Small-scale LEO Satellite Networking for Global-scale Demands

Yuanjie Li[1,2], Yimei Chen[1], Jiabo Yang[1], Jinyao Zhang[1], Bowen Sun[1], Lixin Liu[1], Hewu Li[1,2], Jianping Wu[1,2], Zeqi Lai[1,2], Qian Wu[1,2], Jun Liu[1,2]
[1]Tsinghua University [2]Zhongguancun Laboratory

ABSTRACT

Do we really need 10,000s of Low Earth Orbit (LEO) satellites to meet huge global Internet demands? While proven feasible and valuable, such LEO mega-constellation networks have raised concerns about their prohibitive capital expenditures, market monopoly, and unsustainable use of space. Instead, our analysis reveals that most of their satellites can be wasted due to their mismatch with physically uneven demands. We thus propose TinyLEO, a software-defined solution to shrink LEO network size for enormous global demands via dynamic *spatiotemporal supply-demand matching*. TinyLEO sparsifies satellite supplies on demand by combining diverse yet sparse orbits, hides complexities of this sparse LEO network via orbital model predictive control, and shifts the responsibility for handling these complexities to its geographic segment anycast for higher network usability, lower resource wastes, faster failovers, simpler satellites, and more flexible network orchestration. We have prototyped TinyLEO as a community toolkit for open research. Our evaluation using this toolkit shows that TinyLEO can compress the existing LEO mega-constellation network size by 2.0–7.9×, cut control plane costs by 1–3 orders of magnitude, *and* maintain the same demands and comparable data plane performance.

CCS CONCEPTS

• **Networks** → **Network protocol design**; **Physical topologies**.

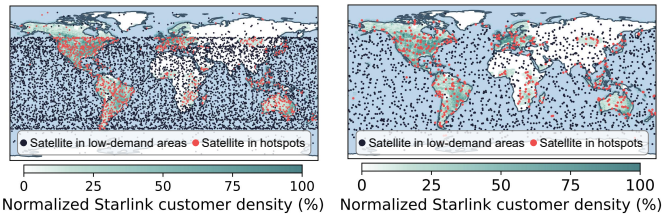
KEYWORDS

Satellite networking, Low Earth Orbit (LEO) satellite constellation

ACM Reference Format:
Yuanjie Li, Yimei Chen, Jiabo Yang, Jinyao Zhang, Bowen Sun, Lixin Liu, Hewu Li, Jianping Wu, Zeqi Lai, Qian Wu, Jun Liu. 2025. Small-scale LEO Satellite Networking for Global-scale Demands . In *ACM SIGCOMM 2025 Conference (SIGCOMM '25), September 8–11, 2025, Coimbra, Portugal.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3718958.3750525

1 INTRODUCTION

The Low Earth Orbit (LEO) satellite mega-constellations are revolutionizing the "Internet from space." Thanks to a vast number of satellites with ultrahigh-capacity links, they promise high-speed Internet access to the numerous "offline" users everywhere on Earth. To



(a) Starlink mega-constellation **(b) Our solution: TinyLEO**

Figure 1: Small LEO network for enormous demands

date, SpaceX Starlink, the leading LEO mega-constellation, has offered 350 Tbps total capacity with around 7,000 satellites and 13,000 inter-satellite links (ISLs) for more than 4.6 million active users from 118 countries, enabling about 100 Mbps download speeds per user [1]. More LEO mega-constellation networks, such as OneWeb [2], GW [3], and Kuiper [4], are also under planning or deployment to catch up with Starlink.

Despite this great success, LEO satellite mega-constellation networks are not without concerns. Even with technological advances in rocket reusability and satellite miniaturization, their manufacturing, launching, and operation costs are still prohibitive for most Internet service providers (ISPs) and countries [5, 6]. This barrier to entry leads to a monopolistic global LEO network market by very few tech giants, which concerns regulators [7] and developing countries [8, 9]. More importantly, numerous satellites in LEO mega-constellations severely congest Earth's orbits to threaten the sustainable and safe use of space for all humanity [10–13].

To resolve these concerns, we explore alternatives to LEO mega-constellation networks. Our goal is to shrink the LEO network scale for fewer satellites *and* retain comparable usability, performance, and resiliency to mega-constellations for large or global-scale users. This can help more small ISPs and countries own affordable satellite networks, democratize this global market with more players, and relieve orbital congestion for sustainability and safety.

Our work starts with a simple insight: *Most satellites in LEO mega-constellation networks are underutilized.* For ease of networking and management, most mega-constellations in use distribute their satellites almost uniformly in space, leading to a homogeneous global network supply. But as shown in Figure 1a, the global network demands are uneven, with over 70% of users concentrated in 5% of the land and very few users in oceans covering 70.8% of the Earth. This physical mismatch wastes most satellites in low-demand areas, which cannot be fully eliminated by higher-layer load balancing like local beam steering or global traffic engineering.

To this end, we seek to shrink LEO networks by cutting these underutilized satellites. Clearly, the most fundamental method is to rearrange their layouts to match the uneven demands with fewer satellites, calling for *non-uniform LEO networks*. At first glance, this

More information about TinyLEO: https://tinyleo-toolkit.github.io/TinyLEO/.

SIGCOMM '25, September 8–11, 2025, Coimbra, Portugal

ACM ISBN 979-8-4007-1524-2/25/09
https://doi.org/10.1145/3718958.3750525

task seems simple since state-of-the-art geostationary (GEO) satellites have achieved this. However, it becomes extremely challenging for dynamic LEO satellite networks for two reasons:

- **Unstable supply-demand match in mobility:** Unlike GEO satellites, LEO satellites suffer from inevitable fast, asynchronous movement relative to the rotating Earth, leading to a rapidly changing coverage area on the ground. This extreme mobility makes it hard for LEO constellations to maintain their persistent match with uneven demands.
- **Hard-to-use complex networking:** A non-uniform LEO network should unevenly place satellites across latitudes, longitudes, and time to match imbalanced demands, incurring complex motions among heterogeneous satellites. This intensifies satellite link switches, topology updates, and routing path changes, all being risks for LEO network availability, efficiency, resiliency, and usability [14–18].

We address both challenges with TinyLEO, a software-defined small-scale LEO networking for global-scale demands via *spatiotemporal supply-demand matching*. TinyLEO utilizes orbital diversity to enable stable on-demand network supply despite extreme satellite mobility. It decouples high-level network intents (stable demands) from their low-level enforcements (dynamic supplies) for high usability. Specifically,

(1) **On-demand LEO network sparsification.** TinyLEO combines *diverse yet sparse orbits* to complement each other for cutting satellite redundancy over space and time (akin to video compression). This can be efficiently achieved via *compressed sensing* [19–22], an advanced sparse signal reconstructor from very few Earth-repeat ground tracks of satellites ("textures").
(2) **Control plane: stable intent + orbital MPC.** To retain high network usability, TinyLEO hides most complexities of the sparse LEO network's complex physical dynamics from high-level networking demands. It decomposes its control plane into *geographic traffic engineering intents* (for topology and routing optimization on a stable basis) and an orbital model predictive control (MPC [23]) shim layer (for runtime compilation into dynamic network supplies) for simple and flexible orchestration and low signaling overhead under extreme LEO dynamics.
(3) **Data plane: geographic segment anycast.** TinyLEO shifts the responsibility for handling most LEO dynamics to each satellite's local data plane, which is closer to these dynamics for more timely and efficient adaptations than the remote control plane. It enhances its data plane with *geographic segment anycast* for policy-compliant, flexible local (re)routing, load balancing, and fast recovery from outages (e.g., solar storms and ISL disruption).

We have prototyped TinyLEO as a community toolkit for open research. Our evaluation using this toolkit validates that TinyLEO can compress the existing LEO mega-constellation network size by 2.0–7.9× (hence alleviating space congestion/pollution and saving ISPs' capital expenditures of satellites) and cut control plane costs by 1–3 orders of magnitude while meeting broadband network demands at comparable data plane performance.

Toolkit release and video: Our TinyLEO community toolkit and demo video are publicly available at [24].

Ethics: This paper raises no ethical concerns.

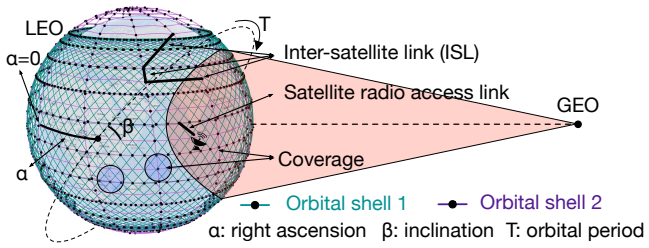


Figure 2: LEO satellite mega-constellation network

2 LEO NETWORK SUPPLY-DEMAND GAP

This section motivates our design of TinyLEO with analysis and empirical validations of the networking supply, demand, and their mismatch in LEO satellite mega-constellations.

2.1 Large-scale LEO Network Supply

LEO satellite networks aim to connect the vast number of "unconnected" users without terrestrial network access. Each LEO satellite operates at 340–2,000 km altitudes, 20–100× closer to users than traditional GEO satellites at the 35,786 km altitude for lower latency and stronger radio signals. It equips ultrahigh-capacity radio links (e.g., 96 Gbps in Starlink v2 mini [1]) for end users and optionally laser links among satellites (e.g., 3 ISLs per Starlink satellite, up to 200 Gbps for each [1]) for global routing. As shown in Figure 2, each LEO satellite only has a finite coverage, so a constellation of satellites is necessary to cover the Earth everywhere.

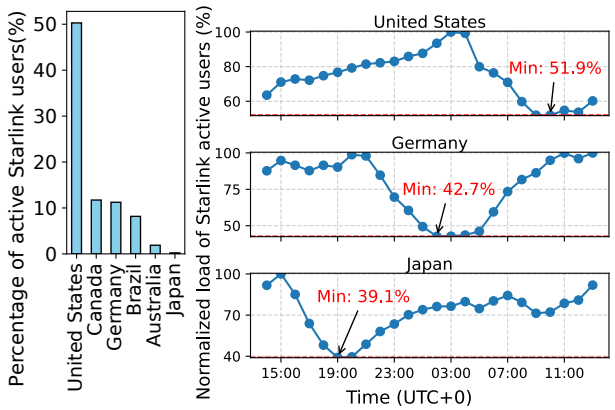
For a basic global coverage, a small constellation with tens of LEO satellites usually suffices. However, such a small LEO constellation does not have sufficient network capacity to meet numerous users' demands for high-speed Internet. For example, each Starlink v2 mini satellite with a 96 Gbps radio link can only serve up to 960 users with concurrent 100 Mbps downlink speed. To this end, operational LEO networks have been actively expanding their capacity with more satellites, aiming for *mega-constellations* with 1,000s–10,000s of satellites [1]. Meanwhile, they are also increasing each satellite's radio link capacity to serve more users (e.g., from 15 Gbps in Starlink v1.5 to 96 Gbps in v2 mini [1, 25]).

Early LEO communications satellites, such as Starlink v1, simply act as **access networks** by relaying signals between terminals and ground stations (gateways to Internet). This "bent pipe" mode suffers from low service coverage due to its reliance on ground stations within each LEO satellite's small visibility. To this end, recent LEO satellites have equipped ISLs to form **backbone networks** for global routing. By 2024, Starlink has activated 13,000 ISLs [1] with over 99% uptime and 42 petabytes of daily traffic delivery [17]. These data are typically forwarded across ISLs via tunneling [26].

2.2 Imbalanced Global Network Demand

Although LEO networks can cover everywhere on Earth, their demands are physically uneven for at least 3 reasons:

- **Uneven global populations:** As shown in Figure 1, more than 70% of the world's population is concentrated in 5% of the land, while oceans with very few users cover 70.8% of the Earth's surface. This spatial population bias inevitably leads to an uneven LEO network user distribution.



