CosMAC: Constellation-Aware Medium Access and Scheduling for IoT Satellites

Jayanth Shenoy[†#], Om Chabra[§#], Tusher Chakraborty[‡], Suraj Jog[‡] Deepak Vasisht[†], Ranveer Chandra[‡]

[†]University of Illinois Urbana-Champaign, [§] Massachusetts Institute of Technology, [‡] Microsoft

ABSTRACT

Pico-satellite (picosat) constellations aim to become the de facto connectivity solution for Internet of Things (IoT) devices. These constellations rely on a large number of small picosats and offer global plug-and-play connectivity at low data rates, without the need for Earth-based gateways. As picosat constellations scale, they run into new bottlenecks due to their traditional medium access designs optimized for single (or few) satellite operations. We present CosMAC – a new constellation-scale medium access and scheduling system for picosat networks. CosMAC includes a new overlap-aware medium access approach for uplink from IoT to picosats and a new network layer that schedules downlink traffic from satellites. We empirically evaluate CosMAC using measurements from three picosats and large-scale trace-driven simulations for a 173 picosat network supporting 100k devices. Our results demonstrate that CosMAC can improve the overall network throughput by up to 6.5× over prior state-of-the-art satellite medium access schemes.

ACM Reference Format:

Jayanth Shenoy[†#], Om Chabra[§#], Tusher Chakraborty[‡], Suraj Jog[‡] Deepak Vasisht[†], Ranveer Chandra[‡]. 2024. CosMAC: Constellation-Aware Medium Access and Scheduling for IoT Satellites. In *The 30th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '24), November 18–22, 2024, Washington D.C., DC, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3636534.3690657

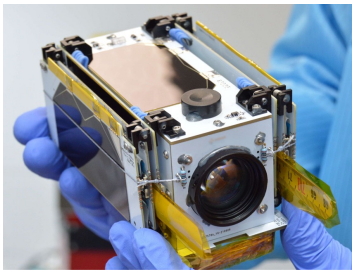
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM MobiCom '24, November 18–22, 2024, Washington D.C., DC, USA

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0489-5/24/11

https://doi.org/10.1145/3636534.3690657



Size: 10x5x5 cm^3
Altitude: ∼ 500 Km
Avg power: 1.67 W
Radio: SX1262 (LoRa)
Uplink: 401.3 MHz
Downlink: 401.1 MHz
Beacon: 401.7 MHz
Bandwidth: 125 kHz
Antenna: Omni

Figure 1: *One of the 3 picosats launched for experiments.*

1 INTRODUCTION

Constellations of pico-satellites (picosats) in low earth orbits (LEO) promise to enable universal plug-and-play connectivity for Internet-of-things (IoT) devices. Picosats (Fig. 1) are small in size and are built using off-the-shelf low-complexity hardware. Therefore, they are inexpensive to build and launch into orbit. With picosat constellations, users do not need the technical expertise to deploy network backhauls, power infrastructure, and gateways before they can enable IoT connectivity. IoT devices, compatible with picosats, can simply be turned on and connected to the Internet. Excited by this vision, more than a dozen companies have deployed constellations of hundreds of picosats and providing commercial IoT connectivity services today [2, 4–6, 10, 13].

Direct-to-satellite (DtS) is the prevalent connectivity model for IoT-picosat connectivity [1, 7, 34, 61, 62]. In DtS, IoT devices first receive a beacon from a picosat overhead and then directly transmit their data to picosats in orbit (Fig. 2a). After receiving the data, the picosat opportunistically forwards the data to ground stations on Earth. Lastly, the ground stations, which are connected to terrestrial networks such as the Internet, forward the received data to the cloud for aggregation.

IoT-picosat networks differ from terrestrial IoT networks in terms of power constraints, backhaul infrastructure, and mobility. In terrestrial IoT networks such as LoRaWAN, IoT devices communicate with stationary gateways that benefit from consistent power sources and backhaul. For picosat networks, IoT devices directly talk to picosats. Such picosats are powe-constrained because their small size (Fig. 1) prevents

# indicates primary student authors.

them from using large solar panels and large batteries. Besides, due to their orbital motion in low orbits, picosats move fast with respect to the Earth and cannot connect to a ground device for more than ten minutes at a stretch [62, 69]. Thus, picosats have intermittent backhaul connectivity to ground stations and a fast-changing picosat-IoT device relationship. In essence, unlike terrestrial networks which operate with constrained IoT devices and well-resourced gateways, picosat networks must deal with constraints at both the IoT devices and picosats (functioning as gateways).

Due to these constraints and the growing scale of picosat networks, IoT-picosat networks face the following key challenges (highlighted in Fig. 2b):

Challenge 1: Uplink Collisions at the Satellite: Uplink transmissions from IoT devices to picosats encounter a high incidence of collisions. This arises because picosats have large footprints – a satellite can receive transmissions from areas spanning a few million Km^2 containing thousands of devices. The large number of devices, combined with the limited spectrum, make it challenging for the satellite to explicitly coordinate medium access through time or frequency-domain multiplexing. These devices, due to their terrestrial distances and obstacles, lack the ability to overhear each other's transmissions and utilize carrier-sense-based medium access. Moreover, IoT devices employ omnidirectional antennas, making it impossible to direct their transmissions toward a specific satellite. Consequently, they rely on random-access protocols like Aloha. Their transmissions generate interference at multiple satellites, resulting in multiple collisions stemming from a single transmission. While there has been limited past work in optimizing random access for individual satellites [20, 70], it does not consider the impact of overlaps in the footprints of multiple satellites, as shown in Fig. 2b.

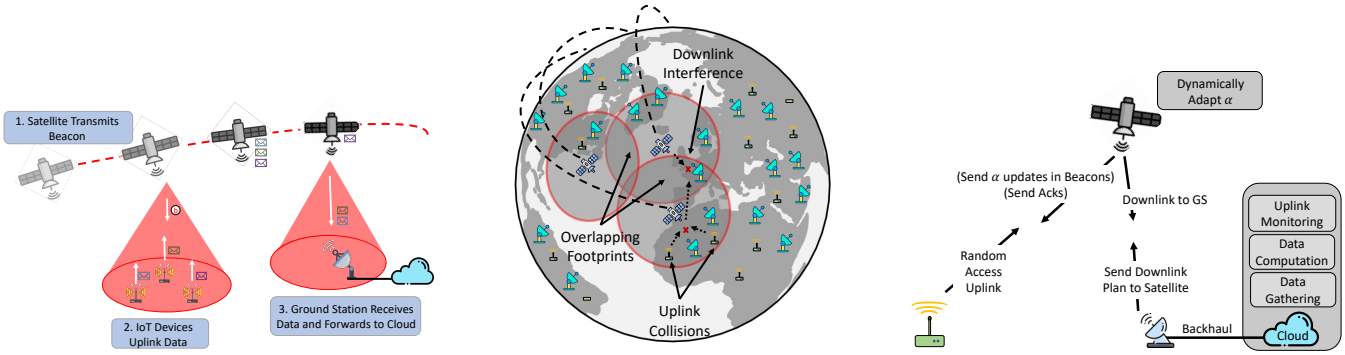
Challenge 2: Acknowledgments and Flow Control: Traditional networks like Wi-Fi and RFIDs rely on feedback (e.g., acknowledgments) from the gateway to control the sending rate. However, in IoT-picosat networks, IoT traffic is infrequent, typically consisting of only a few tens of packets per day. Consequently, the feedback obtained by a device from one transmission (through its corresponding acknowledgment) becomes outdated by the time of the next transmission. Moreover, with IoT devices not enduring contact with any given satellite for more than a few minutes, the subsequent transmissions happen to a different satellite. Given these constraints, we need to find new mechanisms for adapting the data transmission rate from IoT devices.

Challenge 3: Scheduling Ground Station Contacts: Traditional satellites use phased arrays or large dishes to beamform data directly to ground stations. When multiple satellites and ground stations are in sight of each other, a centralized scheduler identifies which satellites should talk to which ground stations and establish one-to-one links. Extensive research [18, 29, 69] exists in identifying the best set of links to activate at any time. However, picosats use omnidirectional antennas and cannot perform beamforming for one-to-one communication with ground stations. Moreover, picosats increasingly rely on distributed ground station designs like TinyGS [12] having 1000+ small ground stations deployed globally [62]. Therefore, picosats establish *one-to-many* links with dense ground station deployment. Specifically, when one satellite is allowed to transmit, its signal interferes at *multiple* ground stations that can no longer receive signals from other satellites. Existing scheduling algorithms do not account for such one-to-many interference patterns.

