SkyLink: Scalable and Resilient Link Management in LEO Satellite Networks

Wanja de Sombre [*], Arash Asadi [†], Debopam Bhattacherjee [‡], Deepak Vasisht [§], Andrea Ortiz [¶]
[*]Communications Engineering Lab, Technical University of Darmstadt, Germany, w.sombre@nt.tu-darmstadt.de
[†]WISE Lab, Delft University of Technology, The Netherlands, a.asadi@tudelft.nl
[‡]Microsoft Research, India, debopamb@microsoft.com
[§]University of Illinois Urbana-Champaign, USA, deepakv@illinois.edu
[¶]Institute of Telecommunications, Vienna University of Technology, Austria, andrea.ortiz@tuwien.ac.at

***Abstract*—The rapid growth of space-based services has established Low Earth Orbit (LEO) satellite networks as a promising option for global broadband connectivity. Next-generation LEO networks leverage inter-satellite links (ISLs) to provide faster and more reliable communications compared to traditional bent-pipe architectures, even in remote regions. However, the high mobility of satellites, dynamic traffic patterns, and potential link failures pose significant challenges for efficient and resilient routing. To address these challenges, we model the LEO satellite network as a time-varying graph comprising a constellation of satellites and ground stations. Our objective is to minimize a weighted sum of average delay and packet drop rate. Each satellite independently decides how to distribute its incoming traffic to neighboring nodes in real time. Given the infeasibility of finding optimal solutions at scale, due to the exponential growth of routing options and uncertainties in link capacities, we propose SkyLink, a novel fully distributed learning strategy for link management in LEO satellite networks. SkyLink enables each satellite to adapt to the time-varying network conditions, ensuring real-time responsiveness, scalability to millions of users, and resilience to network failures, while maintaining low communication overhead and computational complexity. To support the evaluation of SkyLink at global scale, we develop a new simulator for large-scale LEO satellite networks. For 25.4 million users, SkyLink reduces the weighted sum of average delay and drop rate by 29% compared to the bent-pipe approach, and by 92% compared to Dijkstra. It lowers drop rates by 95% relative to k-shortest paths, 99% relative to Dijkstra, and 74% compared to the bent-pipe baseline, while achieving up to 46% higher throughput. At the same time, SkyLink maintains constant computational complexity with respect to constellation size.**

I. Introduction

Driven by lower satellite development and deployment costs, this century has seen the rapid growth of space-based services. As a result, space broadband services have become widely accessible [1]–[5]. In fact, multiple reports highlight this sector as a key growth area with a strong cumulative growth rate [6]–[9]. Early satellite broadband services used bent-pipe connectivity, i.e., data traveled from a user terminal to a satellite and directly back to a ground station [10]. However, this method becomes impractical in remote areas without nearby ground stations. Consequently, providers have started to shift towards Inter-Satellite Links (ISLs), where data is relayed between satellites at the speed of light before reaching the ground [11], [12]. Advantages of using ISLs include reduced delay and higher bandwidth, which is essential for broadband networks [13].

Efficient use of ISLs demands dynamic link management which is challenging due to continuously evolving topologies. Satellites therefore employ regenerative payloads[1], which allow them to manage data flows and perform real-time routing (cf. 3GPP TR 38.811 [31]). Yet, identifying globally optimal link configurations remains difficult due to the exponential growth in routing options as the constellation scales to hundreds of satellites. This complexity is further increased by terabit-scale traffic demands and the need to continuously react to topology changes and failures, whether due to system faults or external events such as solar storms [29], [32].

Link management in LEO satellite networks can be approached using classical shortest-path algorithms [28] or more recent machine-learning-based solutions [14], [15]. While shortest-path algorithms, such as Dijkstra's or k-shortest path, are widely used in terrestrial networks, they fall short in LEO scenarios [28]. These algorithms often result in paths that overlap on the same Ground-Station-Satellite Links (GSLs), creating bottlenecks in areas where large volumes of data need to be transmitted simultaneously or where the GSLs have limited capacities. To avoid such bottlenecks, recent research has explored the use of Deep Q-Networks (DQNs) to redistribute traffic and ensure efficient network performance [14], [15]. However, both, shortest-path algorithms and learning solutions often rely on central controllers to monitor network failures and propagate updated routes to all nodes. The controllers are usually located on earth due to the limited computational capabilities of LEO satellites. As a result, continuous collection of global network state information is required causing slow response time to rapid topology changes, and introducing communication overhead. In highly dynamic networks, any delay in distributing updated control commands can result in inefficient link management, increased packet drop rate, and higher delays. Additionally, the dependency on a central controller reduces the resilience of the system, as failures need

This work was funded by the BMBF project Open6GHub under grant 16KISKO14, by DAAD with funds from the German Federal Ministry of Research, Technology and Space BMFTR, and by the LOEWE Center emergenCity under grant LOEWE/1/12/519/03/05.001(0016)/72. The work of Andrea Ortiz was funded by the Vienna Science and Technology Fund WWTF Grant ID 10.47379/VRG23002.

[1]Enabling onboard signal processing and routing.

	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	[24]	[25]	[26]	[27]	[28]	[29]	[30]	Ours
Joint Minimization of Delay & Drop Rate	✓	✓	✓	✓				✓	✓	✓	✓	✓			✓	✓	✓	✓
Dynamic Network		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ISLs and GSLs				✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Multi-Path Routing					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Distributed Approach	✓				✓				✓	✓	✓				✓	✓		✓
Large Constellation			✓										✓	✓	✓	✓	✓	✓
Global Broadband Communication												✓					✓	✓
Resilience to Network Failures															✓	✓	✓	✓

TABLE I: Summary of the related work on routing in LEO constellations.

to be centrally detected and accounted for.

A partial solution to the necessity for a central controller is to offload link management decisions to ground stations or higher-altitude Medium Earth Orbit (MEO) satellites, where computational resources are less constrained than on LEO satellites [33]–[36]. However, communication to ground stations or MEO satellites introduces additional delay [37] and still requires real-time network state information, which is difficult to maintain given the high mobility and frequent handovers. Additionally, failures in connections to MEO satellites can lead to substantial network disruptions, as link management decisions for entire regions might become inaccessible. Instead of relying on a central controller, pre-trained models can be periodically distributed across the network. However, this comes at the cost of slower response times to disturbances.

The development of a fully distributed, failure-resilient, and scalable link management strategy is as a key research challenge for advancing LEO satellite networks. Therefore, in this paper, we introduce SkyLink, a distributed learning approach that directly tackles the scalability, resilience, and computational complexity of link management in LEO satellite networks. SkyLink combines the Multi-Armed Bandit (MAB) framework [38] with tile coding to enable each satellite to autonomously and efficiently prioritize its communication links using only local information. By making real-time, decentralized decisions, SkyLink minimizes the delay and drop rate even under heavy traffic conditions and satellite failures, and without relying on global coordination.

The evaluation of link management solutions for LEO satellite networks demands global scale analysis. However, existing simulations tools are often limited to network segments (see e.g. [14], [22]). Thus, to realistically benchmark SkyLink, we introduce a novel simulator capable of modeling entire LEO satellite networks. Our simulator models global-scale traffic and allows us to validate SkyLink's performance across various scenarios including millions of users, high traffic volumes, and network failures.

In summary, the primary contributions of this work are:

- We introduce SkyLink, a fully distributed link management approach to jointly minimize the delay and drop rate in LEO satellite networks. Specifically, we break the complexity of the problem by using MAB-based learning including tile coding mechanisms. Due to its distributed nature, SkyLink is scalable in network size and data volume and resilient to network failures.
- For a realistic evaluation, we present a new, extensive and fast simulator capable of modeling LEO satellite networks globally. Our simulator includes advanced tools for space broadband simulations, detailed channel models for ISLs and GSLs, a global data generation framework, a representation of network data streams, and diverse visualization options[2].
- We show via extensive evaluation that SkyLink significantly reduces delay and drop rate in the LEO satellite network and increases the network's throughput compared to various reference schemes, even under scenarios of increased traffic load and satellite failures.

The rest of the paper is organized as follows. We discuss related work in Sec. II. In Sec. III we introduce the system model and formulate the optimization problem in Sec. IV. SkyLink is presented in Sec. V and we describe the details of our simulator in Sec. VI. The numerical evaluation is presented in Sec. VII and Sec. VIII concludes the paper.

II. Related Work

In this section, we review the related work on link management for LEO satellite networks. Table I summarizes recent advancements in the field. The pioneering work of Gounder et al. introduced a k-shortest path algorithm for satellite networks [19]. Over the past five years, the increasing availability and deployment of large LEO satellite networks have sparked considerable interest within the research community.

Contemporary studies already consider dynamic satellite networks using both ISLs and GSLs, and allowing multi-path routing to distribute traffic [20]–[30]. In these works, link management solutions considering different optimization goals have been investigated. Specifically, the joint optimization of

[2]The code is available under https://github.com/wanjads/SkyLinkSimulator.

delay and packet drop rate is studied in [22]–[25], [28]–[30], delay minimization is considered [26], [27], forwarding overhead reduction is investigated in [18], and [21] maximizes the network efficiency.

Although some of these works consider only relatively small LEO constellations, such as Iridium [21]–[23] or Kepler [24], works like [20], [25] address networks with a couple of hundreds of satellites while [26]–[30] investigate mega LEO constellations. Mega constellations, such as Starlink or Kuiper, consist of thousands of satellites and are designed to deliver global broadband communication to millions of users. However, despite their focus on mega constellations, many studies consider only on a subset of communication paths, typically optimizing pre-defined source-destination pairs rather than network-wide traffic patterns [26]–[29]. Only the authors of [30] consider global broadband traffic. In particular, they investigate the satellite network's resilience through a hierarchical model relying on nearest-neighbor searches for route selection. However, the proposed model relies on a centralized architecture which can lead to congestion under high traffic. In fact, a critical gap in the literature is the lack of approaches that simultaneously adopt a distributed framework and focus on resilience against network failures.

In summary, while impressive progress has been made in routing for LEO satellite networks, challenges remain in achieving scalable and resilient solutions for large constellations.