(a) Spatial unevenness (b) Temporal diurnal dynamics

Figure 3: Starlink's global user traffic distribution based on Cloudflare's public measurements [27]

- **Differentiated needs for satellite networks:** Urban residents are usually surrounded by terrestrial broadband cables, WiFi, and cellular networks, thus less needing LEO networks than rural, maritime, and aviation users.
- **Policy constraints:** Due to complicated commercial and international policies, certain areas may be restricted from accessing LEO networks even if they are physically visible.

As a result, the real global distribution of LEO network users is uneven. Figure 3 presents the distribution of Starlink's worldwide traffic based on Cloudflare's DNS-based user activity measurements over 2 days at 15-minute intervals [27], which is in concert with Starlink's official reports in [1] (see Figure 13a). It shows that Starlink users are more concentrated around a few regions than others, resulting in a long-tail distribution. Figure 3b also reveals periodic diurnal fluctuations of user activity across global time zones. It proves that LEO network users vary spatially and temporally.

2.3 Network Supply-Demand Mismatch

Unfortunately, existing LEO mega-constellation network supplies are not well aligned with the uneven demands. For ease of networking and management, most operational LEO networks uniformly distribute their satellites in space based on homogeneous layouts (e.g., Walker constellation [28]). This uniform LEO network severely wastes its satellite link capacity and limits its serviceable users in two aspects:

- *Underutilization in idle areas*: As shown in Figure 4, LEO satellites move across the Earth at about 7 km/s. Most of them are unavoidably exposed in areas with few users and left idle most of the time. This makes each satellite's added capacity in §2.1 less useful to serve more users and even worsens its capacity waste in low-demand areas.
- *Underutilization propagation from hotspots*: Due to their fast mobility, LEO satellites that cover hotspots *for now* will soon leave. To meet excessive demand in these hotspots, LEO networks must ensure that there are always enough satellites covering these areas, which, however, leads to more satellite capacity waste elsewhere as they will move from hotspots to low-demand areas.

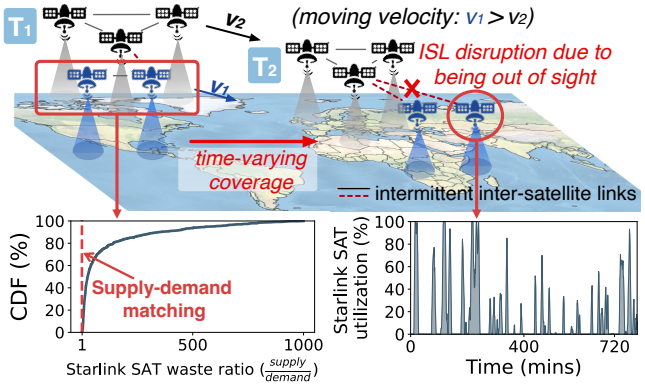
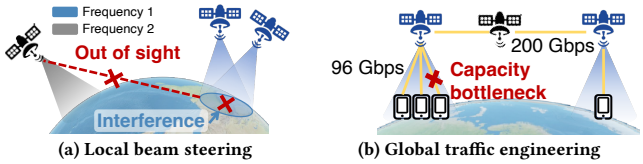


Figure 4: LEO network resource waste under mobility



(a) Local beam steering (b) Global traffic engineering

Figure 5: Spatial limits for LEO network load balancing

Can load balancing help? A standard solution to this mismatch in traditional networking is load balancing. Although partially applicable to LEO networks, it encounters physical constraints to fulfill its potential:

- *Local radio access beam steering:* A common practice for satellite load balancing is to steer its radio access link beams to high-demand areas, which has been extensively used in Starlink [29]. However, as shown in Figure 5a, a satellite beam's maximum steering angle has an upper bound due to physical visibility constraints (e.g., 56.7° in Starlink [30]). Even if it is idle over broad oceanic areas, it cannot always redirect its beams to the nearest terrestrial high-demand areas for load balancing. In addition, to avoid co-channel interferences, nearby satellites' beams using the same RF bands cannot be steered to the same area [29, 30], further limiting their flexibility for load balancing.
- *Global traffic engineering:* Another method to balance the global LEO network load is to shift traffic flows from hotspots to low-demand satellites via ISLs, such as trans-oceanic routes [31] or backhaul traffic aggregation [32]. While helpful for inter-satellite backbone networks, it cannot fully shift the *last-mile* user traffic between hotspots and idle areas as explained above. In reality, most LEO networks' utilizations are bottlenecked by their last-hop radio access links to users rather than ISLs (e.g., 96 Gbps in Starlink versus its 200-Gbps ISLs [1]). Traffic engineering cannot eliminate this bottleneck in uniform LEO networks.

How about a multi-shell LEO network? In fact, Starlink is aware of this supply-demand gap and seeks to mitigate it using multi-shell LEO networks. As exemplified in Figure 2, most Starlink satellites are in orbits with 53–53.2° inclinations to serve most users in low latitudes, while others use 97.6° inclinations to cover few users in the Arctic and Antarctica.

Unfortunately, this simple fix is not enough to eliminate the supply-demand gap. LEO network demands are uneven in not only

latitudes but also *longitudes* (Figure 1). Since each shell's satellites remain uniform, they cannot match uneven longitudinal demands for high utilization (Figure 4). The Earth's rotation also complicates this matching due to its asynchronous longitudinal mobility to LEO satellites. As a result, even with this multi-shell design, Starlink's satellite utilization remains unsatisfactory, as shown in Figure 4.

What if a non-uniform LEO network? The fundamental solution to this satellite underutilization is to rearrange the physical LEO network layout to match the uneven global user demand, calling for non-uniform satellite distributions. While conceptually simple and already feasible for GEO satellites, it is hard to realize in LEO networks for two reasons:

- **Hard to maintain physical match in mobility:** Unlike GEO satellites, LEO satellites suffer from inevitable fast, asynchronous movement relative to the Earth at about 7 km/s, leading to a rapidly changing coverage area on the ground (Figure 4). For instance, each Starlink satellite's coverage to each area can only last up to 3 minutes. This space-terrestrial dynamics makes it difficult for the constellation to keep matching the uneven demands.
- **Dynamic heterogeneous networking:** A non-uniform LEO constellation should unevenly distribute satellites in different orbits to match imbalanced demands. As we will see in §4.2, this complicates satellites' relative motions and intensifies their network topology and routing changes. These dynamics can degrade network availability, efficiency, and resiliency [14–18], eventually challenging the usability of non-uniform LEO networks.

Problem statement: This work explores an alternative to LEO mega-constellation networks by addressing these challenges for the non-uniform LEO network. We seek *usable small-scale* non-uniform LEO networking for global-scale high-speed Internet demands. It can compress the LEO network size for affordability and space sustainability, while retaining comparable network availability, performance, robustness, and flexibility to LEO mega-constellations.

3 SOLUTION OVERVIEW

We propose TinyLEO, a software-defined small-scale LEO networking that achieves all the above goals. At its core, TinyLEO enables *spatiotemporal supply-demand matching* in LEO networks to cut unnecessary satellites. It decouples high-level inter-networking intents (stable demands) from their low-level enforcements (dynamic supplies) for high network usability. This paradigm is achieved by exploiting three unique opportunities in LEO satellite networks:

(1) *Predictive orbital motions:* While the geo-asynchronous extreme mobility of LEO satellites forces a dynamic network supply-demand match, it remains predictable by orbital laws to facilitate a *semi-stable match*;
(2) *Combine diverse yet sparse orbits:* Each individual LEO cannot perfectly match its homogeneous satellites with uneven global demands. However, a combination of heterogeneous LEOs (each having sparse satellites) with different orbital periods, inclination angles, and right ascensions can complement each other for near-optimal matching;
(3) *Geographic invariants for usable networking:* Despite frequent changes of visible satellites, each geographic area retains a constant number of available satellites after the supply-demand match, offering stable guidance to enforce usable satellite topology and routing.

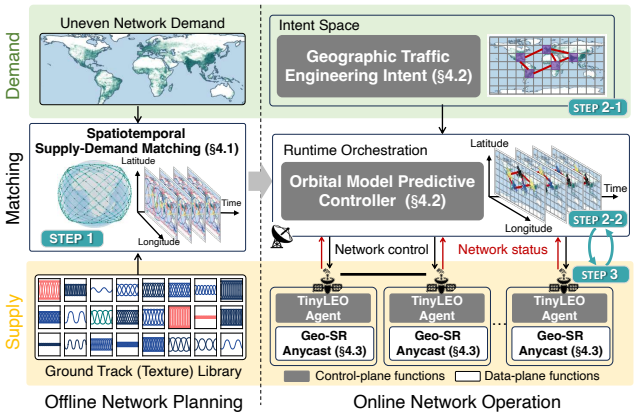


Figure 6: Overview of TinyLEO

Figure 6 overviews the workflow of TinyLEO based on these insights. It comprises an offline sparse network synthesizer and online network control/data-plane functions:

- **On-demand LEO network sparsification (§4.1).** The synthesizer plans a sparse LEO network layout to match spatiotemporally uneven demands. TinyLEO achieves this by combining diverse yet sparse LEOs to cut satellite waste via *compressed sensing* (akin to video compression).
- **Control plane: stable intent + orbital MPC (§4.2).** On top of the sparse LEO network, TinyLEO's control plane compiles high-level topology and routing intents to low-level instructions for the data plane. For high network usability, we split it into a stable geographic networking intent abstraction and orbital model predictive controller (MPC). This separation of concerns hides complexities of complex LEO physical dynamics from network demands for flexible orchestration and low signaling costs.
- **Data plane: geographic segment anycast (§4.3).** TinyLEO moves the responsibility for tackling most sparse LEO networks' complex dynamics to each satellite's data plane. It achieves this via *geographic segment anycast* (i.e., packets destined to a geographic cell can be forwarded to any satellite covering this cell) for policy-compliant, efficient local (re)routing, load balancing, and failovers.

4 DESIGN OF TinyLEO

This section addresses three critical issues to realize TinyLEO:

(I) How to sparsify the LEO network supply while still meeting uneven global broadband demands (§4.1)?
(II) How to streamline this sparse LEO network's control plane for diverse high-level networking intents (§4.2)?
(III) How to enhance the data plane to enforce these intents in the sparse LEO network's dramatic dynamics (§4.3)?

4.1 On-demand LEO Network Sparsification

TinyLEO's core idea is to accurately match the sparse LEO network supply with uneven global demands to cut satellites. As explained in §2.3, this calls for a *non-uniform* LEO network layout

across *latitudes*, *longitudes*, and *time*. It departs from existing LEO mega-constellations that can only differentiate their supply across latitudes via multiple orbital shells at most (§2.3). To realize this, we should answer three questions:

(1) How to enable uneven satellite distributions across latitudes, longitudes, and time for on-demand matching?
(2) How to stabilize this uneven network supply under fast, geo-asynchronous LEO satellite mobility?
(3) How to use as few LEO satellites as possible to match uneven global broadband demands?

Diverse orbits for on-demand matching: TinyLEO does not place dense satellites in homogeneous orbits as mega-constellations. Instead, it combines diverse orbits (each with sparse satellites) to match uneven network demands across latitudes, longitudes, and time. Figure 7 depicts diverse orbital parameters' spatiotemporal impacts on matching:

- *Inclination for latitudinal diversity:* An orbit's inclination angle β determines its maximal latitudes on Earth to cover. As explained in §2.3 and shown in Figure 2, existing LEO mega-constellations deploy multiple orbital shells with various inclinations to match demands across latitudes;
- *Right ascentation for longitudinal diversity:* The longitude of each orbit's ascending angle bounds its horizontal areas to cover. Shifting the right ascension of the ascending node can match a satellite supply to different longitudinal areas;
- *Orbit period for temporal diversity:* It decides how frequently each satellite revisits geographic areas. LEO satellites with diverse orbital periods can complement each other's coverage at different times to match temporal demand changes.