In this paper, we present CosMAC a new system that addresses the above challenges to improve scalability, robustness, and performance of picosat networks. CosMAC contains a new overlap-aware medium access scheme and a novel scheduling architecture. CosMAC does not require hardware changes at the satellites or ground stations and operates at the software/firmware level. While past work has optimized individual satellite or ground station design, CosMAC takes a unique *constellation-scale collaborative* approach that reveals new optimization opportunities. CosMAC's design consists of three key components:

(i) Mitigating Uplink Collisions: In IoT-picosat networks, we observe that the costs of a single transmission from different IoT devices are not equal. For example, if a device is in the overlapping footprint region of multiple satellites, its transmissions prevent these satellites from receiving transmissions from other IoT devices (collision) as illustrated by Fig. 2b. Based on this observation, we create a new overlap-aware random access protocol that penalizes such heavy hitters from an overlapping region. It contrasts to medium access approaches used in terrestrial networks, where such schemes would cause starvation to the devices in overlapping regions. However, in satellite networks, the orbital motion ensures that the satellite footprints as well as the overlapping regions move across the Earth. Our approach minimizes collisions and improves the overall uplink throughput.

(ii) Dynamic Flow Control: Due to the variable data generation rate of devices (e.g., event-driven traffic), we need to regularly adapt the transmission rate of IoT devices. To achieve this objective with minimal overhead, we design a collaborative flow control scheme which leverages beacons transmitted from off-the-shelf picosats. With CosMAC, picosats provide feedback to all devices under their footprints for adjusting their transmission rates. The picosat determines when devices need to scale back their transmissions by detecting and measuring collisions over time. If the number of collisions measured is significantly larger than the amount expected from periodic traffic, the picosat can



(a) *IoT-picosat Network DataFlow* **(b)** *IoT-picosat Network Challenges* **(c)** *CosMAC System and Data Flow*

Figure 2: *CosMAC System Outline: (a) Overview of Direct-to-satellite connectivity models, (b) Interferences due to overlapping footprints in uplink and downlink reduce network efficiency, and (c) System overview and data flow of CosMAC.*

include a 'backoff' signal in its periodic beacon message and vice versa.

(iii) One-to-many Scheduling: CosMAC designs a new scheduling algorithm for one-to-many satellite-ground station connections. Our algorithm strives to ensure that two satellites transmitting concurrently must have at least K non-overlapping (i.e., interference free) ground stations available to each satellite. By achieving the above goal, CosMAC is able to optimize the satellite downlink schedule for both network reliability and network robustness. We formulate CosMAC's schedule as a graph problem and demonstrate that our problem can be mapped to the maximum weighted independent set problem from traditional graph theory, which is known to be NP-Hard, making it challenging to solve. We use a randomized approximation algorithm to determine the final schedule for the satellites. Our formulation is specifically designed such that the scheduler prioritizes links with high channel capacity and achievable data rates, while also accounting for receiver diversity to ensure reliability.

To evaluate CosMAC, we devise a comprehensive research platform for IoT-picosat networks, encompassing both real-world picosat based IoT deployments and large-scale, reliable simulations. Our platform comprises three picosats in orbit, launched in collaboration with FOSSA Systems [5], a commercial IoT-picosat service provider. Specifically, one picosat was launched exclusively for this research endeavor, while the other two are utilized for extended evaluations. The setup further includes two ground stations and multiple IoT nodes. For large-scale experimentation, we develop new open-source trace-driven simulator, CosmicBeats, using data and models derived from our real-world setup. We have released this simulator and the models in the public domain [60]. We conduct simulations involving constellations comprising 173 satellites, 100,000 devices, and 1,048 ground stations. Our results show that CosMAC can improve net end-to-end throughput by 6.5× compared to state-of-the-art baselines.

Our contributions in CosMAC are:

- We describe a new constellation-aware random access uplink technique that improves net uplink throughput.
- We design a novel downlink scheduler for satellite-ground station links that account performance and reliability.
- We build a real-world research platform consisting of three picosats, two ground stations, and multiple IoT nodes.
- We leverage the measurements from our real-world setup to build the first data-driven open source simulator for IoT picosat networks, CosmicBeats [60], and perform a large scale evaluation of CosMAC.

2 BACKGROUND

Picosats represent an emerging category of low-cost and low-complexity LEO satellites. As of 2022, around 2000 picosats have been launched [8]. The defining characteristic of this burgeoning industry is its ability to rapidly expand network coverage at a reduced capital outlay, enabling end-users to access services at lower prices. This feat is accomplished by miniaturizing picosats (Fig. 1) and simplifying their hardware design. This stands in contrast to the satellite constellations for broadband Internet, bent-pipe transponders, and direct-to-cell connectivity (e.g., 4G/5G NTN), which entail substantial capital expenditure. These constellations employ larger satellites equipped with high-end hardware, often exceeding 10x the cost and size of picosats.

DtS IoT: In an IoT-picosat network, IoT devices on Earth directly communicate with picosats using the direct-to-satellite (DtS) model (Fig. 2a). These IoT devices share nearly identical characteristics with traditional terrestrial IoT devices in terms of hardware design, power profile, and cost. On the other end, 10s - 100s of picosats in a constellation serve as gateways, aggregating data from these IoT devices and downloading it to ground stations. Due to their simplistic

design, picosats employ very basic hardware configurations, including omnidirectional antennas without any beamforming capability, off-the-shelf radio and computation units, and low-capacity batteries (Fig. 1). The main advantage of the DtS model is its simplicity in usage and deployment. IoT device users can simply turn a device on and connect to the Internet from anywhere on Earth without requiring a terrestrial gateway. This connectivity proves particularly advantageous in remote areas lacking terrestrial infrastructure, such as farms, forests, and oceans. Moreover, in urban environments, DtS models enable setup-free deployment.

Data Communication: Despite being power-constrained, IoT devices achieve the necessary link budget for direct transmission to the satellite by employing low-power modulation techniques, such as LoRa [58, 61, 62], which is a popular choice among commercial IoT-picosat service providers like Wyld Networks [13], EchoStar [4], Lacuna [48], SWARM [10], and FOSSA [5]. Furthermore, IoT-picosat networks operate within the VHF to S bands (100 MHz - 2.4 GHz), where low-frequency data communication ensures that the wireless links between the IoT devices and picosats are not significantly affected by weather and atmospheric conditions, unlike high-frequency broadband satellites such as Starlink [9]. Low frequencies also experience lower path loss, reducing the power requirement on IoT devices and picosats.

An IoT-picosat network operates exclusively within a licensed spectrum or spectrum allocated for weather watch and meteorology in some countries. However, the allocated operational bandwidth is very limited, typically around one MHz, posing a significant challenge with spectrum availability [58, 59, 62]. The network operator divides the allocated spectrum into multiple channels, each with bandwidths ranging from 10s to 100s of kHz. The combination of low channel bandwidth and low-power modulation techniques like LoRa results in a low data rate satellite-ground communication link of around one kbps. Here, uplink (ground to satellite) and downlink (satellite to ground) operations occur in different channels. However, due to the scarcity of spectrum, network operators face limitations in allocating multiple channels in any direction. For instance, SWARM can hardly avail more than two uplink channels of 125 kHz (standard LoRa channel bandwidth), while FOSSA can only utilize one [66].

Ground Stations: Picosats maintain contact with two types of ground stations – Telemetry, Tracking, and Control (TT&C) and data downloads. Unlike traditional multi-million dollar ground stations, these are low-cost facilities equipped with off-the-shelf radios and rotation-capable Yagi antennas (Fig. 6) [62]. Due to the simplistic and low-power design choices, picosats communicate with the ground stations using low data rate modulation like LoRa. On average, a picosat passes over a ground station 2-3 times a day, with contact times of 6-8 minutes each. However, with such a low data rate and short contact time, a small number (6-8) of ground stations are insufficient to download data from a large constellation comprising 100s of satellites, which aggregate data from 10s of thousands of IoT devices [62, 69]. As a solution, a new concept of distributed ground stations has emerged in both academia and industry [16, 52, 63, 69]. For picosats, such distributed architectures are increasingly being adopted. For example, TinyGS [12] has deployed 1000+ ground stations for picosats, with support for multiple constellations. These ground stations are highly cost-effective (~ $100), utilizing LoRa radios with omnidirectional antennas and WiFi as the backhaul. They are receive-only (no uplink) and utilized for downloading sensor data and health status from picosats.