III. System Model

A. LEO Satellite Networks

We focus on a communication scenario in which the satellites receive data from the users within their coverage area and route it to the internet. As illustrated in Fig. 1, we consider a LEO satellite network consisting of a set $\mathcal{N}$ of N LEO satellites $\mathcal{N} = \{n_1, \dots, n_N\}$ and a set $\mathcal{M}$ of M ground stations $\mathcal{M} = \{m_1, \dots, m_M\}$. The system operates in time slots, starting at $t = 0$ and continuing until a finite time horizon T, with each slot lasting a fixed and constant duration τ.

Satellites establish both, full-duplex optical ISLs [39] and half-duplex radio GSLs and decide, in every time slot, which established ISLs and GSLs to use to relay their incoming data. Considering current technology, we assume that each satellite establishes at most four ISLs to neighboring satellites [40]–[42], two of which are in the same orbit, while the remaining are neighboring satellites in adjacent orbits. Consequently, the ISLs form a +grid as indicated in Fig. 1. Due to irregularities in this grid, pending deployment or failures, the next neighboring satellite in a given direction might not be available for an ISL. In such cases, the satellite does not establish an ISL in that direction. In every time slot, each ground station m_i establishes GSLs to the μ_i closest satellites, where μ_i is the number of antennas at ground station m_i. Satellites use these GSLs to send data to the ground. The ground stations establish connections to the internet via fiber optic links. Both, satellites and ground stations can simultaneously transmit data over their outgoing links and receive data from satellites.

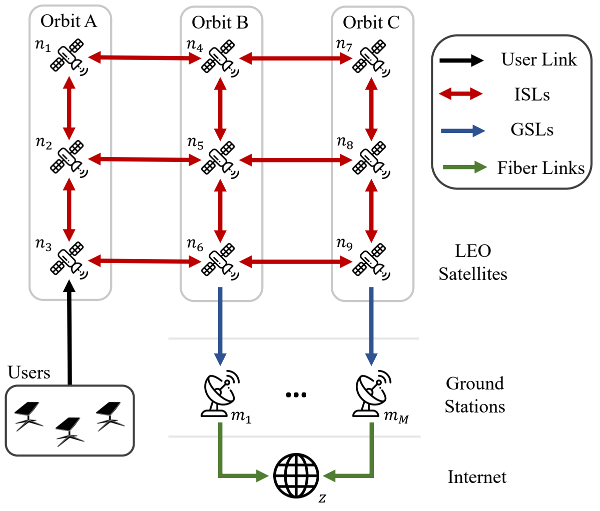


Fig. 1: Diagram of the considered scenario

B. Grid Model

All the links among satellites and ground stations form a directed graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$ at each time slot t. Here, the vertex set $\mathcal{V} = \mathcal{N} \cup \mathcal{M} \cup \{z\}$ contains all satellites, ground stations, and the internet node z. The edge set $\mathcal{E}_t$ includes all links (v, w) between any two nodes $v, w \in \mathcal{V}$ at time slot t. This includes ISLs, GSLs, and the ground stations' links to the internet. For any satellite or ground station v, the set $\mathcal{X}_{v,t}$ contains the paths to the internet. Each path X is a sequence of links $((w_0, w_1), (w_1, w_2), \dots, (w_{n-1}, w_n))$. We call the target node of the last link in a path its *terminal node*. During time slot t, the data transmitted over each path X is modeled as a continuous stream with a fixed rate R_X. The variable $x_{(w,w'),t}$ represents the amount of data that satellite w decides to transmit over the link (w, w') in time slot t. The vector $\mathbf{x}_t$ aggregates the decisions of all the satellites in time slot t and is given by $\mathbf{x}_t = (x_{(v,w),t})_{(v,w)\in\mathcal{E}_t}$. In time slot t, the set of outgoing links at node v is denoted as $O_{v,t} = (\{v\} \times \mathcal{V}) \cap \mathcal{E}_t$.

C. Data Generation Model

We consider a stream-based model for data transmission, where each satellite receives individual data streams and relays them to the internet. The user data rates $R^{\mathrm{g}}_{n,t}$ received at satellite n during time slot t depend on the satellite's location and local time, and are measured in bit/s. Higher rates occur over densely populated regions and peak hours, while lower rates are over remote areas and at night. We divide the day into hourly intervals and the Earth's surface into a one-degree grid. $R^{\mathrm{g}}_{n,t}$ is the sum of data streams of all the cells associated with satellite n as

$$R^{\mathrm{g}}_{n,t} = \mathrm{Pop}_{n,t} \cdot \mathrm{d} \cdot \nu \cdot \mathrm{t}_{n,t}, \tag{1}$$

where in time slot t, $\mathrm{Pop}_{n,t}$ is the population count associated to satellite n, d is the average number of devices per person using the LEO network, ν is the average traffic per device and second and $\mathrm{t}_{n,t}$ is a factor scaling the traffic according to the local time of the day.

Satellites and ground stations have limited data buffers for storing data awaiting transmission, with each node v having a capacity $Q_v^{\max}$. If the outgoing data rate $R_{v,t}^{\text{out}}$ of a node v is less than the incoming data rate $R_{v,t}^{\text{in}}$, the buffer at the receiving node fills up. When the buffer is full, a uniform percentage of data from every incoming stream will be dropped to align the incoming stream size with the outgoing rate ($R_{v,t}^{\text{out}} = R_{v,t}^{\text{in}}$). Our model focuses on the steady state of buffers, either filled or empty, disregarding the transitional phases of filling or emptying. Incoming data at satellites without an outgoing link is dropped.

D. Capacities and Delays

Each link $(v, w) \in \mathcal{E}_t$ has a capacity $C_{(v,w),t}$. If a satellite decides to transmit more data through a link than its capacity allows, i.e. if $x_{(v,w),t} > C_{(v,w),t}$, the excess amount of data is dropped.

The capacity $C_{(v,z),t}$ of the fiber optic link from a ground station v to the internet z is assumed to be constant. To model the capacity $C_{(v,w),t}$ of an ISL between satellites v, $w \in \mathcal{N}$, we first calculate the received power $P_{\text{rx},v,w,t}$ as the transmitted power P_{tx} of satellite v scaled by the pointing loss L_{pointing} and the fraction of the transmitted beam intercepted by satellite w as

$$P_{\text{rx},v,w,t} = P_{\text{tx}} \cdot L_{\text{pointing}} \cdot \frac{\pi\,(0.5 \cdot \varnothing)^2}{\pi(d_{v,w,t} \cdot \theta)^2} \tag{2}$$

where L_{pointing} results from the angular deviation between the transmitting and receiving antennas, $\varnothing$ is the aperture diameter of the receiver at satellite w, $d_{v,w,t}$ is the distance between the satellites, and θ is the beam divergence angle. Note that this model assumes a canonical beam and uniform beam intensity. The noise power is calculated as

$$P_{\sigma,v,w} = k \cdot T_\sigma \cdot B_{\text{ISL}} \tag{3}$$

where k is the Boltzmann constant, T_σ is the noise temperature, and B_{ISL} is the bandwidth of the link between v and w. $C_{(v,w),t}$ is then obtained using the Shannon-Hartley theorem:

$$C_{(v,w),t} = \lambda \cdot B_{\text{ISL}} \cdot \log_2\left(1 + \frac{P_{\text{rx},v,w,t}}{P_{\sigma,v,w}}\right), \tag{4}$$

where, $\lambda < 1$ represents a scaling factor to account for the fact that only part of the link's total capacity can be allocated for data transmission from the user to the internet.

The capacity $C_{(v,w),t}$ for the GSLs between satellite v and ground station w is also modeled using the Shannon-Hartley theorem. In this case, the total received power P_{rx} at the ground station is computed as

$$P_{\text{rx}} = \exp\left(\frac{\log(10) \cdot (\text{EIRP} - \text{FSPL} + G_{\text{rx}} - A_{\text{atmos}})}{10}\right), \tag{5}$$

where EIRP is the equivalent isotropic radiated power transmitted by the satellite, FSPL denotes the free-space path loss, G_{rx} is the gain of the ground station's receiver antenna, and A_{atmos} represents the atmospheric attenuation factor. The free-space path loss is given by

$$\text{FSPL} = 20\log_{10}\left(\frac{4\pi d_{v,w,t} \cdot f_{\text{c}}}{c}\right), \tag{6}$$

where f_{c} is the carrier frequency and c is the speed of light. The atmospheric attenuation A_{atmos} depends on the elevation angle between the satellite and the ground station, with lower elevation angles resulting in greater attenuation due to the longer path through the atmosphere. The noise power $P_{\sigma,\text{v,w}}$ on the GSLs is calculated as

$$P_{\sigma,v,w} = k \cdot T_{\text{sky}} \cdot B_{\text{GSL}}, \tag{7}$$

where B_{GSL} is the bandwidth of the ground-satellite link, and T_{sky} is the sky noise temperature. The sky noise temperature is influenced by atmospheric attenuation and is computed as

$$T_{\text{sky}} = T_{\text{mr}} \cdot \left(1 - 10^{-A_{\text{atmos}}/10}\right) + 2.7 \cdot 10^{-A_{\text{atmos}}/10}, \tag{8}$$

where T_{mr} represents the microwave background radiation temperature.

Here, practical hardware effects can be incorporated by reducing the effective EIRP and/or G_{rx} and/or as an increase of the effective noise temperature. Examples for such effects can be found in the literature (see e.g. [43]). Hence, hardware non-idealities affect capacity through the resulting $\frac{P_{\text{rx}}}{P_\sigma}$ and are implicitly reflected in the GSLs' qualities.

IV. Problem Formulation

Our objective is to find a link management solution that minimizes the average cost given by a weighted sum of delay and drop rate. This formulation ensures that the network operates efficiently by avoiding trade-offs where reducing delay significantly increases data loss or minimizing drop rate causes excessive delays. To rigorously define the problem, we first introduce necessary definitions.