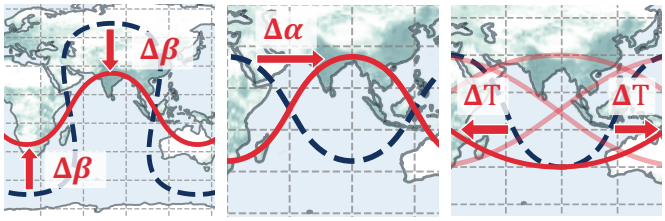
Intuitively, more orbits with these diverse parameters can better match LEO network supplies with uneven demands.

Earth-repeat ground tracks for stable matching: While the above orbital parameters can seemingly generate diverse satellites to match uneven demands in different areas, this matching can be unstable due to LEO mobility. Different from terrestrial or GEO satellite networks, it is hard to fix a geo-asynchronous LEO satellite to serve a single area, making it hard for the non-uniform LEO network to fix its supply-demand match anywhere for continuous high-speed services.

TinyLEO makes a case to solve this issue by exploiting ubiquitous *Earth-repeat ground tracks* [33, 34]. Intuitively, the LEO satellite in an Earth-repeat orbit always periodically revisits the same geographic area despite its mobility and Earth rotation. To achieve this, its orbital period T satisfies

$$T/T_E = p/q,\ p, q \in \mathbb{N}^+ \tag{1}$$

where T_E is the Earth's rotation period (24h) and $p < q$ can be any positive integers. A salient feature of Earth-repeat orbit is its fixed satellite ground track: As visualized in Figure 7, each satellite in this orbit always revisits the same geographic area after q rounds (or equivalently p days of Earth's rotations), forming a stable basis of matching these areas' demands. Moreover, Earth-repeat LEO orbits are abundant and diverse: They can be easily generated on demand by varying (p, q) and diverse orbital parameters in Figure 7, which form an over-complete set of stable LEO satellite network supplies with diverse geographic coverages and satellite densities.



(a) Inclination angle β (amplitude) (b) Right ascension α (phase) (c) The orbital period T (frequency)

Figure 7: On-demand network supply via diverse orbits

Basic network sparsification setup: Based on the above primitives, TinyLEO models LEO network sparsification as a spatiotemporal supply-demand matching problem (akin to video compression), as illustrated in Figure 8:

- *Uneven network demands:* We divide the Earth's surface into m geographic cells to let network designers/operators plan each cell i's maximal serviceable demand at time t (y_i^t, in the unit of the number of satellites) as TinyLEO's input. This maximal serviceable demand y_i^t can be customized to account for various real-world factors, including but not limited to local radio access link capacity, ISL capacity for backbone (§4.2), over-provisioning for remote areas or unpredictable traffic spikes, backup satellites for fault tolerance (e.g., by solar storms and cosmic radiations), and periodic diurnal dynamics (§2.2). How to balance them is out of this paper's scope and has been well-discussed in prior network planning works. This separation of concerns ensures TinyLEO's efficiency despite demand diversity and changes. We denote all geographic cells' demands as a vector $\mathbf{y_t} = [y_t^1, y_t^2, ..., y_t^m]^T$. The uneven demands imply that $\mathbf{y}$ is a *sparse vector*, i.e., $y_i^t \approx 0$ for most i and t.
- *Diverse yet sparse network supply:* TinyLEO maintains a LEO Earth-repeat ground track ("texture") library as an over-complete set of candidates for network supply. These candidates are identified by enumerating (p, q) pairs in Equation 1 whose orbits are not occupied or allocated by space regulators (e.g., ITU and FCC). The j-th ground track in this library is uniquely characterized by its orbit's right ascension α_j, inclination β_j, and period T_j (or p_i and q_i) in Figure 7. TinyLEO does not pose any constraints for these orbital parameters; any Earth-repeat orbit is allowed. It will optimize the number of satellites $x_j \geq 0$ ($x_j \in \mathbb{N}$) to place in this orbit and their distributions. We represent all orbits' number of satellites as a vector $\mathbf{x} = [x_1, x_2, ..., x_n]^T$, where n is the number of candidates in this library and should be large ($m \ll n$) for over-completeness. Due to TinyLEO's nature, most candidates' satellites will be eventually empty (i.e., $x_j \approx 0$ for most j), making $\mathbf{x}$ a sparse vector as well.
- *Spatiotemporal supply-demand matching:* Given the global network demand $\mathbf{y_t}$ and all available Earth-repeat orbital ground tracks covering these areas, we arrange each Earth-repeat orbit's satellites $\mathbf{x}$ and combine them to satisfy $\mathbf{y_t}$ everywhere, anytime. In this process, we minimize the total number of satellites. This can be formulated as

$$\min \quad ||\mathbf{x}||_1 \tag{2}$$
$$\text{s.t.} \quad \mathbf{A_t x} \geq \mathbf{y_t} \quad \forall t = 1, 2, ..., T_{max} \tag{3}$$
$$x_i \in \mathbb{N} \quad \forall i = 1, 2, ..., n \tag{4}$$

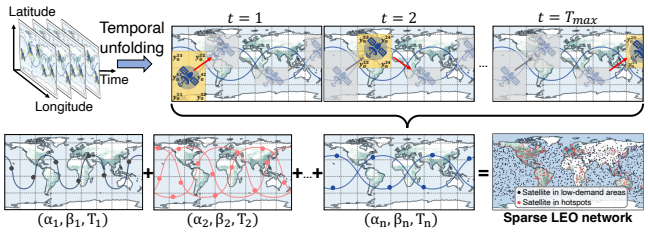


Figure 8: TinyLEO's on-demand network sparsification using diverse yet sparse Earth-repeat ground tracks

where $||x||_1 = \sum_{i=1}^{n} x_i$ is the total number of satellites, $T_{max} = LCM(T_1, T_2, ..., T_n)$ is the maximmal time slot to consider (i.e., the least common multiple of all orbital periods, after which all satellites repeat their ground tracks), $\mathbf{A_t} \in \mathbb{R}^{m \times n}$ is the coverage matrix at time t with $A_t(i,j) \in [0,1]$ being the fraction of satellite j's radio link coverage over cell i, and hence $\mathbf{A_t x}$ is the runtime LEO network capacity supply at time t. Due to predictable LEO mobility, each coverage matrix $\mathbf{A_t}$ can be pre-computed based on orbital laws, the Earth's rotation period T_E, and orbital parameters α_i, β_i, and T_i.

Sparse matching via compressed sensing: Equation 2–4 is a standard integer linear programming problem, which is NP-hard in general [35]. Fortunately, in our context, the LEO network supply $\mathbf{x}$, the network demand $\mathbf{y_t}$, and satellite coverage matrix $\mathbf{A_t}$ are *all sparse*. This domain-specific property can cast our optimization as a sparse signal recovery problem, which can be more efficiently solved with *compressed sensing* techniques [19–22][1].

Algorithm 1 presents TinyLEO's sparse approximation of the solution for Equation 2–4, which is a variant of the matching pursuit (MP) algorithm from compressed sensing [19, 22]. As visualized in Figure 8, to unify the spatial and temporal matching, this algorithm first unfolds the time-evolving LEO network demands and coverage matrix as a concatenation of a single vector/matrix (line 1–2). Then it iteratively searches the ground track i whose satellites can cover the maximum amount of the unsatisfied residual demands (line 6–7), determines the number of satellites x_i this ground track should add (line 8), and updates the residual demands by subtracting this ground track's satisfied demands (line 9). The algorithm stops when the residual demands are less than a predefined network availability threshold ϵ (e.g., $\epsilon \geq 99\%$ across space and time [17]). Since the LEO network demand $\mathbf{y_t}$ and coverage matrix $\mathbf{A_t}$ are sparse, Algorithm 1 can quickly meet this criterion to complete its spatiotemporal matching. Its greedy nature also ensures that its output LEO network supply $\mathbf{x}$ is sparse enough to save satellites.

Incremental LEO network expansion: Algorithm 1 also eases incremental deployment. Its greedy search for satellites that best match residual demands forms a natural step-by-step satellite launching plan. Suppose the ISP wants to serve more users than initially planned. To satisfy it, TinyLEO can simply set the residual demand $\mathbf{z^k}$ as this additional demand and continue Algorithm 1 to determine new satellites to launch without affecting its existing satellites.

Long-term stability: In reality, satellites may deviate from Earth-repeat ground tracks due to orbital decays or collision avoidance maneuvers. To retain network demand-supply match, each satellite can routinely calibrate its location in its daily orbit maintenance (already done by Starlink for stable networking [13, 36]) to stay in its Earth-repeat ground track.

[1]Simply speaking, compressed sensing recovers an unknown k-sparse signal $\mathbf{x} \in \mathbb{R}^{n \times 1}$ from a small number of linear measurements $\mathbf{y} \in \mathbb{R}^{m \times 1}$ ($m \ll n$) that $\mathbf{y} = \mathbf{Ax}$ ($\mathbf{A} \in \mathbb{R}^{m \times n}$). If $m = O(k \log(n/k))$, polynomial-time recovery algorithms exist with provably near-optimal accuracy guarantees [19–21].

Input: $\{\mathbf{y_t}, \mathbf{A_t}\}_{t=1}^{T_{max}}, \epsilon$ # ϵ is network availability ratio

1 # Temporal unfolding of LEO network demands & coverage

2 $\hat{\mathbf{y}} \leftarrow [\mathbf{y_1}; \mathbf{y_2}; ...; \mathbf{y_{T_{max}}}]$; $\hat{\mathbf{A}} \leftarrow [\mathbf{A_1}, \mathbf{A_2}, ..., \mathbf{A_{T_{max}}}]$;

3 $k \leftarrow 0$; $\mathbf{r^0} \leftarrow \hat{\mathbf{y}}$; $\mathbf{x^0} \leftarrow \mathbf{0}$;

4 **while** $||\mathbf{r^k}||_1 \geq 1 - \epsilon$ **do**

5 $\quad k \leftarrow k + 1$;

6 $\quad g^k \leftarrow \hat{\mathbf{A}}^{\mathbf{T}} \mathbf{r^k}$; # Each ground track's satisfiable demand

7 $\quad i_k \leftarrow \arg\max_j |g_j^k|$; # Match the best ground track

8 $\quad x_{i_k}^k \leftarrow x_{i_k}^{k-1} + |g_{i_k}^k|$; # Add satellites on this track

9 $\quad \mathbf{r^k} \leftarrow \mathbf{r^{k-1}} - \hat{A}_{i_k} g_{i_k}^k$; # Update the residual demand

10 **end**

Output: $\lceil \mathbf{x^k} \rceil$ # Approximation of Equation 2–4

Algorithm 1: Sparse spatiotemporal matching pursuit[2]

4.2 Control Plane: Stable Intent + Orbital MPC

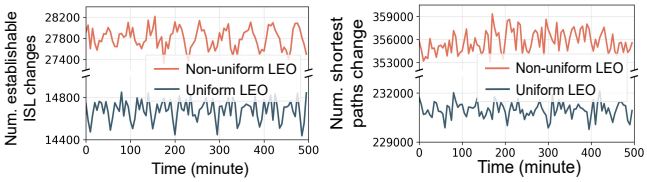
After shrinking the LEO network size, TinyLEO's next task is to make this non-uniform LEO network *easily usable*. If these LEO satellites simply act as **access networks** in §2.1, this mission is already completed in §4.1 since TinyLEO ensures sufficient radio access link capacity everywhere, anytime. If these satellites also need to act as **backbone networks** to deliver data traffic through their ISLs, TinyLEO should also manage additional complexities of their topology and routing in this extremely dynamic, non-uniform LEO network.

To understand the impacts of the non-uniform LEO physical dynamics on topology and routing, Figure 9 projects the update frequency of all potential ISLs and shortest paths among satellites in TinyLEO and uniform LEO networks of the same size. Due to heterogeneous satellite motions by diverse orbital parameters, the sparse LEO network will inevitably encounter more ISL and route changes than the uniform one. If not properly tackled, both can harm network availability, efficiency, and resiliency [14–18]. This complexity is unavoidable for sparse LEO networks.