3 COSMAC OVERVIEW AND GOALS

Our goal is to identify bottlenecks in large-scale IoT-picosat constellations and design new primitives to eliminate those bottlenecks. Specifically, we aim to achieve the following:

- **Scale:** We aim to support a large number of satellites (100s of satellites) and IoT devices (100k devices).
- **Performance:** We aim to maximize the end-to-end throughput for IoT traffic.
- **Robustness:** We should be able to recover from transient failures/errors at satellites or ground stations.
- **Low-power:** We should incur little additional power overhead for IoT devices and picosats.

Fig. 2c shows an overview of CosMAC's design. As shown, CosMAC's uplink medium access algorithm (Sec. 4.2) runs on each IoT device. The rate of data upload, i.e., the flow rate, is mediated by our flow control algorithm (Sec. 4.3) running on picosats that communicate control information to IoT devices through existing periodic beacon transmissions. The centralized scheduler, operating on the cloud, determines the optimal schedule for downlink transmissions from picosats (Sec. 5). This schedule is then communicated to the picosats through TT&C-capable ground stations. Subsequently, the downlink transmissions adhere to this schedule and can be received at one or more ground stations. The received data from the ground stations is forwarded to the cloud.

4 UPLINK MEDIUM ACCESS & FLOW CONTROL

Designing medium access for IoT devices in picosat networks is uniquely challenging because of four distinct attributes: large footprint, high mobility, limited power and bandwidth, and independent operation of picosats (which act as gateways). The large footprint (million sq. miles) means that a single gateway may be exposed to thousands of IoT devices, without the possibility of carrier sensing. The high

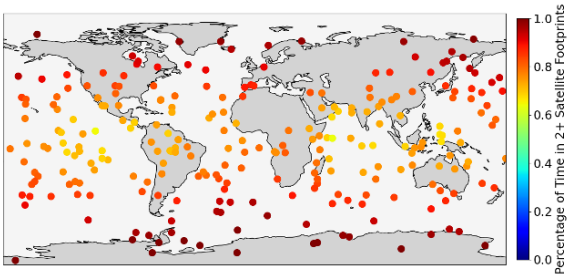
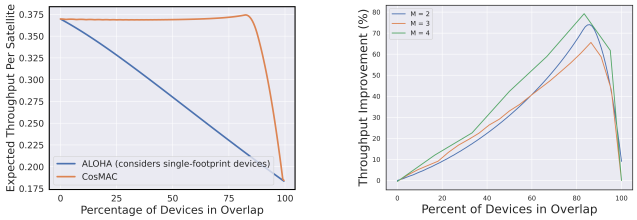


Figure 3: *IoT device spending a percentage of time under 2+ footprints for 173 satellites based on geographic location. IoT device spends 83% of time on average in an overlapping region.*



(a) *Two Satellite Analysis.* **(b)** *Scaling Analysis*

Figure 4: *Mathematical analysis: (a) Past work exhibits poor performance in managing overlapping satellite footprints, where CosMAC boosts performance (b) CosMACś effectiveness scales with the number of overlapping footprints.*

mobility of satellites eliminates static traffic mapping approaches at the gateway (e.g., time slot allocation). Picosats' limited power and bandwidth prevent the use of gateway-coordinated medium access. Finally, picosats operate without inter-satellite links and have scattered contacts with ground stations, making real-time coordination between satellites challenging. As a result, picosats must use randomized medium access protocols like Aloha, with the possibility of nodes operating across multiple frequency bands.

4.1 Observations and Design Choices

Past work [15, 22, 70] focuses on improving the contention in a single satellite-multiple device setting. In contrast, we take a constellation-scale approach for optimizing the uplink transmissions. Specifically, our key observation is that *some devices interfere at multiple satellites simultaneously, and therefore, waste more than one data slots for single unit of transmission.* If a device transmits when it is within the coverage area of multiple satellites, it contends for transmission slots on all those satellites with the same data.

We simulate a commercial picosat constellation – the SWARM constellation [10] with 173 satellites. We place IoT devices uniformly across the globe and observe the fraction of time each device spends in the footprint of two or more satellites, i.e., its transmissions will be heard by more than one satellite. We plot this distribution in Fig. 3. On average, an IoT device spends 83 % of its satellite coverage time in overlapping footprints. This fraction is higher at the poles because picosat orbits are (near) polar and visit the poles in each orbit, even though they scan different parts of the Earth during the orbit. At a given time, devices may simultaneously be in up to 33 (median: 5) footprints.

Our strategy is to penalize devices causing a collision at multiple satellites to improve overall network throughput – for example, a slot left free by a device in the footprint of five satellites can then be used by five independent devices. Such a penalty leads to unfairness in terrestrial networks because penalizing devices in overlapping footprints leads to permanent starvation for a subset of devices. However, in satellite networks, an IoT device sporadically enters and exits overlapping footprints due to the satellite's orbital motion and Earth's diurnal motion. Such motion naturally distributes the penalty to different devices over time.

At first glance, it may appear that such uplink collisions could be addressed using a different frequency channel for each neighboring satellite. However, as discussed in Sec. 2, picosat constellations have a scarce spectrum – they can only manage one or two frequency channels for uplink, despite the potential for overlapping footprints from tens of satellites. Availing a large chunk of spectrum at the global scale is capital-intensive leading to higher costs for end users, which contradicts the business moat of the IoT-picosat industry.

4.2 Overlap-aware Medium Access

In today's IoT-picosat network, when an IoT device has data to transmit, it searches for an overhead satellite by listening on the predefined satellite beacon channel (see Fig. 2a). Satellites transmit very short beacons at regular intervals similar to LoRaWAN Class B protocol [14, 62, 70]. After receiving the beacon, the device then transmits its data using a random access protocol, i.e., Aloha [57–59, 70], where each device d transmits data at time step t after the beacon with a transmission probability p_d. Previous satellite-specific research [70] proposed a modification over classic Aloha where p_d depends on the number of devices (N) in the footprint of a satellite, with a variable value of N over time. The value of N is estimated from the network deployment metadata of the IoT-picosat service provider, a practice commonly employed in the industry [70]. The satellite communicates this information to the devices using beacons, which update their transmission probability p_d.

However, these approaches only consider a single satellite, leading to sub-optimal performance when multiple satellite footprints overlap. In IoT-picosat constellations, such overlapping footprints are very prevalent, as discussed before. To demonstrate this limitation, we run a probabilistic analysis for the simplest scenario where the footprints of two satellites overlap. Here, we numerically calculate the expected

throughput for each satellite using the probability functions of the devices. Fig. 4a shows the expected throughput of Aloha-based approaches. When there is no device in the overlapping region, Aloha achieves its optimal throughput and slotted theoretical efficiency of nearly 37% [24]. As the number of devices in the overlapping region increases, the throughput drops to half (nearly 18% efficiency). The decline is due to the devices in the overlapping region creating collisions with all devices from both footprints. In practice, many more satellites can overlap causing steeper efficiency drops.

To address these limitations, CosMAC proposes an overlap-aware randomized medium access with real-time dynamic flow control. CosMAC establishes overlap awareness at the IoT device level by leveraging existing beacons from picosats. A device in the overlapping footprint of multiple satellites receives multiple beacons within the standard beacon interval T_b. If N_{Sat_i} is the number of estimated devices in the footprint of satellite, Sat_i, then a device in footprints of k satellites potentially interferes with $\sum_{i=1}^{K} |N_{Sat_i}|$ devices. Then,:

$$p_d = \frac{\alpha}{\sum_{i=1}^{K} |N_{Sat_i}|} \quad (1)$$

The denominator of Eq. 1 naturally penalizes heavy hitters because they interfere at multiple satellites, and hence cause contention in multiple slots. We incorporate a transmission hyper-parameter α for dynamic traffic flow control as described in Sec. 4.3.

