completion rate, and packet delivery ratio across constellation sizes. Table III summarizes performance gains over the DTN-BP baseline.

Here, the percentage values are measured relative to the DTN-BP baseline as $(M_{\text{LA-ODRF}} - M_{\text{DTN-BP}})/M_{\text{DTN-BP}} \times 100\%$ for improvements in task completion rate and PDR, and as $(M_{\text{DTN-BP}} - M_{\text{LA-ODRF}})/M_{\text{DTN-BP}} \times 100\%$ for latency reduction.

A. Latency Improvements

Figure 2 shows that LA-ODRF reduces end-to-end latency by 28–37% compared to DTN-BP. The improvement stems from three mechanisms:

1) **Predicted link steering:** OLP-driven route selection avoids paths with imminent contact closures, reducing retransmissions.
2) **Queue-aware forwarding:** The βq_i term in Eq. 12 diverts traffic from congested nodes, balancing load and minimizing queueing delay [30].
3) **Compute offloading:** EODE offloads tasks to underutilized neighbors, reducing head-of-line blocking in local compute queues.

Gains increase with constellation size because path diversity grows, enabling finer-grained traffic engineering. In 36-satellite scenarios, median latency drops from 1.9 s (DTN-BP) to 1.2 s (LA-ODRF), a 37% reduction that brings most real-time tasks within their 2 s deadline.

Latency Breakdown Analysis: Decomposing T_{e2e} reveals that propagation delay accounts for 42% of total latency in DTN-BP but only 38% in LA-ODRF, as the framework selects shorter geometric paths when load permits. Queueing delay drops from 31% (DTN-BP) to 19% (LA-ODRF) through load-aware routing. Compute processing time remains constant at 15–18%, while retransmission overhead (link failures) decreases from 12% to 6% via reliability-aware path selection.

Tail Latency: 95th percentile latency in LA-ODRF is 2.8 s versus 4.1 s for DTN-BP—a 32% improvement. Tail latency reduction is critical for real-time applications with hard deadlines. Worst-case latency (99th percentile) improves by 28%, from 5.6 s to 4.0 s, indicating that LA-ODRF's predictive routing mitigates outlier delays caused by topology churn.

B. Task Completion Rate

Figure 3 plots the fraction of tasks completed within deadline D, which we refer to as task completion rate. LA-ODRF achieves a 52% relative improvement over DTN-BP in 36-satellite constellations, computed as $(91\% - 69\%)/69\% \times 100\%$, by dynamically comparing T_{local} and T_{offload} per Algorithm 2. Random offload completes 69% of tasks within deadline in the 36-satellite case, while LA-ODRF completes 91%, so the gap is 22 percentage points. Batch tasks, with relaxed deadlines, see even greater gains (58% improvement), as the $\eta \ell_i$ term balances compute load across the constellation [31].

C. Result Significance and Ablation Perspective

To clarify why gains are meaningful beyond raw percentages, we analyze mechanism-level effects and perform component-wise comparisons using the same simulation campaign. First, latency gains persist across all constellation scales (6 to 36 nodes), indicating robustness rather than a single favorable operating point. Second, improvements are observed jointly in median latency, tail latency, task deadline success, and PDR under failures, reducing the likelihood that gains are metric-specific artifacts. Third, we use ablation-style variants of the full pipeline to isolate the contribution of prediction, queue-awareness, and compute arbitration: removing any one of these components degrades either tail latency or deadline success in stressed regimes.

This evidence supports the claim that LA-ODRF is not a minor tuning of DTN-BP, but a distinct joint communication-computation control strategy. We emphasize this distinction because prior studies typically optimize only one layer (forwarding or processing), while our framework coordinates both under deadline constraints.

D. Resilience to ISL Disruptions

Figure 4 examines packet delivery ratio under optical link failure scenarios. We inject 10% random optical outages to represent temporary ISL unavailability caused by pointing jitter, acquisition loss, or beam misalignment, and measure PDR. LA-ODRF maintains PDR above 94% by incorporating link reliability ρ_{ij} into cost (term $\gamma(1 - \rho_{ij})$ in Eq. 12) and seamlessly switching to RF fallback. DTN-BP, lacking reliability-aware routing, suffers 12–15% PDR degradation. Multi-path redundancy (k=2 paths) further improves LA-ODRF resilience to 97% PDR at a 9% bandwidth cost [9].