In operational LEO networks, the contemporary solution to manage this complexity is to utilize regular orbital motions for *predictive* temporospatial software-defined networking (TS-SDN). For instance, Starlink [17, 26] and Aalyria [14–16] adopt TS-SDN to forecast satellite motions, search for potential ISLs and paths over space and time, and optimize the evolving satellite network topology and routing accordingly. While effective and affordable in uniform LEO networks, they may be unstable and expensive in the extremely dynamic non-uniform LEO networks for two reasons:

- *High control costs:* Existing TS-SDN exposes almost all low-level LEO physical dynamics, logical network topology, and hop-by-hop satellite routing to the upper layer for control [14–16, 37]. As evidenced in Figure 9, with frequent ISL and path changes, this fine-grained control becomes overwhelming and triggers signaling storms to satellites for exhaustive inter-satellite routing reconfigurations.

[2]In Algorithm 1, $\hat{A}_j$ is the j-th column of $\hat{\mathbf{A}}$ and $\lceil \mathbf{x} \rceil$ is the ceiling of $\mathbf{x}$.



(a) Establishable ISLs (b) Shortest path among satellites

Figure 9: Dynamic non-uniform LEO physical network

- *Slow response to unpredictable dynamics:* Random satellite failures and intermittent ISLs are a norm rather than an exception. They may be exacerbated in non-uniform LEO networks due to the complexity of ISL acquisition, tracking, and pointing under satellite mobility [17, 38]. TS-SDN relies on the control plane to react to these failures, which can be slow (multi-seconds to minutes [14, 16]) due to the disparity in timescales between intermittent failure occurrence (seconds or shorter) TS-SDN's remote control.

To address both challenges, TinyLEO streamlines state-of-the-art TS-SDNs by stabilizing high-level networking intents (demands) *by satellite-independent geography* and decoupling them from their low-level dynamic enforcements (supplies). It splits its control plane into geographic traffic engineering intents and an orbital model predictive controller to move the responsibility of handling most LEO dynamics to the data plane in §4.3.

Geographic traffic engineering intent: In practice, most operators care more about ***what*** network policy to adopt rather than ***which*** satellite(s) to enforce it. No matter how satellites move and interconnect, the geographic locations of users remain stable. Moreover, with TinyLEO's supply-demand matching in §4.1, the minimal number of available satellites over each geographic cell is also stable. Based on both invariants, TinyLEO offers stable high-level APIs for operators to customize their traffic engineering intents *by geography* without worrying about low-level LEO dynamics.

Figure 10 shows TinyLEO's geographic networking intent abstraction. It reuses each geographic cell u in §4.1 as a basic unit and assigns it two attributes: its *geographic location* L_u and *minimal number of available satellites* n_u (i.e., radio access link and ISL capacity) given by Algorithm 1. The operator can define the *geographic topology* $G(V, E, N)$ on top of these cells, where each node $u \in V$ is a geographic cell, an edge $(u, v) \in E$ exists if cell u and v needs to be connected via ISLs, and its weight $n_{u,v} \in N$ ($n_{u,v} \leq \min\{n_u, n_v\}$) represents the required number of ISLs between u and v. To support versatile policies, TinyLEO lets the operator define *any* geographic topology under only two physical constraints: the maximal per-cell satellite count ($n_u \geq \sum_{(u,v)\in V} n_{u,v}, \forall u \in E$) and inter-cell distance for visible ISL setup. On top of this geographic topology, the operator can customize versatile traffic engineering policies, such as the shortest-path routing, multipath load balancing [39], trans-oceanic traffic offloading [31], and detour away from undesirable areas [40, 41], to name a few. These routes can be modified over time, e.g., to optimize for diurnal user activities in §2.2. Each route is encoded as a list of hop-by-hop geographic cells $u \to w_1 \to w_2 \to \ldots \to v$ and passed to data plane in §4.3 for runtime enforcement.

Orbital model predictive controller (MPC): To decouple the above stable traffic engineering policy from its runtime dynamic enforcement, TinyLEO inserts a MPC shim layer [23] between the

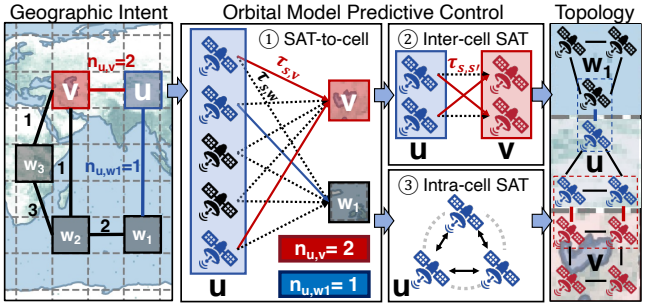


Figure 10: TinyLEO decouples geographic topological intents from enforcement via model predictive control

high-level control intent and the data plane. As a classic sequential decision-making framework, MPC lets us iteratively collect the runtime LEO network status, forecast its short-term evolutions based on these feedbacks and orbital laws, and decide near-term control actions with these predictions. Different from existing TS-SDN, this MPC is only responsible for geographic topology enforcement and does not adapt routing to all LEO physical dynamics. Instead, TinyLEO moves the latter task to its data plane for higher efficiency and lower signaling costs, as we will detail in §4.3.

Figure 10 presents how TinyLEO's MPC compiles the high-level geographic topology intent $G(V, E, N)$ to a low-level equivalent satellite network topology at runtime. It seeks to stabilize the LEO network topology as long as possible since frequent ISL reconfigurations are not desirable [17, 38]. To this end, TinyLEO adopts a three-stage stable matching. At each time slot t, for each geographic cell u, it first predicts which satellites cover it based on orbital laws. For each u's connected neighbor v with $n_{u,v} > 0$, u should allocate $n_{u,v}$ out of its n_u satellites as "gateways" toward v via ISLs to meet the demand. TinyLEO models it as a classic many-to-one stable matching [42]: It constructs a weighted bipartie graph with left nodes as each of u's satellites s, right nodes as u's connected neighboring cells $\{v|n_{u,v} > 0\}$, and the edge weight $\tau_{s,v} \geq 0$ as each satellite s's preference to be the gateway toward v. To stabilize the LEO network topology, we set $\tau_{s,v}$ as the expected ISL lifetime (larger $\tau_{s,v}$ preferred):

$$\tau_{s,v} = \frac{1}{n_v} \sum_{s' \in v} \tau_{s,s'}$$

where $\tau_{s,s'}$ is the ISL lifetime if satellite s and s' are connected and can be predicted based on their visibility and orbital laws. To stabilize this topology, TinyLEO runs the Gale-Shapley algorithm [42] to generate a stable many-to-one matching for each neighboring cell v demanding $n_{u,v}$ satellites from u. Afterward, each connected cell pair (u, v) has allocated $n_{u,v}$ satellites to each other. Then, TinyLEO runs another one-to-one stable matching [42] between these satellites in u and v using the ISL lifetime $\{\tau_{s,s'}\}_{s\in u, s'\in v}$ as satellite $s \in u$'s preference to $s' \in v$. Last, inside each cell u, TinyLEO connects all above matched satellites $s \in u$ as a ring to ensure their connectivity. This ends up with a LEO network topology that enforces the geographic topology intent $G(V, E, N)$ *and* maximizes its average ISL lifetime for topological stability.

Repairing unpredictable failures: Once TinyLEO's MPC receives runtime ISL/satellite failure reports from the data plane, it will repair

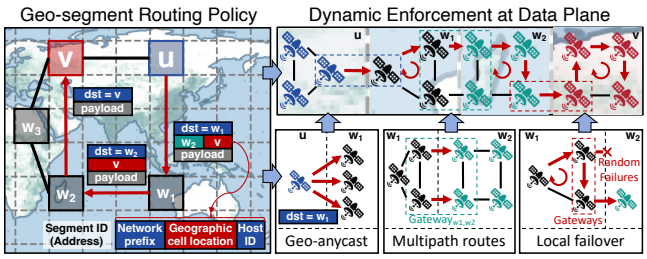


Figure 11: TinyLEO's geographic segment anycast

them to keep satisfying geographic topology intents. It checks the unsatisfied residual demands of the geographic intent after failures, incrementally runs the above three-stage matching for residual demands, and finds new ISLs/satellites to replace failed ones.

4.3 Data Plane: Geographic Segment Anycast

As explained in §4.2, TinyLEO decouples geographic routing policy intents from their runtime enforcement by satellites. To achieve this, it enhances its data plane for efficient, policy-compliant routing under the sparse LEO network's dramatic dynamics. Since each satellite's local data plane is closer to LEO dynamics than remote control planes, this separation of concern can increase the network availability and utilization, speed up failure recoveries, and lower the control overhead.

To achieve these objectives, TinyLEO builds its data plane with *geographic segment routing*. Segment routing (SR) [43–46] is an IETF-standardized source routing scheme over IPv6 or MPLS (both used in Starlink [47, 48]). In §4.2, TinyLEO encodes each network operator-defined route intent as a list of geographic cells $u \rightarrow w_1 \rightarrow w_2 \rightarrow ... \rightarrow v$ for hop-by-hop forwarding, which is a natural use case of SR. As shown in Figure 11 (cross-oceanic traffic offloading as an example), each segment in this route represents a geographic cell. It is stable and decoupled from the rapidly changing satellites. *Any* satellite covering this cell can receive a packet destined for this segment, extract the next-hop cell from the segment list, and locally forward it to *any* satellite in the next cell via in-packet location-based geographic routing [49–53]. Unlike logical IP/MPLS routing over satellite topologies, this geographic anycast defeats LEO dynamics for various benefits:

- **Near-stateless:** Unlike link-state [54], distance-vector [14], tunneling [26], and ad hoc on-demand routing [55, 56] that require every LEO satellite to exchange, update, and maintain local stateful routes, geographic segment anycast eliminates these states and their frequent updates caused by heterogeneous satellite mobility. This saves remarkable signaling overheads for resource-constrained satellites;
- **High network availability/utilization:** *Any* satellite and its ISLs in each cell can deliver traffic for fault-tolerant, geographically load-balanced services;
- **Fast failure recovery:** Upon unpredictable link/satellite outages (e.g., by solar storms and radiations), each satellite can locally reroute through any alternative ISLs/satellites in the cell *without* waiting for the remote control plane;
- **Low control overhead:** Geographic routing eliminates the need for frequent computations and updates of satellite routes over the dynamic LEO network topology;

Despite appealing, geographic routing is prone to forwarding failures due to its local greedy nature. Without global routes, it can get stuck in a "local minimum" where a node is closer to the destination than all its neighbors but is obstructed from reaching it. While this problem can be solved via face routing in 2D planar networks [49, 50], it becomes complicated [51] or even impossible [52] to solve in 3D spaces, such as non-uniform LEO networks in our case. Recent LEO-specific geographic routing schemes [18, 53] resolve this issue with domain knowledge. However, they only work for uniform LEO networks like Walker constellations [28] or multi-shell networks [18]. Unfortunately, neither assumption holds for TinyLEO's non-uniform, heterogeneous network topology.

Instead, TinyLEO's data plane ensures traffic delivery with a simple observation: Its segment-by-segment geographic anycast is navigated by its higher-layer route intents in §4.2, which is based on global topological information. Once this high-level route intent is loop-free and reachable to the destination (easily verifiable at the control plane), TinyLEO's data-plane enforcement also ensures traffic delivery as long as each hop is reachable. This condition is easily satisfied in TinyLEO's topology in §4.2: As shown in Figure 10, any two connected geographic cells in the topology intent have corresponding 1-hop ISLs in the runtime satellite topology.

Figure 11 shows TinyLEO's data-plane workflow based on these observations. Upon receiving a packet, each satellite extracts its next-hop geographic cell (segment) to forward to. If it has a direct ISL to a gateway satellite covering this next-hop cell, it immediately forwards this packet through this ISL. Otherwise, it uses the "intra-domain" ring in §4.2 to clockwise pass this packet to its next neighboring satellite inside the same cell. Since this ring connects all gateway satellites of each cell, the packet is guaranteed to eventually reach a gateway destined to the next-hop cell for successful delivery. In the worst case that this ring is disconnected by random failures, this packet will be buffered until TinyLEO's MPC in §4.2 repairs this ring to continue its delivery.