CosMAC's re-definition of p_d is significant. From our probabilistic analysis of the two-footprint overlapping scenario in Fig. 4a, it's evident that CosMAC exhibits the potential to maintain the optimal throughput even with an increasing number of devices in the overlapping region. This is achieved through accounting for the heavy hitters from the overlapping region and dynamically tuning α to maximize the average expected throughput. CosMAC offers on average 37% and up to 75% performance improvement over existing approaches in the simplest two-footprint overlapping scenario. However, we observe tens of overlapping footprints in real-world settings. To show CosMAC's performance with an increase in the number of overlapping satellite footprints (M), we run probabilistic analysis up to $M = 4$. Note that in the analysis of scenarios with $M > 2$ footprints, we distribute the devices across all possible overlapping regions consisting of 2 to M footprints. Fig. 4b shows that the average performance improvement of CosMAC over existing approaches persists with the number of overlapping footprints.

4.3 Flow Control

CosMAC uses the parameter α defined above to account for dynamism in IoT traffic and enable flow control, i.e., controlling data transmissions depending on traffic load and orbital parameters. Such flow control has not been enabled by past work and is unique to CosMAC.

Specifically, an IoT network carries two types of traffic: a) periodic, where sensors report data at predefined intervals [68], and b) event-driven, where data reporting is triggered by specific events like forest fires, rain monitoring, or asset tracking. While the former could be predicted in advance, the latter is highly unpredictable and often takes precedence [19]. Consequently, flow control must be dynamic and operate in real time. In traditional networks, real-time flow control, such as backoff when the transmission rate is too high, relies on feedback from the receiver, typically through ACKs. However, in IoT networks, devices only communicate with the gateway (picosat here) when they have data to transmit. Typically, an IoT device sends at most a few tens of packets sporadically throughout the day. Since the device-to-picosat contact lasts for less than 10 minutes, these packets are highly likely to be received by different picosats. As a result, any flow control feedback received during a transmission becomes obsolete for the next transmission. Hence, an IoT device must refresh its flow control parameters before initiating any new transmission.

CosMAC adjusts the hyper-parameter α in Eq. 1 to facilitate flow control. Instead of relying on per-device feedback, a satellite incorporates aggregate indications regarding network capacity utilization within its beacons. When the satellite detects that the network is under-utilized (over-utilized), it includes a *binary* signal in its beacons to increase (decrease) the value of α (Fig. 2c). This approach enables us to furnish aggregate flow control feedback to the devices before attempting a transmission, all without incurring significant overhead. Our approach operates in a decentralized manner, with each device locally tracking the evolving values of α. The dynamic tuning of α also plays a crucial role in ensuring fairness within the network across various geographical locations. For instance, as depicted in Fig. 3, devices located toward the poles are situated in significantly higher overlapping regions. This overlap introduces a challenge, as our penalizing scheme risks causing devices in these areas to starve. This is because the denominator's sum of Eq. 1 overestimates the number of devices contending for resources. In response, CosMAC dynamically adjusts α to guarantee fairness for devices spending more time in densely overlapping areas (see Fig. 11a).

Traffic sensing: To enable flow control, a satellite needs to identify when to send increase/decrease signals, i.e., (a) when is the channel too idle (inactivity)? and (b) when is the channel oversubscribed (too many collisions)? Collision detection and carrier sensing are well-studied in the wireless domain. Collision detection based on carrier energy density in LoRa is challenging because LoRa devices can operate below the

noise floor [19, 38]. How does one detect transmissions and collisions below the noise floor? CosMAC addresses this challenge by taking advantage of the Channel Activity Detection (CAD) technique offered by off-the-shelf LoRa gateways, including the ones carried by picosats [11]. CAD detects the presence of LoRa symbols on a channel with minimal power consumption [36], even when the complete LoRa packet cannot be decoded. CosMAC runs very short CAD (e.g., of four LoRa symbols length) at a regular interval. The interval is set such that multiple CADs can be conducted even within the shortest expected LoRa packet time. By observing positive CAD results but a lack of valid packet reception, CosMAC can estimate collision events. CosMAC uses CAD to observe unusually long channel inactivity as well. Note that collision estimation using CAD considers the presence of LoRa symbols, not a packet's validity. Although this leads to a partial overestimation of collisions, it is factored into the threshold hyperparameters described below. Additionally, while CosMAC is unable to precisely pinpoint the collision and determine the number of packets colliding, the information obtained through this process is sufficient for our traffic control mechanism to operate optimally (see Sec. 7.1).

Traffic control decision: At a high level, our flow control decisions are two-fold: if the fraction of colliding packets is too high, then reduce α. If the channel is idle for a significant fraction of time, increase α. However, we need to precisely define the increase and decrease criteria.

First, we compare the observed collisions to the expected number of collisions. Given M IoT devices within the satellite footprint. The expected number of collisions depends on: 1) the amount of data that these devices have generated and 2) the number of devices transmitting data. For every device d, we model the data generation of each device $D \sim poisson(\lambda)$ where λ is the data generation rate of typical traffic on the network. The estimated probability of a device transmission is simply the product of the probability of a device having data and its fair-share probability ($p' = \frac{1}{M}$): $P_{dTx} = P(D \geq 1) * p'$. Note that, it would be more accurate to use p_d instead of p', but the satellite doesn't have access to per-device p_d.

Then, the binomial $P_{Tx} = Bin(P_{dTx}, M)$ approximates the expected number of transmissions in a given interval of time. The probability of expected collision, P_{EC} is:

$$P_{EC} = 1 - P_{Tx}(Tx = 1) - P_{Tx}(Tx = 0) \quad (2)$$

Similarly, the channel inactivity threshold, τ_{EI}, is based on the probability of unoccupied uplink slots on a satellite ($P_{Tx}(Tx = 0)$). If observed inactivity is more than τ_{EI}, we need to decrease α. It is important to note that the optimal values for λ, P_{EC}, and τ_{EI} may change depending on the geographical region and time. While the real-time adaptation to such variation is implemented by satellites, the satellite network performance could also be monitored in the cloud (Fig. 2c). Lastly, we find that simply relying on these thresholds for increasing and decreasing α proves to be too reactive. As a result, we also consider temporal traffic trends over one or multiple beacon intervals as defined below.

Case I: An increasing or flat collision trend beyond P_{EC} is a sign of the satellite entering a high-traffic region or experiencing a new burst of traffic. The satellite decides to send a "slow-down" response in its beacon.

Case II: A flat or declining trend below τ_{EI} may indicate two scenarios: the devices may have no data to transmit or they may be restricted by a previous "slow-down" decision from an earlier satellite. The satellite will instruct the devices to transmit more aggressively ("speed up").

Case III: In the event of a declining trend in collisions, regardless of whether it is above or below P_{EC}, we do not enforce traffic control. This could indicate that the satellite is moving away from a heavy traffic congested area or that a burst of traffic has subsided.

Drawing from the above analysis, we highlight a key feature of our *constellation-scale collaborative* design principle: any short-term unfavorable decisions made by a satellite for a region are quickly rectified by subsequent satellites.

Traffic control execution: In response to the "speed-up" or "slow-down" messages in satellite beacons, individual devices tune α in Eq. 1 – increasing α results in more aggressive transmission and decreasing it has the opposite effect. To determine the appropriate value of α, we draw inspiration from prior work on congestion control [43]. We use a multiplicative-decrease additive-increase strategy to ensure fairness among devices, with the multiplicative factor and additive increase value chosen empirically.

5 CENTRALIZED DOWNLINK

Downlink satellite scheduling is a traditionally well-studied problem with rich literature [18, 29, 35, 64, 69]. However, past work considered bulky and sophisticated satellites that had the ability to beamform to individual ground stations, thereby eliminating interference at neighboring ground stations. In contrast, picosats use omnidirectional antennas that cast a very wide footprint on the earth's surface as shown previously in Fig. 2b. This means that transmissions from a single picosat can be received by multiple ground stations in its footprint, and can potentially create interference at these ground stations, precluding them from receiving transmissions from any other nearby satellites. In our experiments with SWARM constellation and TinyGS ground stations [12], a satellite can have up to 300 different ground stations in its footprint and can cause massive interference.