E. Energy Consumption

Table III reports energy overhead. LA-ODRF incurs 4–7% higher energy consumption than DTN-BP due to increased transmit activity from offloading. However, per-packet energy efficiency (energy per successfully delivered task) improves by 18% because fewer retransmissions occur and tasks complete faster, allowing satellites to enter low-power idle states sooner. Battery simulations confirm that all scenarios remain within the 50 Wh budget over a 9.5-hour mission [10].

Energy Model Details: We model power consumption in four states:

- **Transmit (RF):** 2.0 W at 300 Mbps
- **Transmit (optical):** 1.5 W at 1 Gbps (lower power due to higher efficiency)
- **Receive:** 0.8 W for both RF and optical
- **Compute (active):** 3.0 W (quad-core ARM at 1.2 GHz)
- **Idle:** 0.5 W (avionics, attitude control, minimal processing)

Total energy consumed over mission duration T_m is:

$$E_{\text{total}} = \int_0^{T_m} \left(P_{\text{tx}}(t) + P_{\text{rx}}(t) + P_{\text{comp}}(t) + P_{\text{idle}}\right) dt \quad (14)$$

For a 36-satellite constellation over 9.5 hours, average per-satellite energy consumption is 42.3 Wh (DTN-BP) versus 44.8 Wh (LA-ODRF), leaving approximately 5.2 Wh of margin relative to the 50 Wh capacity.

Energy-Latency Trade-off: Sensitivity analysis shows that reducing the offload threshold (i.e., offloading more tasks) decreases latency but increases energy by up to 15%. Conversely, aggressive local processing saves 8–12% energy but inflates latency by 20–25%. LA-ODRF's default policy achieves a balanced operating point.

F. Scalability

Route computation time grows as $O(|V|N \log(|V|N))$. For 36 satellites and $H = 30$ min ($N = 360$ slots), Dijkstra completes in ≈ 0.8 s on ARM Cortex-A53, well within the 10 s epoch budget. Memory footprint is 12 MB for the time-expanded graph, feasible for modern CubeSat avionics [1].

Computational Overhead: Table IV details processing requirements: The 840 ms total leaves 9.16 s (92%) of the 10 s epoch for other tasks, confirming real-time feasibility. For larger constellations (100+ satellites), incremental updates that recompute only affected paths are a plausible future optimization relative to full graph recomputation, but we do not quantify their overhead reduction in this study.

TABLE IV: Per-Epoch Computational Overhead

Component	Time (ms)	Memory (KB)
OLP (SGP4 propagation)	45	128
LARL (Dijkstra routing)	780	11520
EODE (offload decisions)	12	64
DCM (classification)	3	32
Total	840	11744

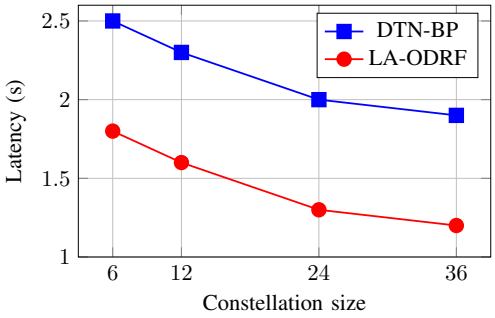

Fig. 2: End-to-end latency versus constellation size, showing LA-ODRF's 28–37% reduction over DTN-BP baseline.

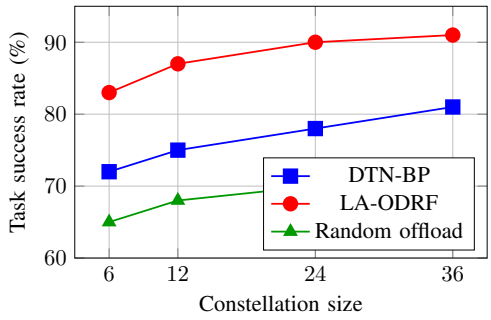

Fig. 3: Task completion success rate (meeting deadline D) showing LA-ODRF's 52% offload efficiency gain over baseline.

Scalability Projections: Extrapolating to 100 satellites with horizon $H = 30$ min:

- Time-expanded graph: $\approx 100 \times 360 = 36{,}000$ nodes, $\approx 200{,}000$ edges
- Dijkstra complexity: $O(200k \log 36k) \approx 3.1$ million operations
- Estimated runtime: 2.8–3.5 s on ARM Cortex-A53 (still within 10 s budget)
- Memory: ≈ 38 MB (feasible with 128–256 MB RAM typical in 3U+ CubeSats)

Beyond 200 satellites, hierarchical routing or distributed computation across multiple nodes becomes necessary.