5 TinyLEO COMMUNITY TOOLKIT

In essence, TinyLEO is designed as an affordable, sustainable satellite network solution for small ISPs and countries. To achieve this goal and foster more community efforts in this direction, we have implemented all features in TinyLEO as a complete community toolkit for open research and experiments. Our community toolkit distinguishes itself from recent LEO network simulators [57–60] since it not only supports packet-level data-plane tests, but also offers upstream LEO network planning and control-plane features. It also departs from current commercial TS-SDN controllers [14–16] by offering geographic networking intent APIs and open-source orbital MPC-based control logic. This toolkit can be used to synthesize sparse LEO networks on demand, specify high-level networking intents by geography, enforce these intents at runtime with orbital MPC, and conduct per-packet emulations with hardware in the loop (see [24] for details). As shown in Figure 12, it consists of two core components:

Offline LEO network synthesizer: This tool implements TinyLEO's demand-driven LEO network sparsification in §4.1. On the network supply side, it routinely tracks the public `space-track` satellite orbital traces [61] and ITU orbit allocation databases [62] to learn

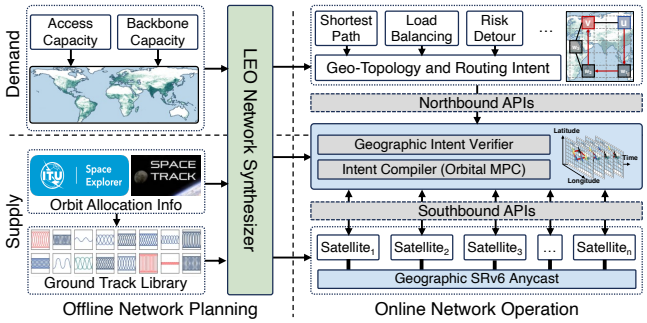


Figure 12: TinyLEO community toolkit's modules

existing satellites and space objects in orbit, extracts all unoccupied/unallocated Earth-repeat orbits accordingly, and stores their ground tracks into the texture library as TinyLEO's candidates for its network supply-demand matching. On the network demand side, this tool offers APIs to specify serviceable demands for each geographic cell in the unit of the number of satellites. Given both inputs, this synthesizer runs Algorithm 1 to output a sparse LEO network layout to meet the demands. To optimize its runtime efficiency, our implementation encodes the LEO network supplies $\mathbf{x}$, demands $\mathbf{y_t}$, and coverage matrix $\mathbf{A_t}$ using compressed sparse row (CSR) matrices [63] to speed up matrix additions/multiplications and save storage costs. We have also parallelized Algorithm 1's demand matching of all orbit candidates (line 6–7) for acceleration.

Online LEO network orchestrator: It comprises a series of control-plane and data-plane tools, including

(1) **Geographic northbound API:** It implements TinyLEO's geographic traffic engineering intent abstractions in §4.2 and a network verifier to pre-check the geographic topology connectivity and routing path reachability and loop-freedom (§4.3). It allows researchers, developers, and operators to specify and test versatile topology, routing, and traffic engineering policies in LEO satellite networks.

(2) **Orbital model predictive controller:** This shim layer realizes TinyLEO's orbital MPC in §4.2 to compile the above geographic topology intents to runtime satellite topology and adapt to random satellite/ISL failures.

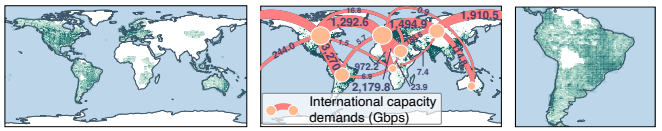
(3) **Geo-segment anycast:** This module realizes TinyLEO's data-plane geographic segment anycast in §4.3. It is built atop StarryNet [58] with three key enhancements. First, we use Linux 5.4.0 kernel's native in-kernel support for segment routing over IPv6 (SRv6 [44, 45]) to implement geographic segment anycast. Second, we add a gRPC-based southbound API agent per satellite to exchange control commands and runtime ISL/satellite status with TinyLEO's MPC (§4.2). Third, we optimize StarryNet's satellite virtualizations by customizing lightweight Linux namespace-based containers and supporting uneven LEO topologies. This allows for a larger-scale, more flexible packet-level LEO network emulation in the next section.

6 EVALUATION

We use our toolkit to assess TinyLEO's ability to sparsify the LEO network (§6.1) while still satisfying global demands and retaining comparable usability to LEO mega-constellation networks at the control plane (§6.2) and data plane (§6.3).

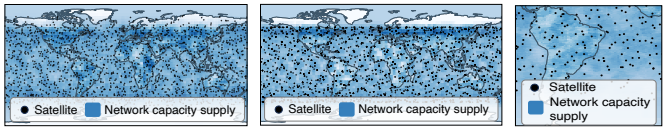
Orbital parameter	Orbital altitude H	Orbital period T	RAAN α	Inclination angle β	Total num. ground tracks
Range	423–1,873 km	92.8–124.2 min	$[-\pi, \pi]$	$[0, \pi]$	64,800

Table 1: Statistics of candidate Earth-repeat ground tracks ("textures") in our test (detailed in Appendix A)



(a) Starlink [1, 64] (b) Internet backbone [65] (c) Latin America

Figure 13: LEO network broadband demands in our test



(a) Starlink (b) Internet backbone (c) Latin America

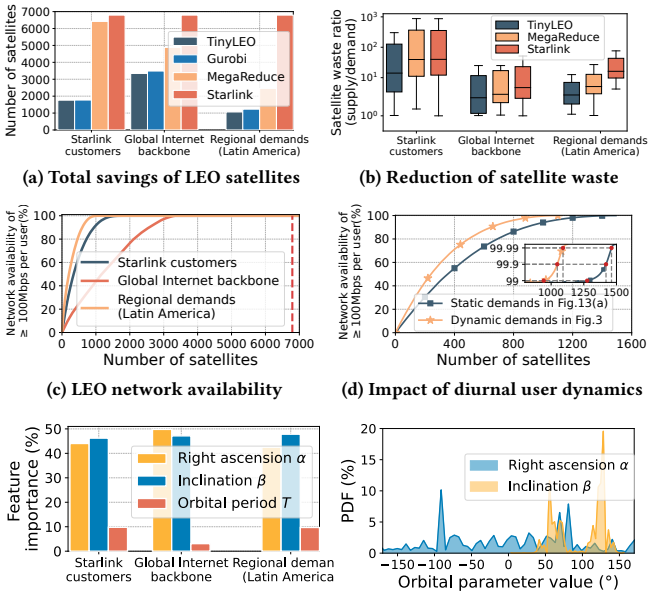
Figure 14: TinyLEO on-demand sparse LEO network

6.1 On-demand LEO Network Sparsification

We first assess how many satellites TinyLEO can save over LEO mega-constellations under the same large-scale broadband demands. Following Starlink's satellite specifications [1, 17], we consider each LEO satellite with 3 ISLs (each with 200 Gbps capacity) and 1 radio access link for users (with 96 Gbps capacity). We assume each user terminal requires a 100 Mbps downlink speed [1], implying that each satellite can satisfy up to 960 users' concurrent demands.

We run TinyLEO's sparse network synthesizer in §5 with 64,800 candidate ground tracks in Table 1 (exemplified in Table 2 in Appendix A) to match 3 representative real demands on 4,050 geographic cells: (1) **Starlink (Figure 13a):** We extract Starlink's global customer distributions based on its official report [1, 64] and scale it by its total radio access link capacity (652 Tbps from 6,793 satellites) so that every user can gain a 100 Mbps downlink speed [1]. We consider its real diurnal fluctuations in Figure 3; (2) **International Internet backbone (Figure 13b):** We assess the potential of using LEO networks as a backup for Internet's submarine cables [65] under disasters (e.g., Earthquakes), while still retaining the same inter-regional capacity; (3) **Regional demands (Figure 13c):** Small ISPs/countries may only want small LEO networks for regional demands (§7). We exemplify this demand with Starlink's Latin America customers.

We compare TinyLEO with three candidate solutions: (1) **Starlink**, the leading LEO mega-constellation network with 6,793 satellites in 5 orbital shells as of 2025.01 (Figure 1a). We extract its real distribution of satellites from their public orbital trajectories in `space-track` [61]; (2) **MegaReduce** [66, 67], which fine-tunes uniform LEO mega-constellations by iteration to save satellites; (3) **Gurobi**, which implements exact solution of Equation 2–4 using Gurobi v12.0 [68] and runs it over a high-performance workstation with 2 Intel Xeon Gold 6430 CPUs (2.1–3.4 GHz, 32 cores, 64 threads, and 60MB cache for each) and 1 TB DDR5-4800 RAMs for its multi-thread acceleration. Despite such powerful computing, this workstation still cannot finish a single run of this integer programming within 2 months. We truncate each run after 2 months and present the minimal LEO network size it obtains.



(a) Total savings of LEO satellites

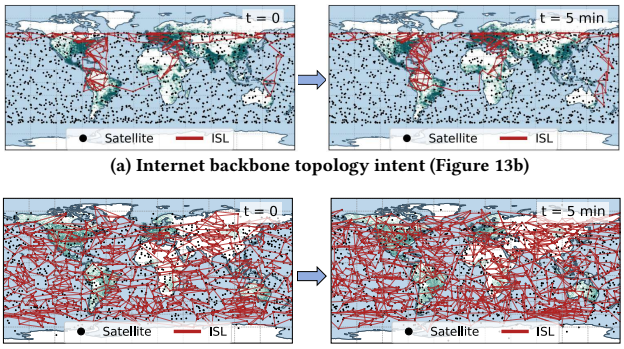
(b) Reduction of satellite waste

(c) LEO network availability

(d) Impact of diurnal user dynamics

(e) Impact and distribution of diverse orbital parameters in TinyLEO

Figure 15: LEO network sparsification in TinyLEO



(a) Internet backbone topology intent (Figure 13b)

(b) Geographic mesh grid topology intent

Figure 16: Dynamic enforcement of topological intents

Overall LEO network sparsification: Figure 14 shows TinyLEO's sparse LEO network layout to match the demands in Figure 13 (demo video available at [69]), and Figure 15 compares its network size with other solutions. Compared to Starlink's 6,793 satellites in Figure 1a, TinyLEO shrinks its network size to 1,763 (3.9×), 3,344 (2.0×), and 1,066 (6.4×) satellites to meet the same broadband demands for Starlink's customers, Internet backbone backup, and Latin America in Figure 13, respectively. This saving is attributed to TinyLEO's supply-demand matching for lower LEO satellite waste (Figure 15b). Its non-uniform layout can better match uneven demands than the uniform MegaReduce and hence save much more satellites. TinyLEO's sparse matching approximation even saves more satellites than the Gurobi-based truncated exact solution *and* is much faster: TinyLEO completes each task in Figure 13 in 6.5–7.7 hours, while Gurobi cannot finish any of them in 2 months.

More savings with flexible network availability: Our experiments further find that the marginal benefits of adding more satellite diminish over the LEO network scale, as shown in Figure 15c–15d. This is attributed to the fact that when most demands are satisfied, meeting the residual demands (usually in hotspots) becomes more expensive: LEO satellites added to meet these residual demands are more likely wasted when they later move to other already-satisfied areas. If the network operator slightly lowers its network availability goal (e.g., 99% in Starlink [17]), TinyLEO can further shrink the LEO network size to 1,391 (4.9×), 3,184 (2.1×), 865 (7.9×) satellites for demands in Figure 13, respectively.