The drop rate $\zeta_{v,t}$ of a node v in time slot t is defined as

$$\zeta_{v,t} = 1 - \tilde{R}_{v,t}, \tag{9}$$

where $\tilde{R}_{v,t} = R_{v,t}^{\text{s}}/R_{v,t}^{\text{g}}$ is the share of traffic that was successfully routed to the internet. $R_{v,t}^{\text{s}}$ is the actual amount of user data traffic successfully routed to the internet and $R_{v,t}^{\text{g}}$ is defined in (1). We do not consider single packets. Drop rates are computed directly at the rate level in our stream-based model. This modeling choice allows us to scale to hundreds of satellites and tens of millions of users while preserving the congestion, buffering, and delay phenomena that drive routing decisions. Data streams are dropped due to low link capacity, whenever they are transmitted in a loop, if they reach a node without outgoing links, or if their delay exceeds the maximum tolerable delay $T_{\max}$.

The propagation delay $D_{(v,w),t}^{\text{Tx}} \in \mathbb{R}^+$ for the link between $v \in \mathcal{N}$ and $w \in \mathcal{N} \cup \mathcal{M}$ is approximated as

$$D_{(v,w),t}^{\text{Tx}} := \frac{d_{v,w,t}}{c}. \tag{10}$$

For simplicity, the propagation delay $D_{v,t}^{\text{Tx}}$ of the fiber optic link between ground station v and the internet is modeled as a random variable drawn from a uniform distribution, with fluctuations introduced by Gaussian noise in each time slot.

While the choice of ground station does influence the actual delay, this effect is beyond the scope of this work.

The queuing delay $D^{\mathrm{q}}_{v,t}$ of node $v \in \mathcal{N} \cup \mathcal{M}$ is the average duration the data remains in the node's data buffer before being forwarded. Assuming a First-In-First-Out queue, $D^{\mathrm{q}}_{v,t}$ is:

$$D^{\mathrm{q}}_{v,t} := \begin{cases} 0 & \text{if } \Delta_C \leq 0 \\ \frac{Q^{\max}_v}{\sum_{(v,w)\in O_{v,t}} \min(x_{(v,w),t}, C_{(v,w),t})} & \text{if } \Delta_C > 0, \end{cases} \tag{11}$$

where $\Delta_C := R^{\mathrm{in}}_{v,t} - \sum_{(v,w)\in O_{v,t}} \min(x_{(v,w),t}, C_{(v,w),t})$.

The delay of a path X with terminal node z is defined as the sum of the propagation delays $D^{\mathrm{Tx}}_{(v,w),t}$ of the links (v,w) in the path and the queuing delays $D^{\mathrm{q}}_{v,t}$ of all intermediate nodes v:

$$D_X := \min\left(\sum\nolimits_{(v,w)\in X} D^{\mathrm{Tx}}_{(v,w),t} + D^{\mathrm{q}}_{v,t}, T_{\max}\right). \tag{12}$$

If the terminal node in path X is not the internet node z or if the path contains a loop, the traffic is dropped and D_X is set to $T_{\max}$. The same applies if the paths' delay exceeds $T_{\max}$.

The cost $c_{v,t}$ at satellite v considers all paths in $\mathcal{X}_{v,t}$ and is calculated as the weighted sum of the delays of each path:

$$c_{v,t} = \frac{\sum_{X\in\mathcal{X}_{v,t}} R_X D_X}{\sum_{X\in\mathcal{X}_{v,t}} R_X}, \tag{13}$$

where R_X is the amount of traffic transmitted over path X.

The average cost c_t of all satellites at t is given by

$$c_t(\mathbf{x}_t) = \frac{\sum_{n\in\mathcal{N}} R^{\mathrm{g}}_{n,t} c_{n,t}}{\sum_{n\in\mathcal{N}} R^{\mathrm{g}}_{n,t}}. \tag{14}$$

As dropped data contributes the highest possible delay of $T_{\max}$ to $c_t(\mathbf{x}_t)$, considering $c_t(\mathbf{x}_t)$ as the optimization target leads to a joint minimization of average delay and drop rate. From the network perspective, $c_t(\mathbf{x}_t)$ can be minimized by solving:

$$\mathbf{x}^*_t = \operatorname*{arg\,min}_{\mathbf{x}_t\in(\mathbb{R}_+)^{|\mathcal{E}_t|}} c_t(\mathbf{x}_t), \quad \forall t = 0, ..., T-1 \tag{15}$$

Solving (15) requires that every satellite has perfect knowledge of $C_{(v,w),t}$ and $R^{\mathrm{g}}_{v,t}$ for all other satellites $v \in \mathcal{N} \cup \mathcal{M}$, an assumption that is hard to fulfill in real systems.

V. SkyLink

In this section, we present SkyLink, our proposed solution for link management in satellite networks. Unlike existing algorithms, SkyLink is fully distributed, scalable, resilient, and does not require perfect knowledge of all $C_{(v,w),t}$ and $R^{\mathrm{g}}_{v,t}$ at every satellite. We begin with an overview of its concept, followed by a detailed description and the pseudocode.

A. Concept

Using SkyLink, each satellite autonomously decides which of its established links to prioritize for data transmission in order to minimize the average delay and drop rate in the network. SkyLink is based on the MAB framework and uses the Upper Confidence Bound (UCB) criterion. The satellite uses the MAB framework to decide which links to use. It evaluates each option using the UCB criterion and selects its next action based on the updated evaluation.

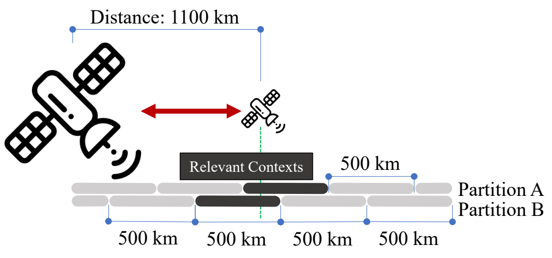


Fig. 2: Visualization of SkyLink's tile-coding mechanism.

The decision $(x_{(v,w),t} \mid w \in O_{v,t})$ of a satellite v represents the distribution of traffic across available links w in $O_{v,t}$. Unlike traditional MAB problems with discrete choices, this decision is continuous. SkyLink addresses this by first creating a ranked list of preferences for the established links. Incoming traffic is then directed through the highest-ranked link on this list. Once its capacity is reached, additional traffic is routed through the next link in the ranking, and so forth. SkyLink additionally adapts its link distribution based on the particular conditions, i.e., the context, of each satellite. We consider that each satellite uses the distances to its neighbors as context. As a result, instead of a single global context, the satellite observes a separate context (i.e., distance) for each link. Such model allows the satellite to evaluate each link separately and based on its specific characteristics. Note however, that the considered context is continuous. Therefore, to maintain a low computational complexity and low learning time, we quantize it into discrete partitions using a tile coding mechanism [38]. Specifically, we divide the continuous contexts into multiple overlapping partitions. Under the assumption that the cost distribution is stationary in each context, our link selection algorithm inherits the convergence properties of the contextual UCB algorithm.

In Fig. 2, we show an example with two partitions, A and B, which represent quantized distance ranges to a neighbor. The satellite discretizes the distances to its neighbors into fixed-length intervals (e.g., 500 km segments; see Fig. 2), with a maximum distance defining the size of each partition. For every neighbor and for each context within these partitions, the satellite learns independently. When evaluating a link during decision-making, the satellite calculates the distance to its neighbors (1100 km in this example), identifies the corresponding tile (highlighted in a darker shade in the figure) within the quantized partition, and aggregates the knowledge associated with these tiles. In the following subsection, we explain central concepts of SkyLink in more detail.

B. Ranking Links

Each satellite uses its experience from previous time slots to rank the established links. For every link (v,w), each satellite stores and updates the average cost $\bar{c}_{v,w}(g, d_{(v,w)})$ experienced when using this connection in each given context and in each partition. The average cost $\bar{c}_{v,w}(g, d_{(v,w)})$ is updated using a running mean. Additionally, each satellite keeps track of the number $n(v, w, g, d_{(v,w)})$ of times the link (v, w) is used in each context. With this information, the satellites are able to evaluate the UCB-criterion for every partition g as

$$\text{UCB}_t^g(v, w) = \bar{c}_{v,w}(g, d_{(v,w)}) - \sqrt{\frac{2\log(t)}{n(v, w, g, d_{(v,w)})}}. \quad (16)$$

The UCB score is then calculated as the average over all scores for each partition:

$$\text{UCB}_t(v, w) = \frac{1}{|\mathcal{G}|} \sum_{g\in\mathcal{G}} \text{UCB}_t^g(v, w), \quad (17)$$

where $\mathcal{G}$ is the set of partitions. Each satellite calculates this UCB score for all its established links. The links are then ranked based on these scores, with lower scores indicating a higher preference. The ranking determines the order in which the satellite routes and distributes incoming traffic. By continuously updating the delays and counts for each context, the satellites dynamically adjust their preferences based on the observed delays.

C. Distributing Traffic

Given the list of preferences, SKYLINK distributes the incoming traffic to its outgoing links following a water-filling pattern. Each satellite first estimates the capacity $C_{(v,w),t}$ of each ISL and GSL link (v, w), as described in Sec. III. To account for uncertainties concerning the capacity, for example, the uncertainty caused by the fact that the atmospheric attenuation A_{atmos} is weather-dependent, we include an uncertainty factor $\sigma < 1$ when determining the link's capacity. By using $\sigma C_{(v,w),t}$, SKYLINK ensures that the capacity of a link is not overestimated, which would result in higher drop rates.

After estimating the link capacities, the decision of satellite v is given as follows. For an incoming data rate $R^{\text{in}}_{v,t}$ at time slot t, let the list of preferences for its established links be given by $(w_1, ..., w_k)$, $k = |O_{v,t}|$. Now assume that for some $i < k$

$$\sum_{j=1}^{i} \sigma C_{(v,w_j),t} < R^{\text{in}}_{v,t},$$

and

$$\sum_{j=1}^{i+1} \sigma C_{(v,w_j),t} \geq R^{\text{in}}_{v,t}.$$

Then the full capacity of the first i links is used, i.e.,

$$x_{(v,w_j),t} = \sigma C_j, \quad j = 1, ..., i. \quad (18)$$

For link $i + 1$, SKYLINK sets:

$$x_{(v,w_{i+1}),t} = R^{\text{in}}_{v,t} - \sum_{j=1}^{i} \sigma C_j. \quad (19)$$

Within any used outgoing link, capacity is divided proportionally among all incoming streams. Through this flow-level proportional sharing, we approximate the behavior of packet schedulers. The remaining links $j = i + 2, ..., k$ are not used and $x_{(v,w_j),t} = 0$. If the incoming traffic exceeds the sum of the capacity of all established links, each link is used to its full capacity and the remaining data in the buffer is dropped.