In addition, picosats are extremely power-constrained. Transient packet losses caused by phenomena such as microclimate variations or terrestrial interference will require retransmissions (high power cost). To achieve reception reliability and network robustness, we aim to ensure that at least $K \geq 2$ non-overlapping (i.e. interference-free) ground stations are made available to each satellite. CosMAC should choose these ground stations, such that packet losses from the satellite to the chosen ground stations are uncorrelated. Hence, CosMAC's downlink scheduler must:

(1) *Optimizes the downlink throughput by maximizing the number of satellites that are downlinking simultaneously such that none of the transmissions create interference with each other.*

(2) *Generate a schedule that is robust to transient packet losses, by assigning $K \geq 2$ ground stations with uncorrelated packet losses for each satellite.*

5.1 Downlink Scheduling Overview

We adopt a centralized scheduling approach, similar to past work [18, 29, 35, 64, 69]. This is possible because the orbital motions and locations of the satellites, including their contact periods with ground stations, are all highly predictable using satellite orbital TLEs. Additionally, the link capacity and achievable data rates from satellites to different ground stations can also be estimated using standard ITU link quality models [39–41]. A centralized scheduler can also take into account interference patterns between different satellite links and how they evolve with time, and use this predicted network information to compute future schedules that are simply executed by each satellite independently without having to worry about carrier sense or collision avoidance.

We formulate the downlink scheduling problem as an iterative graph problem. We divide time into time slots of 1 minute each, and in each time slot, the central scheduler solves a graph problem on a graph constructed based on the current downlink network configuration. From the solution to the graph problem, the central scheduler chooses a subset of satellites to transmit during that time slot. Our graph formulation can also be naturally extended to support downlink networking for constellations operating with multiple different frequency channels.

5.2 Graph Formulation

We start by describing the construction of the graph, $G_t(V, E)$ which represents the physical configuration of the network at time t, and captures link qualities and interference patterns in the network. Suppose we have N satellites represented as $\{s_1, s_2, \ldots, s_N\}$, and M ground stations $\{g_1, g_2, \ldots, g_M\}$.

Vertices: Fig. 5(a) shows the physical configuration of a downlink network at time instance t, and Fig. 5(b) shows the corresponding graph formulation for the network at t. As shown, the vertices $v_i \in V$ in the graph correspond to feasible links (s_i, g_j) that can be used for downlink communication. Feasible links (s_i, g_j) are communication links that provide data rates greater than some threshold cap_{thr}. Every satellite-ground station link from the different constellations will have its own corresponding vertex.

Edges: Given the graph with vertices V, scheduling is equivalent to choosing vertices (satellite-ground links). However, CosMAC's scheduler has to account for increased interference and resource conflicts, and we represent these conflicts in the network through edges in our graph. An edge between two vertices, $e = (v_x, v_y)$, implies that the links corresponding to vertices v_x and v_y cannot be scheduled simultaneously. CosMAC accounts for three conflict types:

(1) *Ground Station Conflict:* A ground station g_j, can receive from only 1 satellite at any given time. Hence, there will be an edge between all vertices that share the same ground station. That is, all vertex pairs (v_x, v_y) where $v_x = (s_i, g_j)$ and $v_y = (s_k, g_j)$ for all $j \in \{1, 2, \ldots M\}$, will have an edge between them. This conflict remains irrespective of whether s_i and s_k belong to the same or different constellations. We see an example of such conflicts in Fig. 5(b), where there is an edge between vertices (c) and (d).

(2) *Interference from neighboring links:* Our formulation must also consider the increased interference caused by the wide footprint of omnidirectional picosats. Consider the scenario in Fig. 5(a), where GS3 falls under the overlapped footprint of both Sat 1 and Sat 2. Let us say, we begin by scheduling Sat 2 along link (e) to transmit to GS4 which is interference-free. However, because Sat 2 transmissions are omnidirectional, the signal will also reach GS3, and in turn not allow GS3 to receive from any other satellite in its vicinity (Sat 1 in this example). Note that in this configuration, you could still schedule Sat 1 and Sat 2 to transmit simultaneously to GS1/GS2 and GS4 respectively, and consequently there are no edges between vertices (a) and (e) or (b) and (e).

Therefore, if a ground station g_j falls under the overlapped footprint of satellites s_i and s_k, then to schedule a successful transmission to g_j from s_i/s_k, we need to stop all transmissions from the other satellite s_k/s_i, even if the other satellite is attempting to transmit to a different ground station. Note that this conflict is relevant only to satellites belonging to the same constellation since satellites in different constellations operate at different frequencies. Concretely, for interference-free transmissions, if g_j falls under the overlapped footprint of satellites s_i and s_k, then the following edges are added to the graph — (i) edges between all vertex pairs (v_x, v_y) where $v_x = (s_i, g_j)$ and $v_y = (s_k, g_l)$, $\forall\, l \in \{1, \ldots M\}$, and (ii) edges

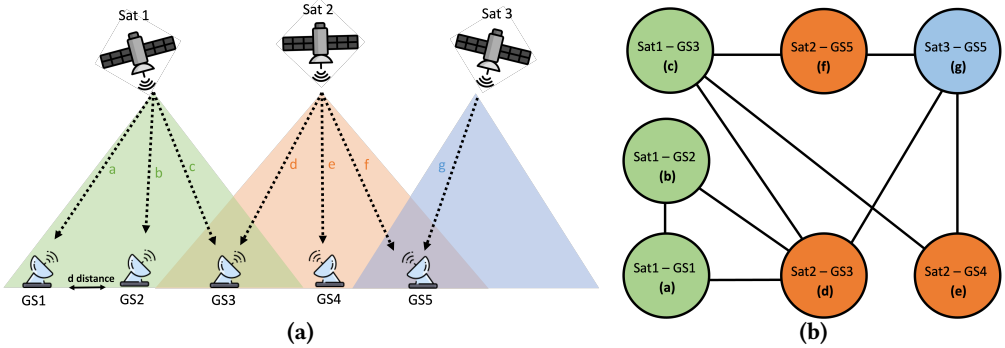


Figure 5: *Centralized Downlink Scheduling: (a) Physical configuration of picosats and available ground stations, (b) Graph formulation that accounts for interference.*

Figure 6: *Our ground station deployment.*

between all vertex pairs (v_x, v_y) where $v_x = (s_i, g_l)$, $\forall\, l \in \{1, \ldots M\}$ and $v_y = (s_k, g_j)$.

(3) *Receiver Diversity for Reliability:* CosMAC's formulation does not add edges between vertices corresponding to the same satellite. This is because we want the algorithm to schedule multiple ground stations receiving from a single satellite ensuring reliability. For reliability, we want different ground stations to have sufficient receiver diversity such that the packet losses across the chosen ground stations are uncorrelated. Packet losses are caused by transient factors like microclimate variations or localized terrestrial interference which affects some ground stations based on geographic locations. However, if CosMAC's algorithm chooses multiple closely located ground stations for a satellite, then these transient packet losses are likely to occur simultaneously across all chosen ground stations. To address this, CosMAC adds edges between vertices corresponding to the same satellite, if the two ground stations are closer than a minimum distance d_{min}, i.e., an edge exists between vertices (v_x, v_y) where $v_x = (s_i, g_j)$ and $v_y = (s_i, g_k)$ if distance $|g_j, g_k| \leq d_{min}$. Such a case is shown in Fig. 5, where GS1 and GS2 being close leads to an edge between vertices (a) and (b).

5.3 CosMAC's Scheduling Algorithm

Given graph $G_t(V, E)$, CosMAC's goal is to maximize the number of satellites that are downlinking data at time t, while ensuring that none of the satellite transmissions interfere with each other. This problem can be mapped to a Maximum Independent Set problem for graphs, where the goal is to pick the maximum number of vertices from the graph (analogous to scheduling the maximum number of satellite-ground station links) such that no two of the chosen vertices have edges between them (analogous to saying that no two scheduled links conflict with each other). However, we need to adapt this formulation for CosMAC in two key ways.

Link Capacities: Different satellite-ground station links have different capacities and achievable data rates. The satellite's orbital positions and the link capacities at those positions can be predicted well in advance [69], allowing us to plan our downlink schedule ahead of time. CosMAC's algorithm should prioritize scheduling links with higher channel capacities, but the current formulation of the graph does not account for this. As a result, we modify the graph by assigning *weights* $w_i(t) = r_i(t)$ to each vertex $v_i \in V$, where $r_i(t)$ is the RSSI for link corresponding to v_i at time t. With this definition of weights, the Maximum *Weighted* Independent Set problem (MWIS) algorithm will prioritize links with higher achievable data rates while maximizing the number of simultaneous active links. However, the MWIS problem is known to be NP-hard, and we use the randomized approximation algorithm presented in [30, 46]. Using this approximation algorithm, we can efficiently calculate CosMAC's optimal downlink schedule with an efficient polynomial time complexity.