Impact of diurnal user dynamics: Figure 15d compares the size of TinyLEO's LEO network to meet Starlink customers with/without considering their diurnal activities in Figure 3b. If this temporal demand dynamics is considered, TinyLEO does not need to match peak demands everywhere and hence saves more satellites. In our experiments with Cloudflare's DNS-based Starlink user activity measurements over 2 days at 15-minute intervals across cities, TinyLEO can save 271 additional satellites (18.5%) compared to its savings under the static demand baseline in Figure 13a. When combined with flexible network availability (e.g., 99% in Starlink), the reduction grows to 333 satellites, achieving a 26% reduction relative to the static demand baseline.

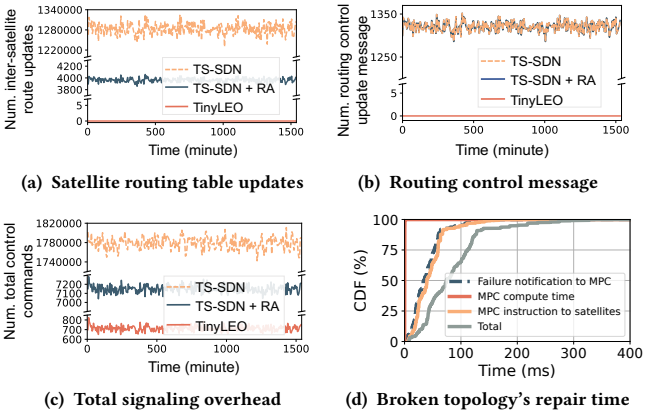
Impact of orbital parameters: Figure 15e characterizes the diverse orbital parameters in TinyLEO's network layout in Figure 14. It confirms that TinyLEO's savings stem from its use of diverse yet sparse orbits to align with the uneven network demands. We also follow [70] to calculate the importance score of each orbital parameter in this process. As shown in Figure 15e, the orbital inclination (β) and right ascension (α) contribute most to match latitudinal and longitudinal uneven demands. Instead, the orbital period T's impact is mixed: While helpful with uneven temporal demand satisfaction, heterogeneous orbital periods also complicate temporal matching since $T_{max} = LCM(T_1, T_2, ..., T_n)$. We have not observed significant satellite savings with them.

6.2 Control-Plane: Stable Intent + Orbital MPC

We next deploy our TinyLEO orchestrator in §5 into a large-scale virtualized, hardware-in-the-loop testbed to evaluate its network usability under complex LEO physical dynamics. This testbed employs a variant of the StarryNet platform [58] to emulate 1,741 TinyLEO satellites in Figure 14a on a high-performance computing cluster for packet-level experiments. Each virtual satellite runs as an independent lightweight Linux container with complete TinyLEO features in Figure 12. It equips 3 ISLs and 1 radio access link [1, 17]. All satellites are orchestrated by TinyLEO's terrestrial controller in Figure 12 via southbound APIs. We evaluate TinyLEO's control plane in this section and its data plane in the next section.

Enforcement of geographic topology intents: Figure 16 showcases TinyLEO's dynamic enforcement of the Internet backbone intent in Figure 13b and a geographic mesh topology intent over the sparse LEO network in Figure 14a. For both intents, despite extreme satellite mobility, TinyLEO's orbital MPC in §4.2 can dynamically enforce the upper-layer stable topological intent by compiling it into time-evolving LEO physical network topologies. The geographic topology intent remains fixed in this process, allowing the network operator to optimize its traffic engineering policy on a stable basis. The results for other topological intents are similar.

Control-plane overhead reductions: Figure 17 measures the number of signaling messages from the controller to all satellites under the time-evolving topology in Figure 16b. We compare TinyLEO



(a) Satellite routing table updates (b) Routing control message

(c) Total signaling overhead (d) Broken topology's repair time

Figure 17: TinyLEO's control-plane efficiency and cost

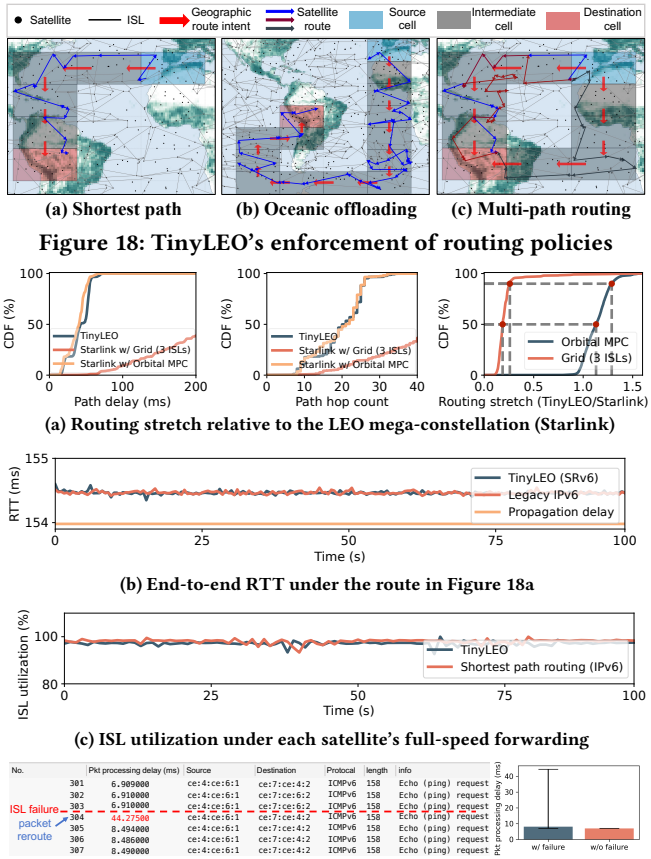
with Aalyria's open-source TS-SDN [37] that implements solutions in [14–16] for direct dynamic control of LEO satellite topology and routing. As shown in Figure 17a, TinyLEO's stable topology intent prevents frequent updates of geographic routes under LEO dynamics, thus eliminating inter-satellite routing control updates. Instead, TS-SDN suffers from high inter-satellite routing control costs even with route aggregation (RA), because TinyLEO's non-uniform layout makes the satellite routing less hierarchical. In TinyLEO's sparse LEO network in Figure 14a, this results in 1–3 orders of magnitude saving of total signaling messages for TinyLEO compared to TS-SDN. The remaining control messages in TinyLEO are mostly for each satellite's local ISL updates to enforce geographic topology intents. They are 10× fewer than those in Figure 16 since each ISL change will incur multiple inter-satellite route changes.

Repairing unpredictable failures: Figure 17d demonstrates the time TinyLEO's orbital MPC takes to repair the topology by injecting 1,000 random link failures. TinyLEO repairs each of them within 83.8 ms on average, most of which arises from the inevitable round-trip time of the failure report and repair command between the satellite and the controller (83.5 ms on average). This latency is comparable to or shorter than TS-SDN, which takes 100 ms [16] to minutes [14] for route failure recovery. Meanwhile, TinyLEO's data plane locally tolerates these failures for faster data delivery, as we will evaluate in the next section.

6.3 Data Plane: Geographic Segment Anycast

We last use the same testbed and experimental setup as §6.2 to evaluate TinyLEO's data-plane functions in §4.3.

Routing policy enforcement: Figure 18 showcases three satellite routes at TinyLEO's data plane in §4.3 when it is instructed to enforce the shortest-path routing, cross-oceanic traffic offloading, and multi-path routing policies. In all cases, TinyLEO correctly enforces these geographic routing intents with guaranteed packet delivery. Its SRv6-based geographic segment list embeds these policies into the packet header. Each satellite follows this in-packet policy for hop-by-hop anycast to tolerate LEO dynamics. This geo-segment anycast stabilizes network operators' routing and traffic engineering policies, who can then optimize these high-level policies without worrying about low-level LEO dynamics. It also shows the TinyLEO toolkit's value for the community to design and evaluate various routing schemes.



(a) Shortest path (b) Oceanic offloading (c) Multi-path routing

Figure 18: TinyLEO's enforcement of routing policies

(a) Routing stretch relative to the LEO mega-constellation (Starlink)

(b) End-to-end RTT under the route in Figure 18a

(c) ISL utilization under each satellite's full-speed forwarding

(d) Each satellite's local rerouting under random ISL failures

Figure 19: TinyLEO's data-plane performance

Data-plane performance: We first evaluate TinyLEO's routing performance after its network sparsification. At first glance, the sparse nature of TinyLEO seems to force longer routing latencies than LEO mega-constellations: While removing satellites in low-demand areas, TinyLEO may also destroy shortcuts for certain origin-destination pairs and force their traffic to detour. But recall in §4.1–4.2 that TinyLEO can take the backbone traffic matrix demand into account when sparsifying the LEO network. The operator can specify each origin-destination pair's traffic demand and intended geographic routing path (e.g., the shortest path), aggregate all these pairs' traffic demands hop by hop based on both information to form the geographic cell demands $\mathbf{y_t}$, and use $\mathbf{y_t}$ as TinyLEO's input to synthesize the sparse LEO network based on Algorithm 1. This sparse LEO network can still retain each origin-destination pair's routing path intent with the help of TinyLEO's control/data-plane functions in §4.2–4.3, hence maintaining low routing stretches relative to LEO mega-constellations. In addition, TinyLEO's hop-by-hop geographic anycast ensures that each hop's additional delay is bounded by its corresponding geographic cell's size, hence further mitigating the overall routing path stretches.

To validate this hypothesis, we run TinyLEO over the backbone traffic demand in Figure 13b and the shortest path intent for all its origin-destination pairs. We compare each origin-destination pair's routing path length (in hop counts and propagation delays) in TinyLEO and Starlink under the same setting. Since Starlink does

not disclose its topology, we consider two candidate solutions for it: the standard 3-ISL grid topology (i.e., each satellite connects to its two nearest intra-orbit neighbors and one nearest inter-orbit neighbor), and the orbital MPC-based topology compilation in §4.2. As shown in Figure 19a, TinyLEO achieves low routing stretches (1.29 at 90th percentile, 1.63 at maximum) relative to Starlink using the same orbital MPC for topology setup. Surprisingly, its routing paths can be even shorter than Starlink using the standard grid topology, since its orbital MPC allows for cross-orbit ISLs to reduce hop counts and path lengths. Instead, the standard grid topology only connects each satellite to its nearest neighbors, hence forcing more hop counts in end-to-end paths (also observed in [71]).

We next characterize TinyLEO's packet forwarding performance. Figure 19 measures TinyLEO's end-to-end RTT and throughput when enforcing the shortest-path routing in Figure 18. The results under other high-level routing policies are similar. Due to its greedy hop-by-hop geographic routing nature, TinyLEO approximates the shortest path under the high-level routing policy constraints. Its use of Linux's in-kernel SRv6 support allows each satellite router to process and forward packets at a comparable efficiency to legacy IPv6 and almost full-speed link utilization.

Resiliency to unpredictable failures: We inject random ISL and satellite failures into our testbed and assess TinyLEO's robustness to them. Once the satellite topology remains connected under these failures, Figure 19d confirms that TinyLEO's data-plane anycast can locally bypass them and reroute through other available ISLs between geographic cells. This only incurs 13.6–44.3 ms delays (i.e., 1–3 orders of magnitude faster than TS-SDN [14, 16]). In case the topology is disconnected by failures, the satellite will buffer the packets, wait for TinyLEO's control plane for repair (83.8 ms on average based on Figure 17d), and continue forwarding.

7 DISCUSSION

Radio link optimizations with TinyLEO: While primarily designed for sparse LEO satellite networking, TinyLEO can also help with their last-hop radio access links' spectrum management, interference mitigation, resource scheduling, and more tasks. It can sparsify satellite radio links covering each area to alleviate their overlapping, interference, and resource competition. Its stable geographic intents in §4.2 can also mask complex LEO mobility to simplify these tasks as less dynamic optimizations like terrestrial mesh networks.

Deployment incentives: TinyLEO may be more attractive to small ISPs and countries since it lowers their barriers to entering the market. Regulators (e.g., ITU) may also welcome it since it can stimulate the LEO network market with more players, unleash orbit resources, and relieve space congestion. While LEO mega-constellation operators (e.g., Starlink) may not be incentivized to shrink their network scale due to their market hold, they can still adopt TinyLEO's control and data planes for higher network efficiency. Their satellites can also be safer if others adopt TinyLEO with low orbital congestion.