Algorithm 1 SKYLINK at satellite v in time slot t

```
1:  for $w_i \in O_{v,t}$ do:                       ▷ Calculate UCB-value, Eq. (17)
2:      for $g \in \mathcal{G}$ do:
3:          $\text{UCB}_t^g(v, w_i) = \bar{c}_{v,w_i}(g, d_{(v,w_i)}) - \sqrt{\frac{2\log(t)}{n(v,w_i,g,d_{(v,w_i)})}}$
4:      end for
5:      $\text{UCB}_t(v, w_i) = \frac{1}{|\mathcal{G}|}\sum_{g\in\mathcal{G}} \text{UCB}_t^g(v, w_i)$
6:  end for
7:  Sort neighbors $(w_i)_{i=1}^k$ in ascending order of $\text{UCB}_t(v, w_i)$
8:  for $i = 1$ to $k$ do                          ▷ Allocate flow, Eqs. (18), (19)
9:      if $\sum_{j=1}^{i} C_j < R^{\text{in}}_{v,t}$ then
10:         $x_{(v,w_i),t} \leftarrow C_i$
11:     else
12:         $x_{(v,w_i),t} \leftarrow R^{\text{in}}_{v,t} - \sum_{j=1}^{i-1} C_j$
13:         break
14:     end if
15: end for
16: Observe cost $c_{v,t}$
17: for $g \in \mathcal{G}$ do:
18:     $n \leftarrow n(v, w_1, g, d_{(v,w_1)})$
19:     $\bar{c}_{v,w_1}(g, d_{(v,w_1)}) \leftarrow \frac{n\cdot\bar{c}_{v,w_1}(g,d_{(v,w_1)})+c_{v,t}}{n+1}$
20: end for
21: $n(v, w_1, g, d_{(v,w_1)}) \leftarrow n(v, w_1, g, d_{(v,w_1)}) + 1$
```

D. Implementation

We describe how SKYLINK is implemented by explaining the pseudo-code shown in Alg. 1. Note that the pseudo-code is provided for a single satellite v at time slot t. Each of the satellites simultaneously executes this algorithm.

In every time slot t, SKYLINK observes the satellite's neighbors $(w_i)_{i=1}^k$ to which v is connected either by an ISLs or by a GSLs and calculates the UCB-scores for each link using tile coding as described in Eq. 17 (lines 1-6). Based on these scores, SKYLINK creates the list of preferences (line 7). Using this list, v decides on the amount of traffic it relays through each of its links according to the water-filling mechanism described in Eq. 18 and Eq. 19 (lines 8-15).

Finally, SKYLINK observes the delay resulting from its decision (line 16) and updates the counter and the average delay of its preferred link (lines 17-21).

We based our implementation of SKYLINK on distances as contexts, because experiments showed that, among all tested features, per-link distance has the most direct and strongest influence on routing performance, outperforming any single alternative and all feature combinations. Beyond per-link distance, we evaluated local and UTC time, satellite load, neighbor distance, total path distance, and their combinations as contexts.

E. Computational Complexity

We analyze the time complexity of SKYLINK per satellite and per time slot. Here, $k = |O_{v,t}|$ denotes the number of established outgoing neighbors (ISLs and GSLs), and $|\mathcal{G}|$ is the number of tile-coding grids. For each neighbor and each grid, SKYLINK performs a constant-time update. Each satellite establishes at most four ISLs to neighboring satellites. Additional neighbors are the visible ground stations with which GSLs are established. Hence $k \leq 4 + |\text{visible GSLs at } t|$ and is independent of constellation size. Ranking the k neighbors is in $\mathcal{O}(k \log k)$. The water-filling loop performs at most $\sum_{i=1}^{k} i = \mathcal{O}(k^2)$ operations. With prefix sums, this reduces to $\mathcal{O}(k)$. Finally, updating the running means is linear in the number of grids. Overall, per satellite and time slot, the time complexity of SKYLINK is in $\mathcal{O}(k\,|\mathcal{G}| + k \log k)$. This makes the complexity independent of the number of satellites and users because k is bounded by visibility of ground stations rather than constellation size or demand.

VI. LEO Network Simulator

Existing simulation tools for the evaluation of link management in LEO satellite networks are usually limited to network segments (see e.g. [14], [22]). Therefore, to validate our proposed approach at a global scale and support the advancement of LEO satellite networks, we present a novel simulation framework. Within our simulator, we make use of CosmicBeats [44] to pre-calculate the exact position of each ground stations and each satellite in every time slot over the time horizon. To perform these calculations, CosmicBeats uses a list of coordinates for each ground station, Two-Line Element (TLE)-data for each satellite in the considered constellation, a time slot duration τ and a time frame including a starting time and a number of time slots T. We use a flow-level (stream-based) simulator. As described in Sec. III, user data are represented as continuous rates per time slot. Individual packets are not instantiated. Based on this information, our simulator enables global-scale evaluation of link management approaches by combining the following components, which we describe in detail in the following.

Grid Construction For each satellite, we pre-calculate its direct neighboring satellites in the constellation grid and the visible ground stations it can connect to. Using the satellite's location information from CosmicBeats, the simulator builds a grid containing at most four close, visible neighbor satellites (one in each cardinal direction), as well as the visible ground stations in each time slot. The grid is limited to four neighbors to account for the number of available ISL links.

Channel Models Using the satellites' positions and ground stations' locations, the simulator determines, in each time slot, the channel conditions based on the models described in Sec. III. With this information, the capacities $C_{(v,w),t}$ of the ISLs and GSLs are calculated. Note that we pre-calculate the atmospheric attenuation A_{atmos} for different angles between satellites and ground stations.

Data Generator The simulator uses a global population dataset [45] to assign each satellite an amount of traffic proportional to the population living close to its position. The amount of traffic is then calculated as detailed in Eq. (1).

Data Stream Transmission Using the generated network grid, its channel properties, and the incoming rates, we simulate the transmission of data streams in the network. As a result, we obtain the experienced delay, drop rate, and throughput per satellite, as well as network metrics like total network throughput and average number of hops per path.

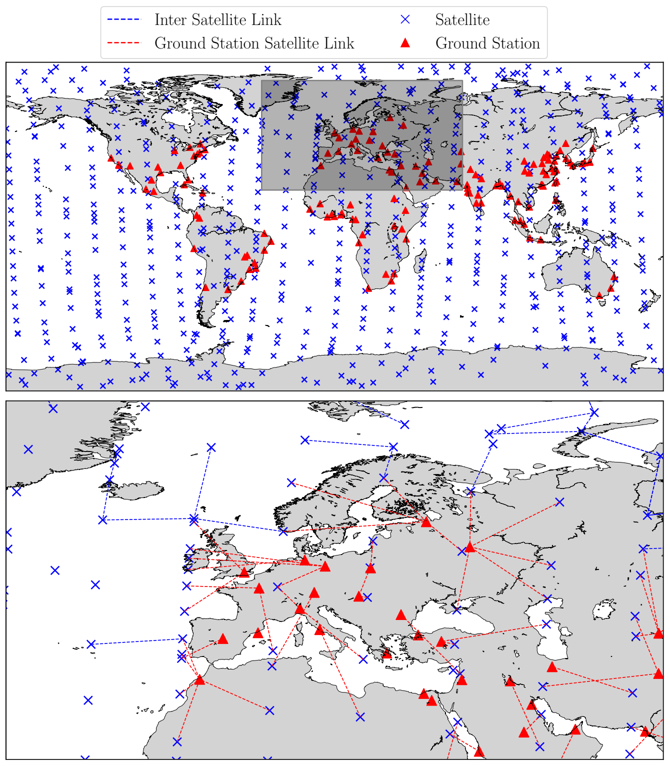


Fig. 3: The simulated network including OneWeb satellites and ground stations. The section at the bottom is highlighted in the top map and includes GSLs and ISLs carrying traffic.

Visualization Our simulator offers multiple options for visualizing both the network structure and the performance of various routing strategies. An example of a simulated network visualization is shown in Fig. 3. The upper portion depicts the entire OneWeb LEO network as of September 26, 2023, at 08:51:00 UTC. In the lower portion, a zoomed-in section of the network is presented, additionally including ISLs and GSLs that are actively carrying traffic.

VII. Evaluation

In this section, we evaluate SKYLINK and compare its performance to reference schemes using data from OneWeb's LEO satellite network. We assess the system's performance across various operational scenarios and conditions. We first explain the setup and the used parameters. Second, we introduce the reference schemes. Third, we demonstrate its scalability by evaluating performance across varying numbers of users. We then show SKYLINK's resilience by analyzing the impact of network failures. Finally, we present a short discussion on the parameter optimization of SKYLINK.