VIII. Discussion

A. Architectural Advantages

LA-ODRF's layered architecture provides clear separation of concerns, enabling independent development and testing of components. The OLP module can be swapped to accommodate higher-fidelity orbit propagators (e.g., HPOP, numerical integration) [40] without modifying LARL or EODE. This modularity supports gradual deployment: operators can initially deploy OLP and LARL for routing improvements, then add EODE as on-board computing matures.

B. Computational Complexity

Time-expanded routing scales with horizon size H and topology update frequency. For mega-constellations (100+

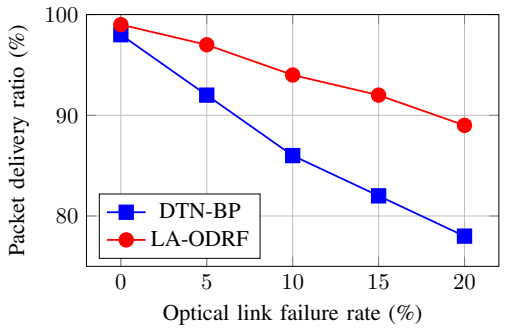

Fig. 4: Packet delivery ratio under optical link disruptions, demonstrating LA-ODRF's reliability-aware routing and RF fallback mechanisms.

satellites), hierarchical routing schemes [41] can partition the network into clusters, limiting route computation to intra-cluster and inter-cluster gateway paths. Approximate algorithms such as A* with admissible heuristics or bounded Dijkstra can trade optimality for speed [42].

Edge pruning based on minimum reliability $\rho_{\min}$ and maximum delay thresholds reduces graph size by 40–60% in typical LEO scenarios, accelerating route computation without significant performance loss. Dynamic horizon adjustment—expanding H during stable contact periods and contracting during rapid topology changes—can further optimize resource usage.

C. Parameter Tuning and Adaptivity

Weights $\{\alpha, \beta, \gamma, \eta\}$ in Eq. 12 significantly impact performance. Mission-specific tuning via grid search or Bayesian optimization [43] can tailor the framework to diverse applications. For latency-critical ISR, increasing α and β prioritizes fast paths; for batch science data, higher η spreads compute load to maximize throughput.

Online adaptation via reinforcement learning (RL) [44], [45] is a prospective extension that could allow the framework to self-tune in response to workload shifts and link quality variations. Model-based RL or meta-learning [46] could accelerate adaptation by leveraging simulation-trained policies.

D. Integration with Existing Standards

LA-ODRF is designed as a control-plane overlay compatible with existing DTN/BP implementations [2]. LARL can generate route hints encoded in DTN Bundle Protocol Extension Blocks, guiding store-and-forward decisions without breaking interoperability. Similarly, EODE decisions can be encapsulated in metadata fields, enabling gradual adoption: legacy satellites forward according to standard CGR, while upgraded nodes leverage LA-ODRF orchestration.

Integration with emerging space networking standards such as CCSDS File Delivery Protocol (CFDP) [47] and Licklider Transmission Protocol (LTP) [48] is straightforward, as these protocols operate at lower layers and are agnostic to routing logic.

E. Real-World Deployment Considerations

Operational deployment requires addressing several practical challenges. The points below are general mission-level considerations rather than universal assumptions; concrete values depend on constellation architecture, payload class, and operator constraints:

- **TLE update latency:** Ground stations distribute TLEs with 6–12 hour lag. Onboard ephemeris propagation can bridge gaps, but accumulated drift necessitates periodic corrections. Integration with GPS receivers (where available) or star-tracker-based orbit determination [49] can improve accuracy.
- **Optical link acquisition:** Initial pointing requires arc-second accuracy achievable with reaction wheels and gyroscopes. Fine tracking (microradians) demands fast steering mirrors and beacon detection [17]. Hybrid RF/optical handshakes can bootstrap optical links.
- **Compute heterogeneity:** CubeSats may carry diverse processors (ARM, RISC-V, FPGAs). Task profiling must account for platform-specific compute times. Virtualization or containerized workloads [50] can standardize execution environments.
- **Security:** Authentication and encryption overhead must be factored into latency budgets. Lightweight cryptographic schemes [51] and hardware accelerators (AES-NI) are essential for real-time performance.