LEO network decentralization: Besides TinyLEO, another emerging vision of alleviating monopoly, capital costs, and satellite wastes is to build a decentralized multi-party LEO network [72, 73] for multi-tenant access [74], backhaul rental [32], and multi-tasking [75]. Fulfilling this great long-term vision is nontrivial since it calls for substantial efforts of trust establishment, heterogeneous internetworking, and protocol standardization among global players with inconsistent interests. TinyLEO is free of these issues due to its single-entity network nature. Instead, it can boost this LEO network decentralization by enabling more entrants and letting each contribute its (regional) networks at low costs (Figure 14c).

Limitations and future work: As an initial attempt at small LEO networks for global-scale demands, TinyLEO has some inevitable limitations that deserve future work. First, TinyLEO shrinks the satellite network at the cost of more LEO dynamics, hence leading to more challenges for network availability, efficiency, and resiliency. While our current design has significantly mitigated these issues, it currently relies on Earth-repeat ground tracks. It would be nice to unlock ground tracks beyond Earth-repeat ones for more stable and sparser LEO networks. Second, TinyLEO's control plane can be decentralized to facilitate the above multi-party LEO network (akin to BGP). Last but not least, TinyLEO's data plane should be enhanced for future deterministic networking (DetNet).

8 RELATED WORK

The recent surge of LEO mega-constellations has excited academia and industry for satellite networking R&D. Most of these works have centered around LEO mega-constellations, including but not limited to space topology designs [13, 71, 76], long-range radio optimizations [77–80], global collision and interference-free medium access [57, 81, 82], large-scale routing [18, 31, 39, 39, 53, 83], dynamic transport control [84–86], and security [87–90]. It was not until recently that the networking community started to rethink the concerns of LEO mega-constellations and explore decentralized LEO networks [32, 72–75] in §7 as alternatives. Instead, TinyLEO complements these efforts by directing shrinking the LEO network size while still satisfying network demands at scale.

Different from pure LEO constellation design [66] and network-only optimizations [14–17], TinyLEO explores a *co-design* of offline network planning and online control/data plane operation. As shown in §4–§6, this is co-design crucial to both save more satellites *and* make the network usable.

9 CONCLUSION

We propose TinyLEO, an alternative to LEO mega-constellation networks to meet global-scale demands with sparse satellites. TinyLEO is motivated by the insight that most satellites in mega-constellations are wasted due to their mismatch with uneven global network demands. Hence, it sparsifies LEO networks via spatiotemporal supply-demand matching and makes it usable via control/data-plane refinements. We hope TinyLEO can inspire more community efforts to enable affordable LEO satellite networks for small ISPs and countries, democratize this market with more players, and strive for sustainable "Internet from space" for all humanity.

Acknowledgment: We thank our shepherd, Dr. Dave Levin, and all reviewers for their valuable feedback. This work is supported by the National Key Research and Development Plan of China (2022YFB3105201), National Natural Science Foundation of China (62202261), and key fund of National Natural Science Foundation of China (62132009). Hewu Li is the corresponding author.

REFERENCES

[1] SpaceX. Starlink 2024 Progress Report. https://stories.starlink.com/. Also available at https://tinyurl.com/4c22k7d8.

[2] OneWeb constellation. https://www.oneweb.world/, Retrieved 2025-01-14.

[3] Xinhua. China's New Mega-constellation Marks Milestone in Satellite Internet. https://english.www.gov.cn/news/202408/09/content_WS66b55ad9c6d0868f4e8e9cf8.html, 2024.

[4] Amazon. Project Kuiper. https://www.aboutamazon.com/what-we-do/devices-services/project-kuiper, Retrieved 2025-01-14.

[5] Reuters. Musk Says May Need $30 Billion to Keep Starlink in Orbit. https://www.reuters.com/business/aerospace-defense/musk-sees-starlink-winning-500000-customers-next-12-months-2021-06-29/, 2021.

[6] The Deep Dive. Amazon's Satellite Network Faces Skyrocketing Costs in $20 Billion Gamble. https://thedeepdive.ca/amazons-satellite-network-faces-skyrocketing-costs-in-20-billion-gamble/, 2024.

[7] Reuters. FCC Chair Wants More Competition to SpaceX's Starlink Unit. https://www.reuters.com/technology/space/fcc-chair-wants-more-competition-spacexs-starlink-unit-2024-09-11/, 2024.

[8] Le Monde. Brazil Seeks to Break Starlink's Monopoly. https://www.lemonde.fr/en/economy/article/2024/11/18/brazil-seeks-to-break-starlink-s-monopoly_6733222_19.html, 2024.

[9] EL PAÍS. Starlink Satellites: Elon Musk's Other Destabilizing Power in Africa. https://english.elpais.com/economy-and-business/2024-12-06/starlink-satellites-elon-musks-other-destabilizing-power-in-africa.html, 2024.

[10] SpaceX's Approach to Space Sustainability and Safety. https://www.spacex.com/updates/index.html, Feb 2022.

[11] ESA. Space Debris by the Numbers. https://www.esa.int/Safety_Security/Space_Debris/Space_debris_by_the_numbers, Retrieved 2025-01-19.

[12] Donald J Kessler, Nicholas L Johnson, JC Liou, and Mark Matney. The Kessler Syndrome: Implications to Future Space Operations. *Advances in the Astronautical Sciences*, 137(8):2010, 2010.

[13] Yuanjie Li, Hewu Li, Wei Liu, Lixin Liu, Wei Zhao, Yimei Chen, Jianping Wu, Qian Wu, Jun Liu, Zeqi Lai, and Han Qiu. A Networking Perspective on Starlink's Self-Driving LEO Mega-Constellation. In *The 29th International Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2023.

[14] Frank Uyeda, Marc Alvidrez, Erik Kline, Bryce Petrini, Brian Barritt, David Mandle, and Chandy Aswin Alexander. SDN in the Stratosphere: Loon's Aerospace Mesh Network. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*. ACM, 2022.

[15] Brian Barritt and Vint Cerf. Loon SDN: Applicability to NASA's next-generation space communications architecture. In *2018 IEEE Aerospace Conference*, pages 1–9. IEEE, 2018.

[16] Aalyria. SpaceTime: Orchestrating the next generation of aerospace networks. https://www.aalyria.com/spacetime, 2024.

[17] Travis R Brashears. Achieving 99% Link Uptime on a Fleet of 100G Space Laser Inter-Satellite Links in LEO. In *Free-Space Laser Communications XXXVI*, volume 12877, page 1287702. SPIE, 2024.

[18] Yuanjie Li, Lixin Liu, Hewu Li, Wei Liu, Yimei Chen, Wei Zhao, Jianping Wu, Qian Wu, Jun Liu, and Zeqi Lai. Stable Hierarchical Routing for Operational LEO Networks. In *Annual International Conference On Mobile Computing And Networking (MobiCom)*. ACM, 2024.

[19] Yonina C Eldar and Gitta Kutyniok. *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.

[20] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable Signal Recovery From Incomplete and Inaccurate Measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[21] Radu Berinde and Piotr Indyk. Sparse Recovery Using Sparse Random Matrices. *LATIN 2010: Theoretical Informatics*, page 157, 2010.

[22] Deanna Needell and Joel A Tropp. CoSaMP: Iterative Signal Recovery from Incomplete and Inaccurate Samples. *Communications of the ACM*, 53(12):93–100, 2010.

[23] James Blake Rawlings, David Q Mayne, Moritz Diehl, et al. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing Madison, WI, 2017.

[24] TinyLEO Community Toolkit:. https://tinyleo-toolkit.github.io/TinyLEO/.

[25] SpaceX year review company talk. https://x.com/SpaceX/status/1745941814165815717, 2024.

[26] Chen Chen, Pavel Chikulaev, Sergii Ziuzin, David Sacks, Peter Worters, Darshan Purohit, Yashodhan Dandekar, Vladimir Skuratovich, Andrei Pushkin, and Phillip Barber. Low Latency Schedule-driven Handovers, 2023. US Patent 11,729,684.

[27] Cloudflare Radar. SpaceX Starlink's Traffic Worldwide. https://radar.cloudflare.com/traffic/as14593, year = Retrieved 2025-01-01.

[28] Walker Satellite Constellation. https://en.wikipedia.org/wiki/Satellite_constellation#Walker_Constellation, Retrieved 2025-01-14.

[29] Mike Puchol. Modeling Starlink Capacity. https://mikepuchol.com/modeling-starlink-capacity-843b2387f501, 2022.

[30] SpaceX. Application for Fixed Satellite Service. https://fcc.report/IBFS/SAT-MOD-20200417-00037/2274316.pdf, 2020.

[31] Mark Handley. Using Ground Relays for Low-Latency Wide-Area Routing in Megaconstellations. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks (HotNets)*, pages 125–132, 2019.

[32] Ali Abedi, Joshua Sanz, Mariya Zheleva, and Anant Sahai. From Foe to Friend: The Surprising Turn of Mega Constellations in Radio Astronomy. In *Proceedings of the 23rd ACM Workshop on Hot Topics in Networks (HotNets)*, pages 10–16, 2024.

[33] Xiaofeng Fu, Meiping Wu, and Yi Tang. Design and Maintenance of Low-Earth Repeat Ground Track Successive-Coverage Orbits. *Journal of Guidance, Control, and Dynamics*, 35(2):686–691, 2012.

[34] Mahdi Jafari Nadoushan and Nima Assadian. Repeat Ground Track Orbit Design with Desired Revisit Time and Optimal Tilt. *Aerospace Science and technology*, 40:200–208, 2015.

[35] Integer Linear Programming. https://en.wikipedia.org/wiki/Linear_programming#Integral_linear_programs, Retrieved 2025-01-14.

[36] Airong Liu, Xiaoli Xu, Yongqing Xiong, and Shengxian Yu. Maneuver Strategies of Starlink Satellite based on SpaceX-released Ephemeris. *Advances in Space Research*, 2024.

[37] Aalyria. SpaceTime APIs. https://github.com/aalyria/api, Retrieved 2025-01-14.

[38] Hemani Kaushal, VK Jain, Subrat Kar, Hemani Kaushal, VK Jain, and Subrat Kar. Acquisition, Tracking, and Pointing. *Free Space Optical Communication*, pages 119–137, 2017.

[39] Mark Handley. Delay is Not an Option: Low Latency Routing in Space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks (HotNets)*, pages 85–91, 2018.

[40] Chenwei Gu, Qian Wu, Zeqi Lai, Hewu Li, Jihao Li, Weisen Liu, Qi Zhang, Jun Liu, and Yuanjie Li. STARVERI: Efficient and Accurate Verification for Risk-Avoidance Routing in LEO Satellite Networks. In *International Conference on Network Protocols (ICNP)*. IEEE, 2024.

[41] Dave Levin, Youndo Lee, Luke Valenta, Zhihao Li, Victoria Lai, Cristian Lumezanu, Neil Spring, and Bobby Bhattacharjee. Alibi Routing. *ACM SIGCOMM*, 45(4):611–624, 2015.

[42] Stable Marriage Problem. https://en.wikipedia.org/wiki/Stable_marriage_problem, Retrieved 2025-01-14.

[43] Cisco Systems. Segment Routing. https://www.segment-routing.net/, Retrieved 2025-01-14.

[44] L Ginsberg, B Decraene, S Litkowski, and R Shakir. IETF RFC 8402: Segment Routing Architecture. 2018.

[45] C Filsfils, D Dukes, S Previdi, J Leddy, S Matsushima, and D Voyer. IETF RFC 8754: IPv6 Segment Routing Header (SRH). 2020.

[46] Ahmed Bashandy, Clarence Filsfils, Stefano Previdi, Bruno Decraene, Stephane Litkowski, and Rob Shakir. IETF RFC 8660: Segment Routing with the MPLS Data Plane. 2019.