Setup and Parameters: To evaluate the performance of SKYLINK and to compare it to the performance of reference schemes, we employ our simulator described in Sec. VI. Table II provides an overview of the simulation parameters. We repeated all experiments for $R = 100$ times and present an

Parameter Description	**Symbol**	**Value**
Number of repetitions	R	100
Number of satellites	N	636
Number of ground stations	M	146
Number of time steps	T	40320
Time step duration	τ	15s
TTL	T_{max}	200ms
Simulated time frame start	-	28.09.2023 08:26 UTC
Number of users	-	25.4M
Average number of devices per person for 25.4M users	d	0.003175
Average upload traffic per device	ν	22.98 kbps
Data buffer size for ground station m	$Q_m^{\max}$	1GB
Data buffer size for sat. n	$Q_n^{\max}$	50MB
Capacity of a ground station's link to internet	C_v	50 Gbps
Delay between a ground station m and the internet	D_m^{tx}	1-5 ms
Bandwidth for ISLs	B_{ISL}	5 GHz
Aperture diameter of ISL receivers	$\varnothing$	10 cm
ISL beam divergence angle	θ	$1.744 \cdot 10^{-5}$ Rad
Pointing loss factor	L_{pointing}	0.9
Noise temperature ISLs	T_{noise}	290 K
ISLs transmit power	P_{tx}	0.1 W
Upload scale factor	λ	0.08
Bandwidth for GSLs	B_{GSL}	250 MHz
Equivalent isotropic radiated power	EIRP	34.6 dbW
Ground station receiver antenna gain	G_{rx}	10.8 db
GSL carrier frequency	f_{c}	19 GHz
Microwave background radiation temperature	T_{mr}	275 K

TABLE II: Simulation Parameters

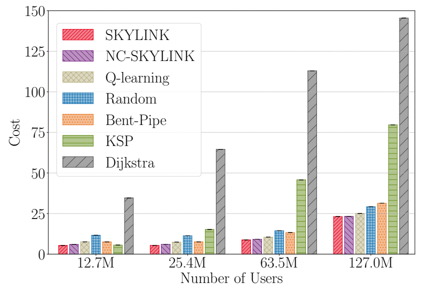
average over these repetitions. We use OneWeb's LEO satellite network with a near-polar Walker Star configuration. Ground stations are placed in the world's $M = 146$ largest cities, which reflects the current scale of network infrastructures [46]. For the simulated time frame, $N = 636$ OneWeb satellites were operational. OneWeb's TLE data is available in [47]. To generate the incoming data, we use the global population dataset in [45]. We assume 25 million users globally out of 8 billion people (i.e., $\mathrm{d} = 3.175 \cdot 10^{-3}$), reflecting the high annual growth and recent statistics. We estimate the annual amount of generated traffic as 6 Zettabytes [48], where upload traffic accounts for 8% of the total traffic [49]. Dividing by the number of seconds per year and a total of 5.3 billion fixed and mobile broadband network users [50], we set $\nu = 22.98$ kbps. The generated traffic per second varies depending on the local time. $\mathrm{t}_{n,t}$ is adjusted every hour, matching a characteristic daily pattern observed from [51].

Reference Schemes: To ensure a fair comparison, we include only baselines whose signaling overhead is not higher than SkyLink 's fully distributed design, which uses only locally available information. We exclude schemes that require exchanging state information or a central controller. Concretely, we compare SkyLink to five reference schemes, each highlighted with a distinct color in the upcoming figures for clarity: (i) a bent-pipe solution that sorts only the established GSLs randomly for its preference list (orange), (ii) a solution based on Dijkstra's algorithm that considers only the shortest path to the ground (gray), and (iii) a solution using not just the shortest, but the k-shortest paths to the ground [19], with $k = 4$ (green). (iv) A random solution that sorts the established links randomly to generate its preference list (blue), (v) A solution based on distributed Q-learning as proposed in [24] (beige).

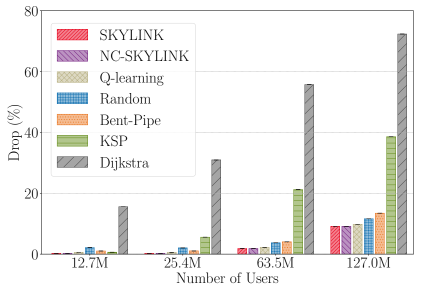
In addition, we evaluate SkyLink against a variant, non-contextual SkyLink (NC-SkyLink), which omits both contextualization and the tile-coding mechanism, and is represented in purple.

A. Scalability

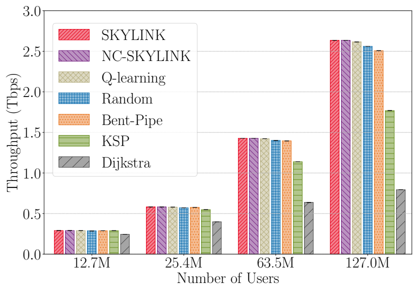
We begin by demonstrating the scalability of SkyLink. In Fig. 4, we present the results of SkyLink and the reference schemes for 12.7, 25.4, 63.5, and 127.0 million users respectively for different metrics. Anchoring to publicly reported Starlink adoption, we fit a simple exponential and use the projected January 2026 levels (12.7 million users) as a realistic upper bound for a constellations near-term user base. We then use its 1×, 2×, 5×, and 10× multiples to test scalability. The standard deviations across all metrics are small, consistently below 1% of the average value. As a result, while error bars are included in the graphs, they remain barely noticeable. In Fig. 4a, we show the average cost, which is the sum of delivered traffic weighted by its respective delay and the drop rate weighted by $T_{\max}$ as in Eq. (14). In every scenario, SkyLink generates the lowest cost and outperforms distributed Q-learning, the random solution, the bent-pipe approach, and the algorithms based on shortest paths. The most significant improvement can be observed compared to Dijkstra's algorithm. Dijkstra's algorithm only uses the shortest path from the satellite back to the ground. This results in frequent congestion of GSLs, higher drops and consequently in higher cost. When using not only one but k shortest paths, this effect is dampened. For 12.7 million users, the k-shortest paths algorithm is still performing close to SkyLink. However, with an increasing number of users, it shows that k-shortest paths suffers from the same inefficiencies as Dijkstra's algorithm. A similar effect can be observed for the bent-pipe solution. For 12.7 million users, SkyLink reduces the cost by 5.0% compared to k-shortest paths, 11.3% compared to NC-SkyLink, 29.5% compared to the bent-pipe



(a) Average Cost.

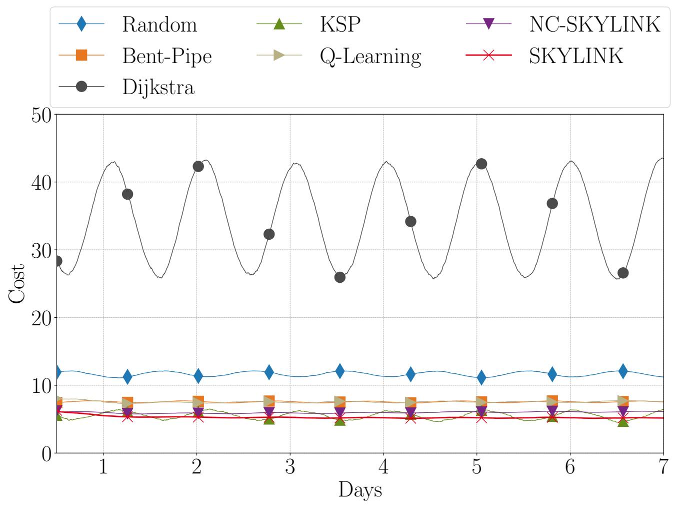


(b) Average Drop Rate.

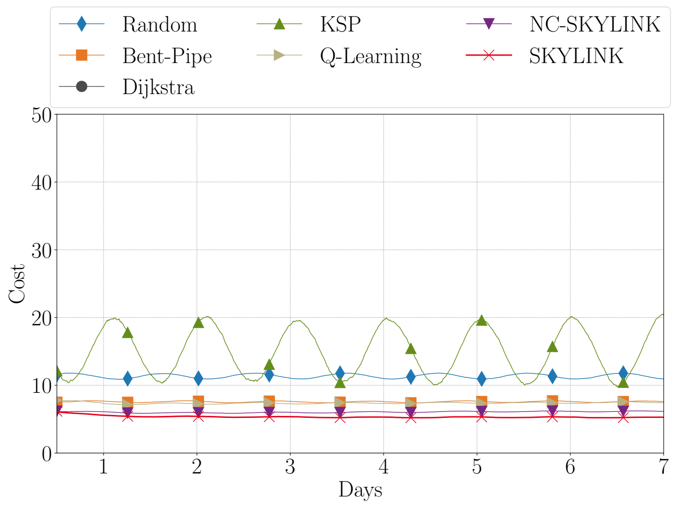


(c) Total Throughput.

Fig. 4: Comparison of SkyLink and the reference schemes for different metrics and different numbers of users.



(a) Cost over time for 12.7 million users.



(b) Cost over time for 25.4 million users.

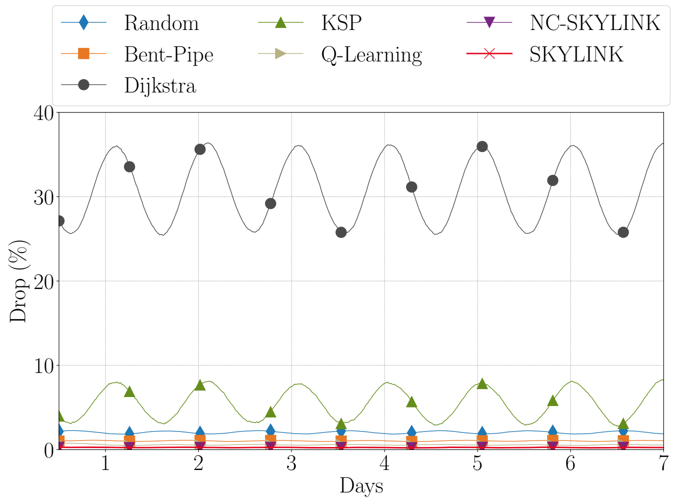
Fig. 5: Evolution of cost over a week for different user scales.

solution, 29.8% compared to distributed Q-learning, 54.4% compared to the random approach, and 84.6% compared to Dijkstra.