Note that, unlike past work [46], our formulation does not suffer from a lack of fairness. That is, our algorithm does not end up scheduling the same set of high RSSI links repeatedly while starving other links. This is because at each time step t, our graph keeps updating the achievable data rates for each link in the network. Given the orbital dynamics of satellites, our formulation naturally ensures that no satellite is consistently starved since it will eventually move to the zenith of a ground station, and in turn, provide the highest data rate link at that time step.

Reliability Guarantees: For reliability, CosMAC tries to schedule $K \geq 2$ ground stations to receive a single satellite transmission. However, the algorithm works in a best-effort fashion and there will be instances where it can only schedule a single receive ground station for a satellite. To address these cases, we leverage a greedy heuristic that tweaks the schedule computed by the MWIS algorithm to meet our reliability goal. Specifically, for satellites in the schedule that are assigned only one receive ground station, the greedy

heuristic looks for opportunities where it can remove certain scheduled links and in its place, add additional ground stations for the satellites that had only one assigned station in the original schedule. It is possible that even through this heuristic, CosMAC is unable to meet the reliability guarantee of $K \geq 2$ ground stations for every satellite transmission. In these cases, CosMAC simply drops satellites from the schedule which do not meet the reliability criterion. While this design decision would admittedly hurt the total downlink throughput, it is a reasonable tradeoff considering that we want to avoid expensive retransmissions from picosats given their extremely limited power budget.

6 RESEARCH PLATFORM

We evaluate CosMAC using a combination of real-world and simulated experiments as outlined below.

6.1 Real-World IoT Satellite Testbed

We collaborate with FOSSA Systems [5], a commercial IoT-picosat service provider and launch our own picosat (details in Fig. 1) as part of their constellation of 12 satellites. The setup further includes two ground stations and multiple IoT nodes. Our picosat is designed and configured to be consistent with the commercial picosats, ensuring a seamless integration of our innovations into contemporary picosat networks. While we launch only one satellite, we utilize two other satellites in the constellation. We use two bidirectional communication-capable ground stations located in Spain (shown in Fig. 6) with satellite tracking capability. The testbed also includes IoT devices that use SX162 LoRa radios with omnidirectional antennas to communicate directly with our picosat.

6.2 CosmicBeats Simulator

As IoT-satellite constellations continue to expand, the research community is confronted with a pressing question: how should we design and assess new solutions for satellite-driven IoT networks? Although satellite simulators [26, 47, 49] have been developed previously, mainly for communication and earth observation satellites, none adequately meet the requirements for IoT-satellite communications. Specifically, these simulators fail to account for one-to-many transmission patterns, picosat-specific variations that can significantly affect link quality and power availability, and are typically designed for a limited number of satellites and ground stations (in the hundreds), which falls short of the large scale of IoT devices.

To address this issue, we introduce CosmicBeats[2], a scalable, data-driven simulator specifically designed for IoT-satellite networks. CosmicBeats is a packet-level simulator

[2]https://github.com/microsoft/CosmicBeats-Simulator

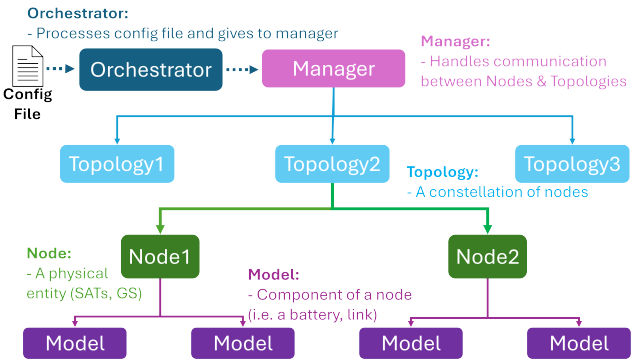


Figure 7: *System Design of CosmicBeats. This modular design allows for versatile simulations.*

that integrates satellite orbital models, field-of-view calculations, and MAC/network layer protocols. It incorporates power and communication models based on real-world data from our testbed and allows users to customize power and communication models according to their choice of solar panels, batteries, modulation, bandwidth, frequency bands, and more. Additionally, CosmicBeats is adaptable, capable of simulating not only IoT satellites but also earth observation and networked satellites.

6.2.1 Design Implementation. To meet these objectives, CosmicBeats uses a discrete-time method, performing simulated operations at regular intervals referred to as "epochs." This approach ensures uniform and predictable execution. In this section, we outline the specific structure of our system as depicted in Figure 7.

Nodes/Topologies: We begin by defining "nodes" as the fundamental computational unit in our simulator. A node represents a physical endpoint, which can include ground stations, user terminals, IoT devices, among others. A "topology" is a collection of nodes, often representing entire constellations. For example, Swarm's topology would include all Swarm satellites, user terminals, and ground stations.

Models: A "model" refers to a distinct component within a node, representing both hardware and software elements. Examples include satellite batteries, orbital models, computational capabilities, solar panels, etc. The interactions between models are crucial for accurate real-world simulations. For example, a computational model might need to check the power availability from the battery model before proceeding with its operations. To enable these interactions, a model provides public APIs that can be accessed by other models.

Manager: The simulation manager is responsible for controlling and running the simulation. During each epoch, the manager calls the *Execute()* method of each node, which in

turn calls the *Execute()* method of each model (the sequence is determined by the configuration file). The manager also includes runtime APIs accessible to external users. These APIs allow functionalities such as stopping/resuming the simulation, checking the state of a node, etc.

Post-Processing: After the simulation, CosmicBeats includes a post-processing pipeline consisting of *Single Model Analyzers (SMAs)* and *Summarizers*. An *SMA* processes the logs of a single model. For example, a data generation model might have an *SMA* that produces a table showing the times when data was created and the content of the data. Similarly, a data reception model could have an *SMA* that shows when data was received and its contents. A *Summarizer* can then combine multiple *SMAs* to calculate specific metrics. For instance, a latency *Summarizer* might use both *SMAs* to calculate the average latency between data generation and successful reception.

6.2.2 Real-world Modeling. CosmicBeats incorporates realistic models that have been confirmed through measurements from data collected from our real-world satellite testbed.

Orbital Dynamics and RF Links: We calculate the orbital movements of satellites using publicly available TLEs from CelesTrak [3]. We use real-world measurements from our satellite and ground devices to create a wireless link model that computes satellite-ground device link quality and data rate. The model takes into account the transmission's frequency, satellite power, distance, elevation, and weather. We adhered to state-of-the-art practices for characterizing the PHY, including power consumption and estimating the bit-error rate from SNR based on real-world traces. These measurements are based on the SX126x radio. Additionally, CosmicBeats models on-board packet queuing and considers the delay from uplink data reception to downloading the same to a ground station.

Power Modeling: We build a satellite power model based on measurements from our picosat in-orbit and coupled with its power allocation algorithm. Our model considers the satellite's power characteristics including generation (solar panels), storage (batteries), and consumption (operation/communications). Power allocation prioritizes critical tasks (e.g., battery heating, flight control) over communication services. We also model power generation based on sunlight exposure in orbit.

6.3 CosMAC Trace-Driven Evaluation

To evaluate CosMAC we use CosmicBeats with the following simulation parameters:

Satellites: Our simulation contains 173 satellites based on the TLEs of the largest IoT satellite constellation, SWARM [10]. The frequency and RF configs used for all the satellites are identical to our launched picosat (Fig. 1).

IoT Devices: We simulate 100,000 IoT devices and randomly but uniformly spread them out across the world. We simulate different traffic patterns by varying the data generation rate of every IoT device. We simulate 8 hours of the network at a 1-second time granularity.

Ground Stations: We model our ground stations using measurements from our real-world ground station deployment. We evaluate our system at scale by simulating the distributed ground station at locations of the TinyGS ground stations [12]. This network has 1000+ ground stations and can support multiple constellations.