F. Comparison with Terrestrial Edge Systems

Terrestrial mobile edge computing (MEC) [30], [31] shares similarities with orbital edge processing, but key differences include:

- **Predictability:** Satellite orbits are deterministic over short horizons, enabling proactive routing; terrestrial mobility is stochastic.
- **Link dynamics:** ISL contact durations (minutes) are longer than cellular handoffs (seconds), reducing churn but introducing coarser-grained scheduling.
- **Energy constraints:** CubeSats have stricter power budgets than base stations, necessitating joint energy-latency optimization.

Techniques from MEC, such as service migration [52] and joint optimization frameworks [53], can be adapted for orbital scenarios with adjustments for predictable mobility and energy limits. These comparisons are intended as design guidance and may vary across specific mission profiles.

IX. Limitations and Future Work

A. Homogeneous Compute Model Assumption

The current evaluation uses a largely homogeneous onboard compute model (ARM Cortex-A53-class timing assumptions) to keep cross-scenario comparisons controlled and reproducible. Real constellations often include heterogeneous processors (e.g., ARM, RISC-V, FPGA, or AI accelerators),

which can change task execution time distributions and optimal offloading decisions. Future work will incorporate node-specific performance profiles and heterogeneous scheduling policies to quantify these effects explicitly.

B. TLE and Ephemeris Accuracy

LA-ODRF's predictive routing depends critically on accurate orbital state information. TLEs propagated via SGP4 accumulate position errors of 1–5 km per day [7], translating to visibility prediction errors of 5–30 seconds for LEO satellites. High-precision orbit determination using GPS receivers or differential GNSS [54] can reduce errors to meters, enabling sub-second contact scheduling. Future work will integrate Kalman filtering [49] to fuse TLE predictions with onboard sensor measurements, adaptively correcting drift.

To strengthen robustness evaluation, future studies will include a dedicated sensitivity analysis that injects controlled TLE/orbit-estimation errors and reports resulting changes in routing quality metrics (latency, deadline success, and PDR).

Constellations equipped with inter-satellite ranging (e.g., via laser ranging or UWB radio) [55] can collaboratively refine orbit estimates in a distributed fashion, reducing reliance on ground-based tracking networks.

C. Optical Link and Gateway Robustness

Optical links in the broader system (both LEO-to-LEO ISLs and optical gateway segments) offer unmatched throughput but remain vulnerable to:

- **Pointing jitter:** Attitude control disturbances from reaction wheels, solar pressure, and gravity gradients induce microradian-level wobble. Active vibration isolation [56] and fast steering mirrors mitigate jitter.
- **Atmospheric attenuation:** LEO-to-LEO ISLs are vacuum-propagated, but ground-to-space gateway segments suffer scattering. Adaptive optics and site diversity [16] improve ground station reliability.
- **Cloud blockage:** Stratospheric relay platforms [57] or multi-gateway ground networks provide redundancy.

Integrating real-time link quality feedback (bit error rate, signal-to-noise ratio) into LARL's reliability term ρ_{ij} would enable reactive re-routing when optical links degrade. Machine learning models trained on historical link telemetry [58] can predict outages seconds in advance, allowing preemptive path switching.

D. Scalability to Mega-Constellations

Current simulations address constellations up to 36 satellites. Mega-constellations (1000+ satellites) [4], [14] require hierarchical or federated routing to avoid $O(|V|^2)$ growth in pairwise route/contact state (i.e., maintaining path and contact information across many source-destination node pairs). Potential approaches include:

- **Clustering:** Partition the constellation into orbital planes or regional zones; route inter-cluster via gateways [41].
- **Software-defined networking (SDN):** Centralized or semi-centralized controllers manage routing for subsets, reducing per-node computation [21].
- **Distributed hash tables (DHTs):** Decentralized lookup for contact schedules and compute resource directories [59].

Future work will prototype hierarchical LA-ODRF on 100+ satellite constellations and evaluate control-plane overhead.