[47] SpaceX Starlink. What IP Address does Starlink Provide? https://www.starlink.com/support/article/1192f3ef-2a17-31d9-261a-a59d215629f4, Retrieved 2025-01-25.

[48] Jianping Pan, Jinwei Zhao, and Lin Cai. Measuring the Satellite Links of a LEO Network. In *IEEE International Conference on Communications (ICC)*, 2024.

[49] Brad Karp and Hsiang-Tsung Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proceedings of the 6th Annual international conference on Mobile computing and networking (MobiCom)*, 2000.

[50] Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. Geographic routing made practical. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 217–230. USENIX Association, 2005.

[51] Simon S Lam and Chen Qian. Geographic Routing in d-dimensional Spaces with Guaranteed Delivery and Low Stretch. *IEEE/ACM Transactions on Networking*, 21(2):663–677, 2012.

[52] Stephane Durocher, David Kirkpatrick, and Lata Narayanan. On routing with guaranteed delivery in three-dimensional ad hoc wireless networks. In *International Conference on Distributed Computing and Networking*, pages 546–557. Springer, 2008.

[53] Valentin Hartig, Marcin Bosk, Nitinder Mohan, and Paulo Mendes. Segment Routing based on Geographic Checkpoints. In *Proceedings of the 2nd International Workshop on LEO Networking and Communication (LEO-NET)*, pages 25–30, 2024.

[54] Thomas Clausen and Philippe Jacquet. IETF RFC 3626: Optimized Link State Routing Protocol (OLSR), 2003.

[55] David B Johnson, David A Maltz, Josh Broch, et al. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Ad hoc networking*, 5(1):139–172, 2001.

[56] Charles Perkins, Elizabeth Belding-Royer, and Samir Das. IETF RFC 3561: Ad hoc On-demand Distance Vector (AODV) Routing, 2003.

[57] Jayanth Shenoy, Om Chabra, Tusher Chakraborty, Suraj Jog, Deepak Vasisht, and Ranveer Chandra. CosMAC: Constellation-Aware Medium Access and Scheduling for IoT Satellites. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 724–739, 2024.

[58] Zeqi Lai, Hewu Li, Yangtao Deng, Qian Wu, Jun Liu, Yuanjie Li, Jihao Li, Lixin Liu, Weisen Liu, and Jianping Wu. StarryNet: Empowering Researchers to Evaluate Futuristic Integrated Space and Terrestrial Networks. In *Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX, 2023.
[59] Zeqi Lai, Hewu Li, and Jihao Li. StarPerf: Characterizing Network Performance for Emerging Mega-Constellations. In *28th International Conference on Network Protocols (ICNP)*, pages 1–11. IEEE, 2020.
[60] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. Exploring the "Internet from space" with Hypatia. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, pages 214–229, 2020.
[61] Space Track. https://www.space-track.org, Retrieved 2025-01-16.
[62] ITU-R. ITU Space Explorer Database. https://www.itu.int/en/ITU-R/space/ITUSpaceExplorer/Pages/default.aspx, Retrieved 2025-01-16.
[63] Aydin Buluç, Jeremy T Fineman, Matteo Frigo, John R Gilbert, and Charles E Leiserson. Parallel Sparse Matrix-Vector and Matrix-Transpose-Vector Multiplication Using Compressed Sparse Blocks. In *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures (SPAA)*, pages 233–244, 2009.
[64] SpaceX. All Hands Meeting on the Company Update. https://twitter.com/SpaceX/status/1745941814165815717, 2024.
[65] TeleGeography. Global Internet Map. https://global-internet-map-2022.telegeography.com/, 2022.
[66] Zeqi Lai, Yibo Wang, Hewu Li, Qi Zhang, Yunan Hou, Jun Liu, and Yuanjie Li. Your Mega-Constellations Can be Slim: A Cost-Effective Approach for Constructing Survivable and Performant LEO Satellite Networks. In *IEEE International Conference on Computer Communications (INFOCOM'24)*. IEEE, 2024.
[67] Open-source code of MegaReduce. https://github.com/SpaceNetLab/MegaReduce, Retrieved 2024-12-01.
[68] Gurobi Optimizer. https://www.gurobi.com/, Retrieved 2025-01-17.
[69] TinyLEO demo video. https://youtu.be/fcwnI78tluw, 2025.
[70] Feature Importance with Random Forests. https://www.geeksforgeeks.org/feature-importance-with-random-forests/, Retrieved 2025-01-23.
[71] Debopam Bhattacherjee and Ankit Singla. Network Topology Design at 27,000 km/hour. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies (CoNEXT)*. ACM, 2019.
[72] Seoyul Oh and Deepak Vasisht. A call for decentralized satellite networks. In *Proceedings of the 23rd ACM Workshop on Hot Topics in Networks (HotNets)*, pages 25–33, 2024.
[73] Veronica Muriga, Swarun Kumar, Akshitha Sriraman, and Assane Gueye. A Road map for the Democratization of Space-Based Communications. In *Proceedings of the 23rd ACM Workshop on Hot Topics in Networks (HotNets)*, pages 17–24, 2024.
[74] Lixin Liu, Yuanjie Li, Hewu Li, Jiabo Yang, Wei Liu, Jingyi Lan, Yufeng Wang, Jiarui Li, Jianping Wu, Qian Wu, Jun Liu, and Zeqi Lai. Democratizing Direct-to-Cell Low Earth Orbit Satellite Networks. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX, 2024.
[75] Demi Lei and Ahmed Saeed. Do We Need a Million Satellites in Orbit? Constellation-as-a-Service with Modular Satellites: Challenges and Opportunities. In *Proceedings of the 2nd International Workshop on LEO Networking and Communication (LEO-NET)*, pages 61–66, 2024.
[76] Wenyi Zhang, Zihan Xu, and Sangeetha Abdu Jyothi. An In-Depth Investigation of LEO Satellite Topology Design Parameters. In *Proceedings of the 2nd International Workshop on LEO Networking and Communication (LEO-NET)*, pages 1–6, 2024.
[77] Hao Pan, Lili Qiu, Bei Ouyang, Shicheng Zheng, Yongzhao Zhang, Yi-Chao Chen, and Guangtao Xue. PMSat: Optimizing Passive Metasurface for Low Earth Orbit Satellite Communication. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–15, 2023.
[78] Kun Woo Cho, Yasaman Ghasempour, and Kyle Jamieson. Towards Dual-band Reconfigurable Metasurfaces for Satellite Networking. In *Proceedings of the 21st ACM Workshop on Hot Topics in Networks (HotNets)*, pages 17–23, 2022.
[79] Bill Tao, Maleeha Masood, Indranil Gupta, and Deepak Vasisht. Transmitting, Fast and Slow: Scheduling Satellite Traffic through Space and Time. In *The 29th International Conference on Mobile Computing and Networking (MobiCom'23)*. ACM, 2023.
[80] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. L2D2: Low Latency Distributed Downlink for LEO Satellites. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*, pages 151–164, 2021.
[81] Vaibhav Singh, Tusher Chakraborty, Suraj Jog, Om Chabra, Deepak Vasisht, and Ranveer Chandra. Spectrumize: Spectrum-efficient Satellite Networks for the Internet of Things. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2024.
[82] Haoge Jia, Zuyao Ni, Chunxiao Jiang, Linling Kuang, and Jianhua Lu. Uplink Interference and Performance Analysis for Megasatellite Constellation. *IEEE Internet of Things Journal*, 9(6):4318–4329, 2021.
[83] Debopam Bhattacherjee, Waqar Aqeel, Ilker Nadi Bozkurt, Anthony Aguirre, Balakrishnan Chandrasekaran, P Brighten Godfrey, Gregory Laughlin, Bruce Maggs, and Ankit Singla. Gearing up for the 21st Century Space Race. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks (HotNets)*, 2018.
[84] Bin Hu, Xumiao Zhang, Qixin Zhang, Nitin Varyani, Z Morley Mao, Feng Qian, and Zhi-Li Zhang. LEO Satellite vs. Cellular Networks: Exploring the Potential for Synergistic Integration. In *Companion of the 19th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, pages 45–51, 2023.
[85] Xuyang Cao and Xinyu Zhang. SaTCP: Link-Layer Informed TCP Adaptation for Highly Dynamic LEO Satellite Networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1–10, 2023.
[86] George Barbosa, Sirapop Theeranantachai, Beichuan Zhang, and Lixia Zhang. A Comparative Evaluation of TCP Congestion Control Schemes over Low-Earth-Orbit (LEO) Satellite Networks. In *Proceedings of the 18th Asian Internet Engineering Conference*, pages 105–112, 2023.
[87] James Pavur, Daniel Moser, Martin Strohmeier, Vincent Lenders, and Ivan Martinovic. A Tale of Sea and Sky on the Security of Maritime VSAT Communications. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1384–1400. IEEE, 2020.
[88] Giacomo Giuliari, Tommaso Ciussani, Adrian Perrig, Ankit Singla, and E Zurich. ICARUS: Attacking Low Earth Orbit Satellite Networks. In *USENIX Annual Technical Conference (ATC)*, pages 317–331, 2021.
[89] Joshua Smailes, Sebastian Köhler, Simon Birnbach, Martin Strohmeier, and Ivan Martinovic. Watch this Space: Securing Satellite Communication Through Resilient Transmitter Fingerprinting. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 608–621, 2023.
[90] David Koisser, Richard Mitev, Marco Chilese, and Ahmad-Reza Sadeghi. Don't Shoot the Messenger: Localization Prevention of Satellite Internet Users. In *The 45th IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2024.

A EARTH-REPEAT GROUND TRACKS

Table 2 exemplifies some of the Earth-repeat ground tracks ("textures") used in TinyLEO's evaluations in §6. For each candidate ground track's orbit in Table 2, our experiments consider all its variants with inclinations β ranging between [−90°, 90°] and right ascensions α between [-180°, 180°]. More Earth-repeat ground tracks can be generated by enumerating the combinations of p and q ($p, q \in \mathbb{N}^+$) in Equation 1. This generates 64,800 heterogeneous Earth-repeat ground tracks for TinyLEO to synthesize sparse LEO networks in §6.1–6.3.

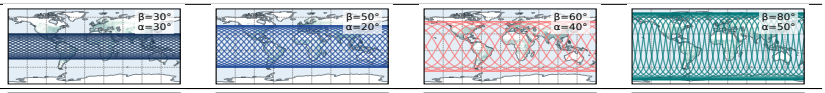
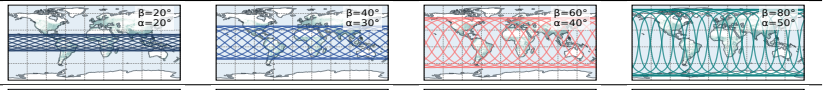
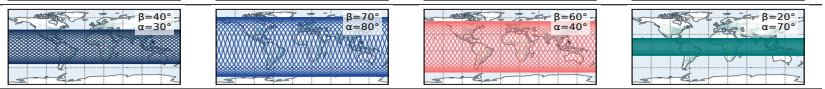
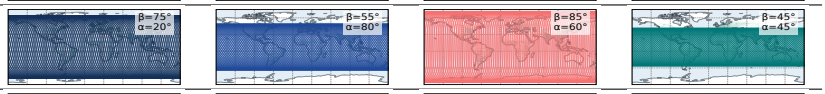
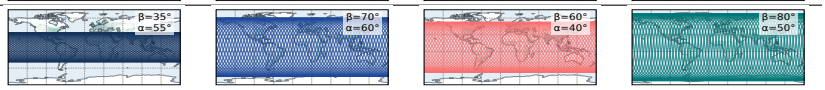
Orbital altitude H	Orbital period T	p	q	Ground track examples
423 km	92.8 min	2	31	
573 km	95.9 min	1	15	
1141 km	108 min	3	40	
1335 km	112.2 min	6	77	
1873 km	124.2 min	5	58	

Table 2: Some candidate Earth-repeat ground tracks