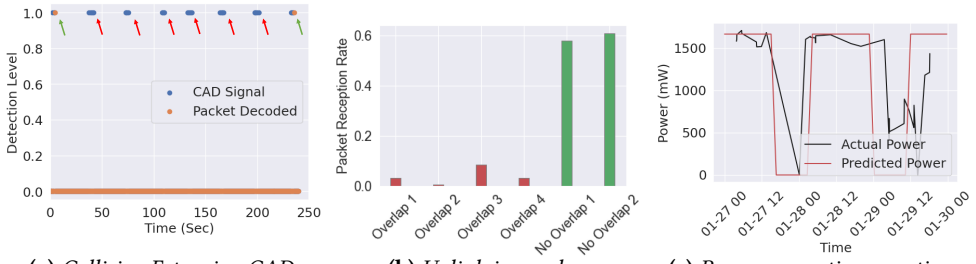
7 REAL-WORLD EXPERIMENTS

7.1 Collision Estimation

We tested CosMAC's ability to estimate collisions in LoRa using the Channel Activity Detection (CAD) mechanism. In CosMAC, this happens on the satellites, but since we cannot modify the satellites post launch, we used another satellite radio on the ground, with transmissions happening at the satellite (akin to IoT devices). Two satellites with overlapping footprints transmitted LoRa packets of 3.5 sec length at a 30 sec interval on the same channel. A LoRa gateway customized for 401 MHz band with an omni-directional antenna was set up on the ground. The gateway was tuned to the satellite transmission channel, and the SX126x radio of the gateway was programmed to run CAD every 0.5 sec. We plotted both the packet received and CAD values from one satellite pass in Fig. 8a. The figure shows that we got positive CAD values for two successfully decoded packets when the gateway was in the footprint of exclusively one satellite. However, when the gateway was in the overlapping footprints of two satellites, no packet was received due to collision. Nevertheless, we found the CAD value to be positive in those cases as well. This experiment validates that we can use CAD for collision estimation in uplink flow control.

7.2 Uplink Collisions at Multiple Satellites

A key claim in CosMAC is that the transmission, made by a device from the overlapping footprints of multiple satellites, will cause collisions at multiple satellites. We validate this claim through a real-world experiment. We set up two IoT devices transmitting uplink to our satellite. We specifically target passes where two of our satellites create an overlap over our IoT devices. During these passes, each device synchronously transmits to both satellites. In Fig. 8b, we plot the packet reception rate on both satellites across four passes when devices are transmitting from the overlap. We compare these passes with 2 control passes where a single IoT device is



(a) *Collision Est. using CAD.* **(b)** *Uplink in overlap.* **(c)** *Power generation over time.*

Figure 8: *Real-World Experiments: (a) Long distance collision detection, (b) Devices from overlapping region hits multiple satellites, (c) Power modeling*

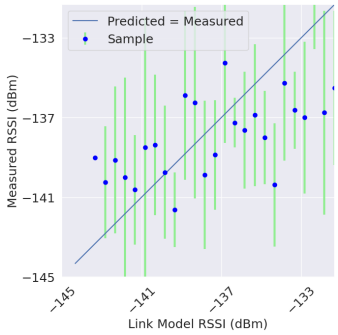


Figure 9: *Predicted vs Measured Link Quality.*

transmitting to a single satellite. When devices are transmitting from the overlapping region, the packet reception rate on both the satellites is significantly lower due to collisions. It establishes that a device in an overlapping footprint causes interference to all satellites creating that overlap. Since LoRa can decode packets despite collisions, some of the packets in our overlap experiment could be decoded.

7.3 Power Model

We compare our designed power model against real-world data from our launched satellite. Fig. 8c reports the power generation over 3 days. The figure illustrates that power generation predicted by our model largely matches that of real-world data. However, in the figure, there is a slight time offset between the actual and predicted power generation. This phenomenon is due to lag between the time the satellite generated power and the time that the power measurement log was received at the ground station. The reliability of our simulator's power model is important because it helps us determine realistically how well CosMAC can perform under the power constraint of picosats.

7.4 Satellite-Ground Device Link Model

We measure the signal strength of the signal received at our ground station from our satellite, over time. Our goal is to validate the parameterized link quality model used in CosmicBeats. Fig. 9 reports the corresponding samples of measured and predicted link qualities for 2 days worth of satellite passes. The plotted samples take the median values of samples for every .5 dbm interval on the x-axis. For each median sample, we plot the inter-quartile ratio as an error bar. The plot demonstrates that our link quality model is largely consistent with real-world link quality measurements.

8 LARGE-SCALE EVALUATION

After validating our key claims and ensuring the fidelity of our simulator through real-world experiments, we present large-scale results using trace-driven simulations.

Baselines: We use two baselines for *uplink* medium access – **(a) Uplink Transmission Probability Function (UTPF):** Prior work [70] explores modifying the transmission probability function (TPF) of devices based on the number of devices in a single satellite footprint. This is a common variant of the Aloha random access protocol for IoT-picosat networks. **(b) Fixed Probability Aloha (FP-Aloha):** Based on CosMACś key observation from Sec. 4.1, we initially attempt to enhance Aloha in a straightforward manner by calculating an optimal transmission probability across the constellation. We implement a fixed probability Aloha approach, wherein we calculated the Earth to have approximately 25 non-overlapping zones and use $p = \frac{25}{100{,}000}$ as each device's slot transmission probability in Aloha.

For downlink, our baseline is **L2D2** [69] – a state-of-the-art downlink schedule designed for broadband satellites. Although this schedule finds a maximal matching, it does not account for one-to-many interference caused by satellites with omnidirectional antennas.

Simulation Setup: For our simulation, we initialize multiple hyperparameters. We set all devices with an initial overly aggressive α as our flow control algorithm corrects for aggression quicker than under-utilization (see Sec. 4.3). Likewise, we set d_{min}, the distance between when two ground stations are considered to have correlated drops (see Sec. 5.2), to 1 km as the optimal tradeoff between robustness and throughput.

8.1 Throughput

CosMAC's primary goal is optimizing network throughput. We measure throughput through the simulation in our 173 satellite-100k device network and plot results in Fig. 10.

Uplink Throughput: Fig. 10a reports throughput in bps across varying data generation patterns for each uplink medium access scheme. Initially, at a low data generation rate of 5 packets per day (100 bytes/packet) the network is not fully utilized, and the UTPF scheme, which is aggressive, achieves relatively high throughput. However, when

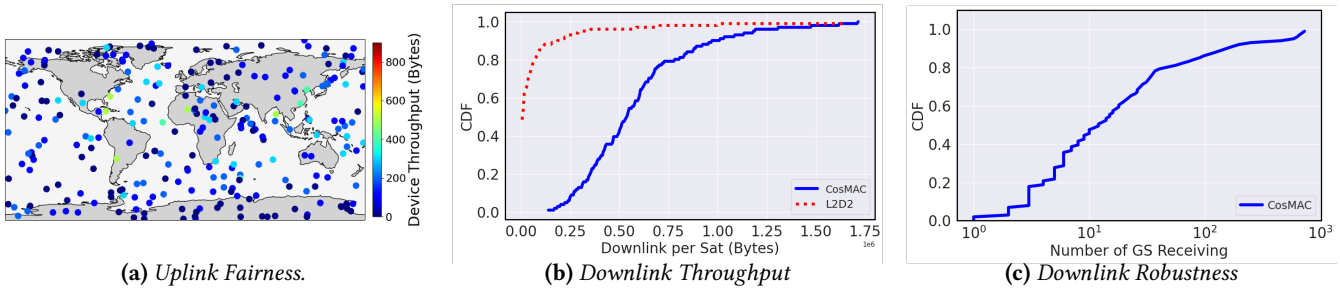
Algorithm	5 Packets	25 Packets	50 Packets	100 Packets
FP-Aloha (CosMAC inspired)	2631	2477	2381	2389
UTPF	2872	295	273	277
CosMAC	**3580**	**3495**	**3522**	**3467**

(a) *Uplink throughput (aggregate bps)*

Algorithm	5 Packets	25 Packets	50 Packets	100 Packets
FP-Aloha+L2D2	638	912	922	918
UTPF+L2D2	682	227	211	214
CosMAC	**1073**	**1378**	**1388**	**1383**

(b) *End-to-end throughput (aggregate bps)*

Figure 10: *CosMAC's system performance (a) Comparison of CosMAC's uplink throughput to baseline random access schemes (b) Comparison of CosMAC's end-to-end throughput to baselines*



(a) *Uplink Fairness.* **(b)** *Downlink Throughput* **(c)** *Downlink Robustness*

Figure 11: *(a) CosMAC achieves fairness on the uplink; (b) CosMAC outperforms L2D2 in downlink throughput; (c) CosMAC enables non-interfering one-to-many transmissions for robustness.*

the network transitions to the basic daily per-device quota offered by IoT-picosat companies, which is around 50 packets/day (5000 bytes) [23], the performance of the state-of-the-art baseline rapidly deteriorates. It shows that UTPF is simply not good enough for handling traffic at basic scale. In contrast, FP-Aloha, improved using CosMAC's simplified concept, begins to perform well with higher traffic. However, CosMAC outperforms others because of its ability to both a) minimize collisions from high interference-creating devices and b) dynamically apply network flow control.