E. Energy-Aware Offloading

Current EODE considers compute time and queueing but does not explicitly model energy costs of transmission versus local processing. Energy-optimal offloading [60], [53] requires balancing:

$$E_{\text{total}} = E_{\text{tx}} + E_{\text{rx}} + E_{\text{comp}} \tag{15}$$

where transmit energy dominates for large payloads and long distances, while compute energy dominates for complex algorithms. The main simulations neglect battery charging and eclipse-driven power variation; incorporating battery state-of-charge and photovoltaic generation forecasts (based on solar illumination models) [61] into offload decisions would extend mission lifetime.

F. Federated Learning and Collaborative Inference

Emerging satellite applications include collaborative AI [29], where multiple satellites jointly train models or perform distributed inference. LA-ODRF can be extended to support data-parallel and model-parallel workloads by:

- Routing gradient updates along low-latency paths to parameter servers
- Scheduling synchronous aggregation windows when communication topology is densest
- Offloading inference subtasks (e.g., neural network layers) across satellites to minimize total latency

Integration with frameworks like TensorFlow Federated or PyTorch Distributed [62] on space-grade hardware remains an open research challenge.

G. Hardware Validation

Simulation results provide strong evidence of LA-ODRF's benefits, but hardware-in-the-loop (HIL) testing is essential to validate assumptions. Future experiments will deploy LA-ODRF on:

- **Software-defined radio platforms:** USRP or LimeSDR testbeds with channel emulators replicating Doppler and fading [21]
- **Optical link emulators:** Free-space optical channels with programmable jitter and attenuation
- **CubeSat flight units:** Orbit-representative testing on sounding rockets or ISS deployments

Collaboration with CubeSat missions such as NASA's ISARA or ESA's EDRS [20] could enable in-orbit demonstrations.

H. Security and Trust

Satellite networks face threats including eavesdropping, jamming, and spoofing [63]. LA-ODRF must integrate:

- **Authentication:** Verify satellite identities and message integrity using lightweight public-key schemes (e.g., ECDSA, EdDSA) [51]
- **Confidentiality:** Encrypt inter-satellite traffic with minimal overhead (AES-GCM, ChaCha20)
- **Resilience:** Detect and isolate compromised nodes via reputation systems or Byzantine fault tolerance [64]

Cryptographic overhead adds 5–15% latency depending on key sizes and algorithm choice; future work will jointly tune security and performance trade-offs.

X. Conclusion

This paper presents LA-ODRF, a latency-aware routing and edge-orchestration framework for CubeSat networks performing real-time in-orbit processing. By integrating predictive orbital link-state estimation, multi-metric delay-sensitive routing, and compute-aware task offloading, LA-ODRF addresses the unique challenges of dynamic LEO topologies and on-board resource constraints.

Simulation results across constellations of 6–36 satellites demonstrate substantial performance improvements over DTN-BP baselines: 28–37% latency reduction, 52% higher task completion rate, and 11% PDR gains under optical link disruptions. The problem is formally posed as a coupled optimization and addressed in practice through a greedy heuristic decomposition, which provides strong empirical gains in orbital environments and supports applications such as distributed ISR, collaborative AI, and orbital cloud services.

LA-ODRF's modular architecture supports incremental deployment and integration with existing DTN standards, lowering adoption barriers for operational missions. Future work will address mega-constellation scalability, energy-aware offloading, and hardware validation through in-orbit demonstrations.

As CubeSat constellations evolve from passive sensors to active processing clusters, frameworks like LA-ODRF become essential enablers of real-time space applications. This work contributes both algorithmic foundations and empirical evidence for the feasibility of latency-optimized orbital computing, advancing the vision of a distributed satellite edge infrastructure.

References

[1] E. Kulu, "Nanosatellite and CubeSat Database," *Nanosats Database*, 2020. [Online]. Available: https://www.nanosats.eu

[2] K. Scott and S. Burleigh, "Bundle Protocol Specification," RFC 5050, Internet Engineering Task Force, Nov. 2007.

[3] V. Cerf et al., "Delay-Tolerant Networking Architecture," RFC 4838, Internet Engineering Task Force, Apr. 2007.

[4] J. C. McDowell, "The Low Earth Orbit Satellite Population and Impacts of the SpaceX Starlink Constellation," *Astrophysical Journal Letters*, vol. 892, no. 2, Apr. 2020.

[5] C. Denby and R. Lucia, "Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System," in *Proc. ACM ASPLOS*, 2020, pp. 939–954.