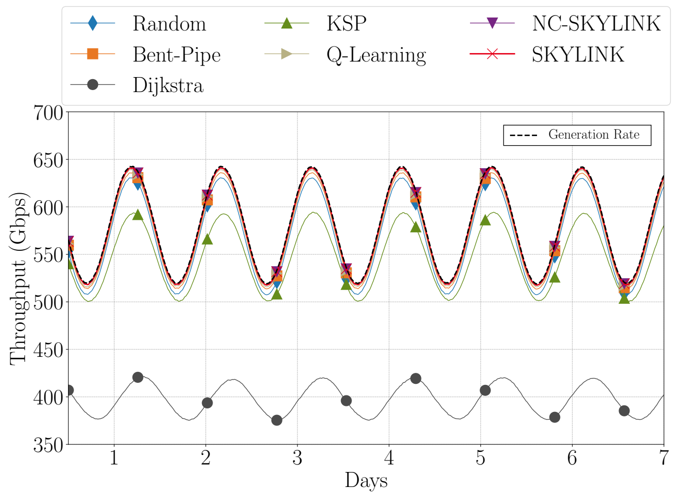
As the number of users, and consequently the data volume, increases, some GSLs become overloaded. The random solution performs comparably well because it actively utilizes the available ISLs, leading to a more balanced distribution of traffic across multiple GSLs. However, since the ISLs are used randomly rather than strategically, this approach reduces the drop rate but increases the delay, affecting the overall cost negatively. The improvement of SkyLink compared to NC-SkyLink is small but constant. The reason for this is that NC-SkyLink uses the same algorithm which is also fully distributed and learns likewise. The improvement is hence solely based on the additional tile-coding mechanism. In terms of cost and for 25.4 million users, SkyLink improves Dijkstra by 91.7%, k-shortest path by 64.5%, the random solution by 52.5%, distributed Q-learning by 27.3%, and the bent-pipe approach by 28.7%. Using contextualization and tile-coding improves SkyLink by 11.1% compared to NC-SkyLink.

We present the average drop rate and the total network throughput in Fig. 4b and Fig. 4c, respectively. Clearly, using only the shortest path results in higher drop rates that increase rapidly with the number of users. Although the k-shortest paths approach initially achieves a low drop rate, it increases even faster than that for Dijkstra's algorithm as the number of users grows. From 12.7 to 127 million users, the drop rate for Dijkstra increases from 15.6% to 72.4%, while for k-shortest paths, it rises from 0.6% to 38.6%. Consequently, the total network throughput of shortest-path algorithms falls significantly behind that of SkyLink. For 25.4 million users, SkyLink achieves a 95.4% reduction in drop rate compared to k-shortest paths and 99.2% compared to Dijkstra. It also improves over distributed Q-learning, the bent-pipe approach and the random solution by 56.1%, 75.6% and 87.6%, respectively. However, both NC-SkyLink and SkyLink achieve a drop rate of 0.3%, showing that the improvements in cost are due to the fact that SkyLink achieves a lower average delay than NC-SkyLink. In terms of throughput, this translates to SkyLink delivering 45.8% higher throughput than Dijkstra and 6.0% more than k-shortest paths.

To analyze the performance of SkyLink and the reference schemes in greater detail, we not only examine aggregate metrics over the entire simulation period but also investigate how these metrics evolve over the course of a week. Note that we display running means over half a day, which is why



(a) Drop rate over time.



(b) Throughput over time.

Fig. 6: Evolution of cost and throughput over a week for 25.4 million users.

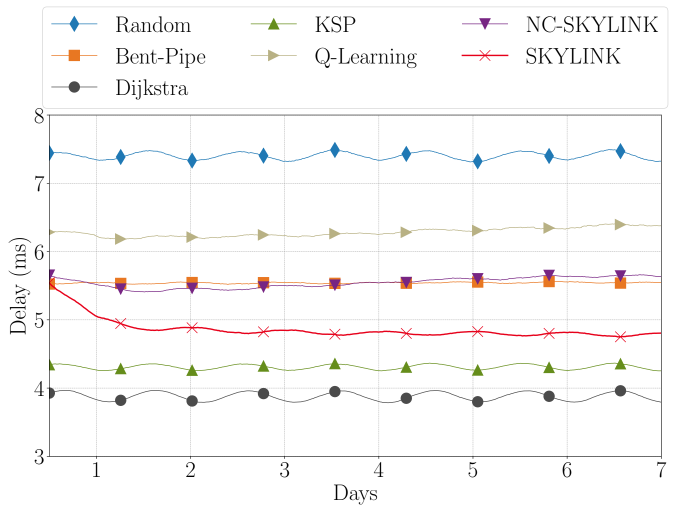
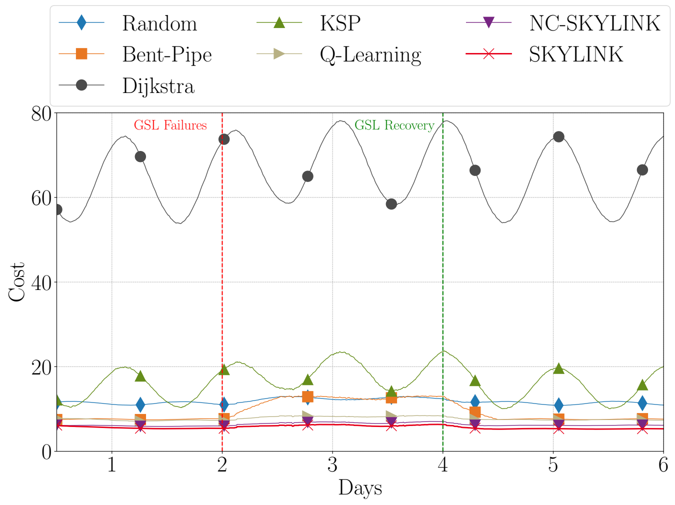


Fig. 7: Evolution of delay over a week for 25.4 million users.

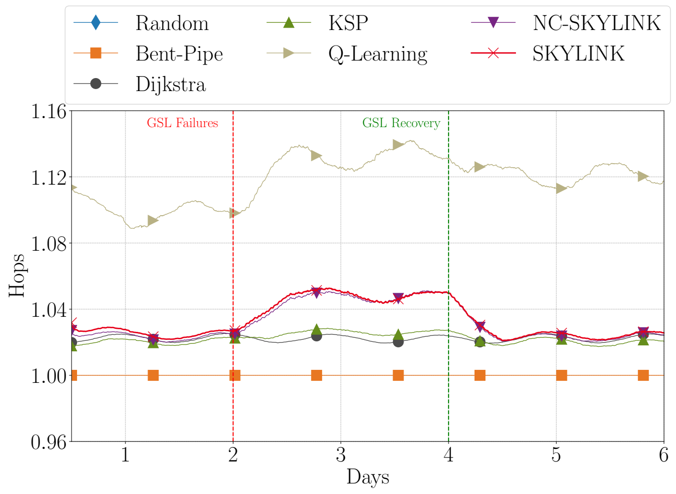
the horizontal axis starts at 0.5 days. The comparison of costs over time for 12.7 million users in Fig. 5a and 25.4 million users in Fig. 5b provides three major insights: (a) shortest-path algorithms lack scalability, (b) SKYLINK has a learning phase and converges within days, and (c) SKYLINK is resilient to fluctuations affecting other schemes. Most strikingly, the lack of scalability in shortest-path algorithms is evident. For 12.7 million users, both Dijkstra and k-shortest paths exhibit significant cost fluctuations throughout the day, which intensify as the network becomes more congested with 25.4 million users. Additionally, SKYLINK undergoes a learning phase during the initial days, fully stabilizing within a week. Finally, SKYLINK demonstrates resilience against the fluctuations that affect all other reference schemes, maintaining consistent performance even under varying network loads. Note that for 25.4 million users, Dijkstra consistently incurs a cost greater than 50 throughout the entire period, which is why it is not visible in Fig. 5b.

In Fig. 6, we show the evolution of drop rate and throughput over a week for 25.4 million users. As for the cost, in Fig. 6a, we observe a daily pattern in the drop rate resulting from the fluctuating amount of data in the network over the course of a day. The fluctuation is higher for shortest path algorithms and negligible for SKYLINK. In Fig. 6b, we additionally included the data generation rate in the network. As a result of the low drop rate, SKYLINK's total throughput is close to this generation rate. At the same time, the gap between throughput and generation rate, especially at times with high network loads is clearly visible for the reference schemes. For the drop rate, improvements achieved by SKYLINK are 56.0% compared to distributed Q-learning, 75.5% compared to Bent-Pipe, 87.5% compared to Random, 95.4% compared to k-shortest paths, and 99.2% compared to Dijkstra. For throughput, SKYLINK shows an improvement below 1.0% compared to distributed Q-learning and Bent-Pipe because these strategies are able to successfully deliver almost the entire traffic. This improvement in throughput grows to 1.8% compared to Random, 6.0% compared to k-shortest paths, and 45.9% compared to Dijkstra.

As mentioned earlier, the advantage of SKYLINK compared to NC-SKYLINK is based on achieving a lower delay of delivered data. To analyze this further, we present the average delay of successfully delivered data for SKYLINK and the reference schemes in Fig. 7. In particular, shortest-path algorithms, such as Dijkstra and k-shortest paths, exhibit relatively low delays. However, this is primarily because these algorithms deliver fewer data in general, favoring data that are closer to the ground. In contrast, SKYLINK delivers more data, which would be dropped by the shortest path algorithms. Both SKYLINK and NC-SKYLINK undergo a learning phase during the first two days. SKYLINK maintains per-link estimates conditioned on the current geometric context via distance-based tile coding, whereas NC-SKYLINK collapses experience across all contexts into a single global estimate. Since the relative geometry of satellites and visible ground stations

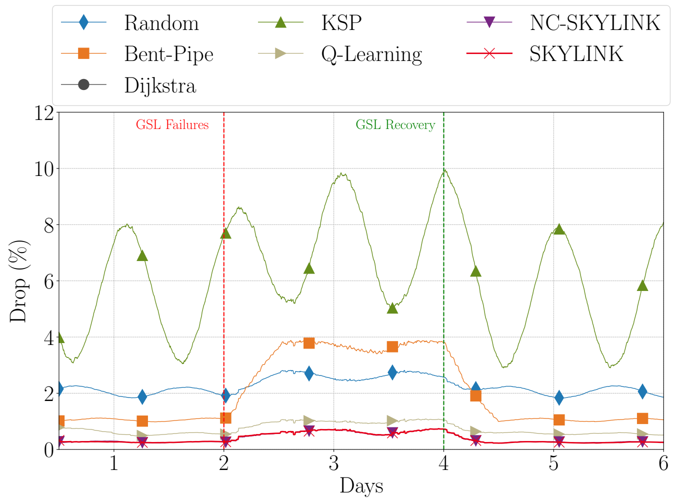


(a) Cost over time.

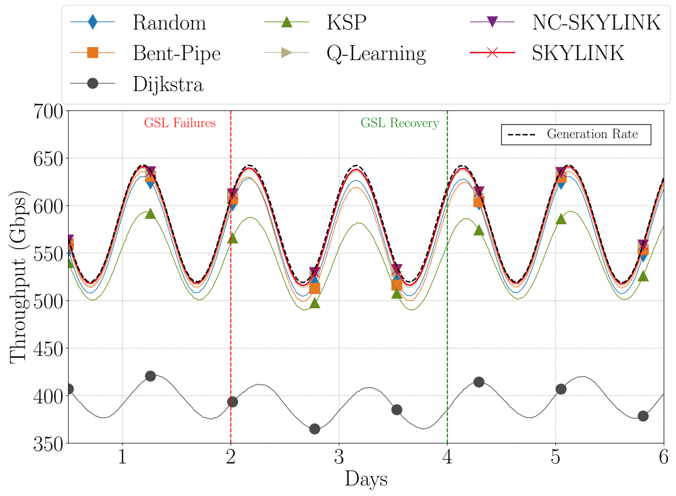


(b) Average number of hops over time.

Fig. 8: Evolution of cost and average hops over a week for 25.4 million users and under GSL-failures.



(a) Drop rate over time.



(b) Throughput over time.

Fig. 9: Evolution of drop rate and throughput over a week for 25.4 million users and under GSL-failures.

continuously drifts with orbital motion, SkyLink adapts its ranking to the present context, while NC-SkyLink keeps averaging over mismatched situations. This leads to a gradual unlearning with a higher delay, as visible in Fig. 7.

B. Resilience

In the previous subsection we demonstrate that SkyLink is resilient to daily fluctuations of network traffic. In this subsection, we focus on SkyLink's ability to maintain its superior performance even under network failures. We restrict our analysis to GSLs failures because they have the higher impact. Our simulations show that, even under a very unlikely scenario in which 50% of satellites lose all ISLs, every strategy faces cost increases of less than 10%. This is because the capacity of the GSLs is the network's bottleneck, as the following results demonstrate.

In Fig. 8a, we present the cost evolution over six days, during which 3% of the satellites experience GSLs outages starting on the third day. Such failures commonly appear in LEO satellite networks [30]. The challenge for the routing strategies is to detect this disruption and adapt by rerouting traffic through the GSLs of unaffected satellites. On the fifth day, the network is restored to its normal state, providing an opportunity to evaluate how effectively the approaches return to regular operation. As expected, all the reference schemes experience increased costs during the third and fourth day, when network failures persist. This increase is 67.6% for the bent-pipe strategy, which heavily relies on stable GSLs and still 23.2% for k-shortest paths. This increase is due to their inability to efficiently adapt to the reduced availability of GSLs, leading to higher congestion. In contrast, SkyLink demonstrates its resilience by quickly rerouting traffic through unaffected satellites, minimizing the impact on overall costs,

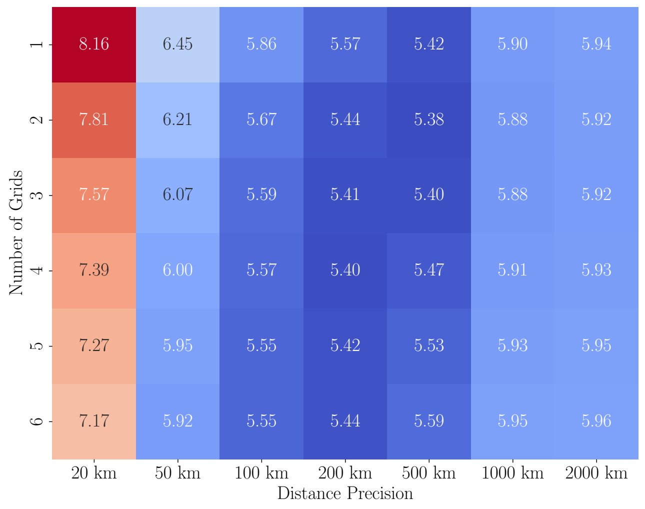


Fig. 10: Average cost of SKYLINK for different parameters.

which increases less than 10%.

The effect of rerouting data is illustrated in Fig. 8b by analyzing the average number of hops taken by data traversing the network. Notably, the majority of data is received in high-population areas near ground stations, allowing for direct transmission to the ground. As a result, the average number of hops is close to 1 for all strategies, and for the bent-pipe strategy, it remains consistently at exactly 1. When network failures occur, the average number of hops for SKYLINK and NC-SKYLINK increases significantly compared to the other strategies. The gradual increase seen in the figure is an artifact of the running mean. This shows that SKYLINK quickly detects failures and promptly reroutes traffic, effectively mitigating the impact on overall costs. A similar effect can be observed for distributed Q-learning with the significant difference that distributed Q-learning is not able to directly return to normal operation as soon as the network recovers. In Fig. 9, we analyze the drop rate and throughput during the failure scenario. As expected, the drop rate increases for all reference schemes during the period of GSLs outages. However, SKYLINK maintains a significantly lower drop rate compared to the other strategies. While the drop rate of k-shortest path increases from 5.7% to 7.5% and the drop rate of the bent-pipe strategy from 1.1% to 3.7%, SKYLINK's average drop rate does not exceed 0.7% even during the failures on the third and forth day. Note that Dijkstra's drop rate consistently exceeds 12%, which is why it is not included in Fig. 9a. Correspondingly, the throughput for the reference schemes declines during the failure period, reflecting their inability to manage the rerouting efficiently. In contrast, SKYLINK sustains a throughput that remains close to the data generation rate, showing its resilience also under network failures.

C. Parameter Optimization

SKYLINK uses a tile-coding mechanism described in Sec. V, which relies on two main parameters: The distance precision and the number of discretizations, which we explain in the following. The continuous context space, defined by the distances to a satellite's neighbors, needs to be quantized. The precision or granularity of this quantization involves a trade-off: if the granularity is too low, SKYLINK fails to sufficiently distinguish between different contexts, resulting in a general solution that largely ignores the distance to neighbors. Conversely, if the granularity is too high, the number of samples per context becomes too low, making it harder for SKYLINK to learn effectively. The second parameter pertains to the number of overlapping partitions used in the tile-coding mechanism. This parameter also presents a trade-off: a single partition leads to overly sharp transitions between contexts, making the model overly sensitive to small changes. In contrast, too many partitions blur the distinctions between contexts, potentially masking important variations.

Both trade-offs are visualized in Fig. 10. The tested granularity for the distance quantization ranges from 20 km to 2000 km, and the tested number of partitions ranges from 1 to 6. Each tile is labeled with the average cost of SKYLINK over a period of 7 simulated days and is colored using a heat map scheme based on these values. Concerning the distance precision, it is clearly visible that 20–50 km are too granular, resulting in a low number of samples per context, while 1000–2000 km are too coarse, leading to a lack of differentiation between contexts and a generalization that overlooks variations. The lowest average cost is obtained for a precision of 500 km. The 500 km quantization renders SKYLINK robust to errors in the estimation of the position of satellites and ground stations. Small deviations rarely change the active tile and therefore have negligible impact on the chosen actions. The impact of the number of partitions is smaller compared to the impact of distance precision; however, for higher granularities, it becomes increasingly important to use more partitions to avoid sharp transitions between contexts. In our evaluation, the lowest average cost was achieved using 2 partitions. Based on these findings, we select 500 km for the distance precision and 2 partitions as the parameters for SKYLINK.

Further experiments showed that other possible contexts such as the data load at the satellite, local time of the day, UTC, or the satellite's location do not improve the performance compared to SKYLINK using the distance to its neighbors as per-arm context. Likewise, adding these contexts to the distance to form a larger context space does not improve the performance. Most probably, the direct influence of the distance to its neighbors on link capacity and link delay are the cause for its relevance during learning.

VIII. CONCLUSION

In this work, we propose SKYLINK, a distributed, scalable and resilient learning approach to minimize delay and drop rate in LEO satellite networks. Considering global traffic, SKYLINK selects ISLs and GSLs in each time slot to route user data to the internet. To address challenges like the dynamic topology of LEO satellite networks, global-scale traffic, potential network failures, and routing complexity, SKYLINK uses a contextualized MAB solution, learning link preferences based on relative distances to satellites' neighbors. It employs tile coding and the UCB criterion for effective generalization over multiple contexts. We evaluate SKYLINK using a new simulator for global-scale simulations of stream-based data traffic. Extensive simulations show SKYLINK outperforms reference schemes in delay, drop rate, and throughput, even under high traffic and satellite outages.

While our evaluation targets LEO, SKYLINK's graph-based formulation, together with its context-aware link ranking and water-filling mechanism, is not restricted to LEO. Hybrid LEO–MEO–GEO and airborne operation can be realized by enlarging the node/link sets and augmenting the per-link context. In future work, we will scale our experiments to larger constellations (e.g., Starlink), incorporate alternative Quality-of-Service objectives, and integrate MEO and GEO satellites as well as airborne relays to evaluate the scalability and resilience benefits of multi-layer deployments. Additionally, a quantitative comparison of SKYLINK's energy consumption with fixed-path routing and centralized approaches is an interesting direction. We expect SKYLINK's per-satellite energy requirements to be low given its extremely small time complexity and its local-information design.

REFERENCES

[1] D. Chow, "To cheaply go: How falling launch costs fueled a thriving economy in orbit," https://www.nbcnews.com/science/space/space-launch-costs-growing-business-industry-rcna23488, 2022, accessed: 2-March-2024.

[2] J. Sanpera, "In the past, a satellite used to cost 250 million euros; now, a million, but being in space is expensive," https://sateliot.space/en/news-sateliot-space/in-the-past-a-satellite-used-to-cost-250-million-euros;-now-a-million-but-being-in-space-is-expensive/, 2023, accessed: 2-March-2024.

[3] J. Matthews, "The decline of commercial space launch costs," https://www2.deloitte.com/us/en/pages/public-sector/articles/commercial-space-launch-cost.html, 2018, accessed: 2-March-2024.

[4] C. Davenport, "The revolution in satellite technology means there are swarms of spacecraft no bigger than a loaf of bread in orbit," https://www.washingtonpost.com/technology/2021/04/06/small-satellites-growth-space/, 2021, accessed: 2-March-2024.