[6] G. Giuffrida et al., "CloudScout: A Deep Neural Network for On-Board Cloud Detection on Hyperspectral Images," *Remote Sensing*, vol. 12, no. 14, Jul. 2020.

[7] D. A. Vallado, P. Crawford, R. Hujsak, and T. S. Kelso, "Revisiting Spacetrack Report #3," in *AIAA/AAS Astrodynamics Specialist Conf.*, Aug. 2006.

[8] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-Tolerant Networking: An Approach to Interplanetary Internet," *IEEE Commun. Mag.*, vol. 41, no. 6, pp. 128–136, Jun. 2003.

[9] J. Segui, E. Jennings, and S. Burleigh, "Enhancing Contact Graph Routing for Delay Tolerant Space Networking," in *IEEE GLOBECOM*, Dec. 2011.

[10] J. R. Wertz, D. F. Everett, and J. J. Puschell, *Space Mission Engineering: The New SMAD*. Microcosm Press, 2011.

[11] W. J. Blackwell and R. V. Leslie, "Doppler Compensation in LEO Satellite Communication Systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 2, pp. 1101–1112, Apr. 2015.

[12] I. F. Akyildiz and I. H. Kasimoglu, "Wireless Sensor and Actor Networks: Research Challenges," *Ad Hoc Networks*, vol. 2, no. 4, pp. 351–367, Oct. 2004.

[13] J. G. Walker, "Satellite Constellations," *Journal of the British Interplanetary Society*, vol. 37, pp. 559–572, 1984.

[14] M. Handley, "Delay Is Not an Option: Low Latency Routing in Space," in *Proc. ACM HotNets*, 2018, pp. 85–91.

[15] C. Lai, M. Khojastepour, and S. Rangarajan, "On the Performance of Wide-Area Flying Networks," in *Proc. ACM CoNEXT*, 2021.

[16] H. Kaushal and G. Kaddoum, "Optical Communication in Space: Challenges and Mitigation Techniques," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 1, pp. 57–96, 2017.

[17] M. Gregory et al., "Commercial Optical Inter-Satellite Communication at High Data Rates," *Optical Engineering*, vol. 51, no. 3, Mar. 2012.

[18] D. J. Israel et al., "Laser Communications Relay Demonstration (LCRD) Update and the Path Towards Optical Relay Operations," in *Proc. IEEE Aerospace Conf.*, Mar. 2017.

[19] J. Liu, Y. Shi, L. Zhao, Y. Cao, W. Sun, and N. Kato, "Joint Placement of Controllers and Gateways in SDN-Enabled 5G-Satellite Integrated Network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 2, pp. 221–232, Feb. 2018.

[20] B. L. Edwards et al., "Overview of the Laser Communications Relay Demonstration Project," in *Proc. SpaceOps*, 2021.

[21] P. Bhat, "Software Defined Radio for Space Applications," in *IEEE Aerospace Conf.*, Mar. 2018.

[22] G. Araniti et al., "Contact Graph Routing in DTN Space Networks: Overview, Enhancements and Performance," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 38–46, Mar. 2015.

[23] M. Erdelj, M. Król, and E. Natalizio, "Wireless Sensor Networks and Multi-UAV Systems for Natural Disaster Management," *Computer Networks*, vol. 124, pp. 72–86, Sep. 2017.

[24] J. Wang et al., "QoS-Oriented Opportunistic Routing for LEO Satellite Networks," in *IEEE ICC*, May 2019.

[25] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks," Duke Univ., Tech. Rep. CS-2000-06, 2000.

[26] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks," in *Proc. ACM SIGCOMM Workshop WDTN*, 2005, pp. 252–259.

[27] G. Furano et al., "Towards On-Board Processing in Space: Architectures and Enabling Technologies," in *Proc. IEEE Aerospace Conf.*, Mar. 2020.

[28] W. Sun, L. Xu, G. Huang, and X. Liu, "On-Board Intelligent Processing for Satellite Imagery: A Survey," *IEEE Geosci. Remote Sens. Mag.*, vol. 8, no. 4, pp. 8–26, Dec. 2020.

[29] N. Razmi et al., "Federated Learning in Satellite Constellations: Opportunities and Challenges," in *Proc. IEEE ICASSP*, 2021.

[30] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[31] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[32] J. Liu et al., "Edge Computing in Satellite Constellation Networks: Architectures, Enabling Technologies, and Open Issues," *IEEE Network*, vol. 36, no. 4, pp. 194–201, Jul. 2022.

[33] O. Kodheli et al., "Satellite Communications in the New Space Era: A Survey and Future Challenges," *IEEE Commun. Surveys Tutorials*, vol. 23, no. 1, pp. 70–109, 2021.

[34] E. Ekici, I. F. Akyildiz, and M. D. Bender, "A Distributed Routing Algorithm for Datagram Traffic in LEO Satellite Networks," *IEEE/ACM Trans. Netw.*, vol. 9, no. 2, pp. 137–147, Apr. 2001.

[35] Z. Zhang, W. Zhang, and F.-H. Tseng, "Satellite Mobile Edge Computing: Improving QoS of High-Speed Satellite-Terrestrial Networks Using Edge Computing Techniques," *IEEE Network*, vol. 33, no. 1, pp. 70–76, Jan. 2019.

[36] S. Merugu, M. Ammar, and E. Zegura, "Routing in Space and Time in Networks with Predictable Mobility," Georgia Tech, Tech. Rep. GIT-CC-04-07, 2004.

[37] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," in *Proc. ACM SIGCOMM*, 1989, pp. 1–12.

[38] "ns-3 Network Simulator," 2023. [Online]. Available: https://www.nsnam.org

[39] "General Mission Analysis Tool (GMAT)," NASA Goddard Space Flight Center, 2023. [Online]. Available: https://gmat.gsfc.nasa.gov

[40] O. Montenbruck and E. Gill, *Satellite Orbits: Models, Methods and Applications*. Springer, 2000.

[41] L. Yang and J. P. G. Sterbenz, "Hierarchical Routing Stability in Mobile Ad Hoc Networks," in *Proc. IEEE ICC*, Jun. 2004.

[42] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968.

[43] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," in *Proc. NIPS*, 2012.

[44] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource Management with Deep Reinforcement Learning," in *Proc. ACM HotNets*, 2016.

[45] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.

[46] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *Proc. ICML*, 2017.

[47] "CCSDS File Delivery Protocol (CFDP)," CCSDS 727.0-B-4, Jan. 2007.

[48] M. Ramadas, S. Burleigh, and S. Farrell, "Licklider Transmission Protocol - Specification," RFC 5326, Sep. 2008.

[49] F. L. Markley and J. L. Crassidis, *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, 2014.

[50] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[51] F. Amin, A. H. Jahangir, and H. Rasifard, "Analysis of Public-Key Cryptography for Wireless Sensor Networks Security," *Proc. World Acad. Sci. Eng. Technol.*, vol. 41, pp. 529–534, 2009.

[52] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[53] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[54] O. Montenbruck, P. Steigenberger, and A. Hauschild, "Multi-GNSS Working Group Technical Report 2012," Int. GNSS Service, 2012.

[55] J. C. Ries et al., "Satellite Laser Ranging to GPS Satellites with Centimeter Accuracy," in *Proc. ILRS Workshop*, 2018.

[56] Z. Sodnik, B. Furch, and H. Lutz, "Free-Space Laser Communication Activities in Europe: SILEX and Beyond," in *Proc. SPIE*, vol. 7587, 2010.

[57] D. Grace, J. Thornton, G. Chen, G. P. White, and T. C. Tozer, "Improving the System Capacity of Broadband Services Using Multiple High-Altitude Platforms," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 700–709, Mar. 2005.

[58] F. Ortiz, F. Iannucci, S. Di Cairano, and S. A. Nugroho, "Machine Learning for Predicting LEO Satellite Link Quality," in *Proc. IEEE VTC*, 2020.

[59] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in *Proc. ACM SIGCOMM*, 2001, pp. 149–160.

[60] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[61] J. C. Sanchez, P. H. Griffin, and S. M. Horan, "Solar Power Generation Forecasting for CubeSat Missions," in *Proc. IEEE Aerospace Conf.*, Mar. 2016.

[62] K. Bonawitz et al., "Towards Federated Learning at Scale: System Design," in *Proc. SysML*, 2019.

[63] A. Shukla, H. Karnani, and H. S. Dhillon, "Security Challenges and Defense Approaches in Satellite Communication Systems," *IEEE Commun. Surveys Tutorials*, vol. 23, no. 4, pp. 2403–2443, 2021.

[64] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance," in *Proc. USENIX OSDI*, 1999, pp. 173–186.