[5] Kongsberg NanoAvionics, "How much do cubesats and smallsats cost?" https://nanoavionics.com/blog/how-much-do-cubesats-and-smallsats-cost/, 2023, accessed: 2-March-2024.

[6] Morgan Stanley, "Space: Investing in the final frontier," https://www.morganstanley.com/ideas/investing-in-space, 2020, accessed: 2-March-2024.

[7] Acumen Research and Consulting, "Satellite internet market size is expected to reach at USD 17,431 million by 2030, registering a CAGR of 18.2%," https://finance.yahoo.com/news/satellite-internet-market-size-expected-150000823.html, 2022, accessed: 2-March-2024.

[8] Straits Research, "Global space-based broadband internet market grows at a staggering CAGR of 20.64%," https://straitsresearch.com/press-release/global-space-based-broadband-internet-market-report, 2023, accessed: 2-March-2024.

[9] BIS Research, "Space-based broadband internet market - a global and regional analysis," https://bisresearch.com/industry-report/space-based-broadband-internet-market.html, 2021, accessed: 2-March-2024.

[10] S. Ma, Y. Chou, H. Zhao, C. Long, X. Ma, and J. Liu, "Network characteristics of LEO satellite constellations: A starlink-based measurement from end users," in *IEEE INFOCOM*, 2023.

[11] M. Kan, "Starlink's laser system is beaming 42 million GB of data per day," https://www.pcmag.com/news/starlinks-laser-system-is-beaming-42-million-gb-of-data-per-day, 2024, accessed: 2-March-2024.

[12] Amazon Staff, "Amazon's project kuiper completes successful tests of optical mesh network in low earth orbit," https://www.aboutamazon.com/news/innovation-at-amazon/amazon-project-kuiper-oisl-space-laser-december-2023-update, 2023, accessed: 2-March-2024.

[13] M. Motzigemba, H. Zech, and P. Biller, "Optical inter satellite links for broadband networks," in *2019 9th International Conference on Recent Advances in Space Technologies (RAST)*, 2019.

[14] M. M. H. Roth, A. Hegde, T. Delamotte, and A. Knopp, "Shaping rewards, shaping routes: On multi-agent deep q-networks for routing in satellite constellation networks," 2024. [Online]. Available: https://arxiv.org/abs/2408.01979

[15] X. Zhou, Y. Weng, B. Mao, J. Liu, and N. Kato, "Intelligent multi-objective routing for future ultra-dense LEO satellite networks," *IEEE Wireless Communications*, 2024.

[16] R. Wang, M. A. Kishk, and M.-S. Alouini, "Stochastic geometry-based low latency routing in massive LEO satellite networks," *IEEE Transactions on Aerospace and Electronic Systems*, 2022.

[17] H. Liming, K. Shaoli, S. Shaohui, M. Deshan, H. Bo, and Z. Meiting, "A load balancing routing method based on real time traffic in LEO satellite constellation space networks," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, 2022.

[18] H. Zhang, Z. Wang, S. Zhang, Q. Meng, and H. Luo, "Optimizing link-identified forwarding framework in LEO satellite networks," in *2023 21st International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2023.

[19] V. Gounder, R. Prakash, and H. Abu-Amara, "Routing in LEO-based satellite networks," in *1999 IEEE Emerging Technologies Symposium. Wireless Communications and Systems*, 1999.

[20] T. Huang, Z. Fang, Q. Tang, R. Xie, T. Chen, R. Zhang, and F. R. Yu, "Integrated computing and networking for LEO satellite mega-constellations: Architecture, challenges and open issues," *IEEE Wireless Communications*, 2024.

[21] Y. Huang, D. Yang, B. Feng, A. Tian, P. Dong, S. Yu, and H. Zhang, "A GNN-enabled multipath routing algorithm for spatial-temporal varying LEO satellite networks," *IEEE Transactions on Vehicular Technology*, 2024.

[22] X. Deng, L. Chang, S. Zeng, L. Cai, and J. Pan, "Distance-based back-pressure routing for load-balancing LEO satellite networks," *IEEE Transactions on Vehicular Technology*, 2023.

[23] J. Liu, R. Luo, T. Huang, and C. M. C., "A load balancing routing strategy for LEO satellite network," *IEEE Access*, 2020.

[24] B. Soret, I. Leyva-Mayorga, F. Lozano-Cuadra, and M. D. Thorsager, "Q-learning for distributed routing in LEO satellite constellations," in *IEEE ICMLCN*, 2024.

[25] M. M. H. Roth, "Analyzing source-routed approaches for low earth orbit satellite constellation networks," in *Proceedings of the 1st ACM Workshop on LEO Networking and Communication*, 2023.

[26] M. Handley, "Delay is not an option: Low latency routing in space," in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, 2018.

[27] K. Han, B. Xu, S. Guo, W. Gong, S. Chatzinotas, I. Maity, Q. Zhang, and Q. Ren, "Non-grid-mesh topology design for megaLEO constellations: An algorithm based on NSGA-III," *IEEE Transactions on Communications*, 2024.

[28] S. Zhang and K. L. Yeung, "Scalable routing in low-earth orbit satellite constellations: Architecture and algorithms," *Computer Commun.*, 2022.

[29] Y. Li, L. Liu, H. Li, W. Liu, Y. Chen, W. Zhao, J. Wu, Q. Wu, J. Liu, and Z. Lai, "Stable hierarchical routing for operational LEO networks," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, 2024. [Online]. Available: https://doi.org/10.1145/3636534.3649362

[30] Z. Lai, H. Li, Y. Wang, Q. Wu, Y. Deng, J. Liu, Y. Li, and J. Wu, "Achieving resilient and performance-guaranteed routing in space-terrestrial integrated networks," in *IEEE INFOCOM 2023*, 2023.

[31] 3rd Generation Partnership Project (3GPP), "Study on New Radio (NR) to support non-terrestrial networks," 3GPP, Tech. Rep. TR 38.811 V15.2.0, 2019, [Online; accessed April 7, 2025]. [Online]. Available: https://www.3gpp.org/dynareport/38811.htm

[32] NASA's Scientific Visualization Studio, "Geomagnetic storm causes satellite loss," https://svs.gsfc.nasa.gov/5193/, 2024, accessed: 2-March-2024.

[33] R. Ma, X. Yang, Y. Liu *et al.*, "Leo satellite network access in the wild: Potentials, experiences, and challenges," *arXiv preprint arXiv:2405.06801*, 2024. [Online]. Available: https://arxiv.org/abs/2405.06801

[34] C. B. Office, "Large constellations of low-altitude satellites: A primer," Congress of the United States, Tech. Rep., 2023. [Online]. Available: https://www.cbo.gov/publication/59175

[35] J. Lin, H. Liu, J. Zhou *et al.*, "Large-volume leo satellite imaging data networked transmission scheduling problem: Model and algorithm," *Expert Systems with Applications*, vol. 249, p. 123649, 2024.

[36] G. Silva and et al., "Distributed intrusion detection system for cubesats, based on deep learning packets classification model," in *European Space Agency–CubeSat Industry Days*, 2023. [Online]. Available: https://indico.esa.int/event/528/attachments/5988/10192/Distributed_intrusion_detection_system_for_CubeSats_based_on_deep_learning_packets_classification_model.pdf

[37] A. Keller, "Towards corba-based enterprise management: managing corba-based systems with snmp platforms," in *Proceedings Second International Enterprise Distributed Object Computing (Cat. No.98EX244)*, 1998.

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

[39] Mynaric, "Condor Mk3 - optical communication terminals for air and space applications," https://mynaric.com/products/space/condor-mk3/, 2023, accessed: 5-March-2024.

[40] Space Exploration Technologies Corp. (SpaceX), "Modification of the authorization for the spacex NGSO satellite system," FCC Report, 2018, accessed: 7-March-2024. [Online]. Available: https://www.fcc.report/IBFS/SAT-MOD-20181108-00083/1569860.pdf

[41] X. Wang, X. Han, M. Yang, C. Xing, Y. Wang, S. Han, and W. Li, "Investigating inter-satellite link spanning patterns on networking performance in mega-constellations," 2023. [Online]. Available: https://arxiv.org/abs/2312.15873

[42] C. Wu, S. Han, Q. Chen, Y. Wang, W. Meng, and A. Benslimane, "Enhancing LEO mega-constellations with inter-satellite links: Vision and challenges," 2024. [Online]. Available: https://arxiv.org/abs/2406.05078

[43] Q. Li, M. El-Hajjar, K. Cao, C. Xu, H. Haas, and L. Hanzo, "Holographic metasurface-based beamforming for multi-altitude leo satellite networks," 2025. [Online]. Available: https://arxiv.org/abs/2501.04164

[44] Microsoft, "Cosmicbeats-simulator: A space simulation platform," 2023. [Online]. Available: https://github.com/microsoft/CosmicBeats-Simulator

[45] Center for International Earth Science Information Network (CIESIN), Columbia University, "Gridded population of the world, version 4 (GPWv4): Population count, revision 11," 2018. [Online]. Available: https://doi.org/10.7927/H4JW8BX5

[46] Starlink Insider, "Starlink by the numbers," 2024. [Online]. Available: https://starlinkinsider.com/starlink-statistics/

[47] Celestrak, "Oneweb TLEs (two-line element sets)," https://celestrak.org/norad/elements/table.php?GROUP=oneweb&FORMAT=tle, 2023, accessed: 23-September-2023.

[48] International Telecommunication Union (ITU), "Facts and figures 2023: Internet traffic," https://www.itu.int/itu-d/reports/statistics/2023/10/10/ff23-internet-traffic/, 2023, accessed: 29-February-2024.

[49] Ericsson, "Analysis of traffic profiles – mobility report," 2023, accessed: 29-February-2024. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/analysis-traffic-profiles

[50] Cisco, "Cisco annual internet report," https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html, 2023, accessed: 29-February-2024.

[51] Amsterdam Internet Exchange (AMS-IX), "AMS-IX total traffic," https://stats.ams-ix.net/index.html, 2024, accessed: 29-February-2024.