Downlink Throughput: Fig. 11b plots the per-satellite downlink throughput of CosMAC's scheduler and the corresponding baselines. In this experiment, we generate infinite data directly on the satellite to understand how our downlink scheduler performs independently of uplink. The mean (90th percentile) throughput for CosMAC is 538,000 (969,800) bytes while the mean (90th percentile) throughput for L2D2 is 7,000 (181,600) bytes. CosMAC's one-to-many downlink scheduling achieves greater throughput than broadband satellite scheduling like L2D2 as CosMAC accounts for interference that stems from picosat's omnidirectional transmission in a dense ground station deployment [12].

End-to-End Throughput: Fig. 10b reports the end-to-end network throughput in bps across the varying traffic generation patterns applying both CosMAC's both uplink and downlink solutions. CosMAC outperforms medium access and scheduling settings composed of competing baseline uplink and downlink approaches. The trends we observe in the end-to-end throughput results are similar to those we observe in the uplink-only results from Fig. 10a. Among all approaches, CosMAC maintains consistently high performance across a variety of traffic generation schemes. As a result, CosMAC maximizes data from IoT devices to the cloud by providing a scalable solution that tackles the challenges in IoT-picosat networks from a constellation level. This result also implies that CosMAC can support a larger scale of device deployments, compared to other baselines.

8.2 Uplink Fairness

Given that CosMAC forces devices in overlapping regions to transmit more conservatively, we evaluate the uplink fairness of our random access scheme. Fig. 11a plots per device uplink throughput across varying geographic locations for the 25 packets a day traffic pattern. We observe that almost all devices roughly achieve similar uplink throughput during the simulation. Although CosMAC penalizes devices in the overlap, fairness is possible due to the continuously moving satellite footprints. We hypothesize that uplink fairness stems from (a) the design goal of constellations to provide constant coverage, and (b) CosMAC's flow control mechanism. Through meticulous orbit planning, satellite constellations like SWARM (CosMAC's simulated constellation) optimize for global coverage. However, such coverage can still have variability, e.g., more overlaps closer to poles due to near-polar orbits. CosMAC's α tuning-based fairness strategy responds to such variation and prevents starvation of devices in high-overlap-prone geographic regions towards the North and South poles.

8.3 Downlink Robustness

CosMAC's downlink scheduling is designed to generate a schedule that is robust against packet loss, attempting to ensure that data is downlinked to at least $K \geq 2$ ground stations. Fig. 11c reports the CDF of the number of ground stations that received each packet. The CDF is computed across all packets. The median (90th percentile) number of ground stations that receive a satellite transmission is 12 (140). Unlike prior scheduling algorithms (L2D2) that only transmit using a point-to-point link schedule, CosMAC significantly improves downlink robustness by leveraging the advantage of the large coverage area of IoT-picosat footprints.

9 RELATED WORK

Satellite Networking: Previous work in the satellite networking domain has mostly focused on satellite ground station architectures [25, 42, 69], RF link prediction [17, 27, 72], and orbital and power modeling [21, 28, 32, 55], as well as analyzing large-scale constellations [44, 45, 65]. However, the majority of this prior work focuses on broadband satellites, which have much fewer constraints than IoT picosats in terms of coordination and data communication. In contrast, previous work specifically on IoT satellite networking is limited and tends to focus on improving networks around individual satellites [22, 31, 33, 37, 56, 71].

Uplink Protocols: Due to current hardware constraints on both IoT devices and picosats, complex network medium access control cannot be used. Therefore, data uplink to picosats can only be done using random access MAC protocols [15]. State-of-the-art uplink protocols are limited to Aloha-based schemes because other random access protocols, such as CSMA, would be difficult to implement without direct coordination among IoT devices. These Aloha variants typically modify IoT device transmit backoff time by calculating the number of devices within range and estimating the trajectory/link variation of a single satellite [70]. In contrast to terrestrial networks, where Aloha with overlapping cells has been explored to a much more limited extent [53], the satellite network case is largely different because the cells are constantly moving. CosMAC is a MAC layer solution that can be easily integrated with various popular PHY layer approaches. For example, recent work [15, 51, 67] has discussed long-range frequency hopping spread spectrum (LR-FHSS) as a mechanism to improve long-range uplink performance in satellite networks at the PHY layer.

Downlink Scheduling: Although there has been significant research on downlink scheduling for satellite ground station links, much of this work does not account for the real-world complexity of time-varying wireless links [18, 35, 64]. The few studies that address this complexity primarily focus on broadband satellites that can beamform and enable one-to-one scheduling of satellites and ground station nodes [29, 69]. In contrast, CosMAC's scheduling approach is tailored for picosats and addresses interference in the satellite network resulting from omnidirectional antennas used for downlink.

10 DISCUSSION

We discuss some limitations and future work below:

Network Size Estimation: As mentioned in Section 4.2, CosMAC's satellites rely on the knowledge of IoT device locations to estimate and broadcast the number of devices in their footprint. In today's IoT networks, these locations are easily maintained by the network operators and configured at setup time. Managing device locations will likely become more difficult for networks with decentralized ownership and/or high mobility. Some recent work [50, 54] and potential future work can facilitate to better configure CosMAC's overlap aware uplink protocol through estimating the distribution of IoT devices in different regions of the Earth.

Reliable Data Transfer: Today's IoT-picosat networks do not ensure reliability and lack the ability to send per-packet acknowledgments. Therefore, IoT data is delivered in a best effort manner and can be lost. Enabling reliable data transfer, under the resource constraints of picosats, is a challenging and interesting research problem.

Ground Station Backhaul: Although CosMAC provides an effective means of coordinating communication between IoT devices and the ground stations, we have not explored its effects on backhaul from the ground station to the cloud. On the backhaul side, fault tolerance of ground stations may be a key factor to consider, especially in a distributed ground station network where most ground stations have a low hardware complexity. In this scenario, it may even be helpful for IoT satellites to transmit multiple copies of the data to multiple ground stations under its footprint.

11 CONCLUSION

We present CosMAC – a constellation-aware medium access and scheduling system for picosat-based IoT networks. CosMAC leverages satellite-specific insights at the constellation scale. Specifically, CosMAC introduces: 1) a novel overlap-aware uplink protocol for IoT devices to satellites, and 2) a novel one-to-many downlink scheduling algorithms for satellites to ground stations. We evaluate CosMAC using a combination of real-world measurements and trace-driven simulation. We found that CosMAC's new link-layer enhancements improve IoT picosat networks for important network features across the board including network throughput, fairness, and robustness. We hope our measurements, insights, and simulation frameworks will enable the academic community to identify and solve new challenges in this space.

REFERENCES

[1] 2021. Benefits of Direct-To-Satellite IoT and Use Cases to Avoid. https://www.leverege.com/blogpost/benefits-of-direct-to-satellite-iot-and-use-cases-to-avoid

[2] 2021. SpaceX buys out satellite IOT startup SWARM technologies. https://www.satellitetoday.com/business/2021/08/09/spacex-buys-out-satellite-iot-startup-swarm-technologies/.

[3] 2023. Celestrak. https://celestrak.org/.

[4] 2023. EchoStar. https://www.echostar.com/.

[5] 2023. FOSSA Systems. https://fossa.systems/.

[6] 2023. Ingenu. https://www.ingenu.com/?doing_wp_cron=1695275274.5296790599822998046875.

[7] 2023. Myriota Tank monitoring, simplified. https://myriota.com/tank-monitoring-satellite-iot/.

[8] 2023. Nanosatellite Launch Forecasts - Track Record and Latest Prediction. https://tinyurl.com/2p948kwv.

[9] 2023. Starlink. https://www.starlink.com/.

[10] 2023. Swarm. https://swarm.space/.

[11] 2023. SX1302CSSXXXGW1. https://www.semtech.com/products/wireless-rf/lora-core/sx1302cssxxxgw1.

[12] 2024. TinyGS – The Open Source Global Satellite Network.

[13] 2024. Wyld Networks: Sensor-to-Satellite IoT. https://wyldnetworks.com/wyldconnect.

[14] LoRaWAN Alliance. 2017. LoRaWAN 1.1 Specification. https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1

[15] Guido Alvarez, Juan A. Fraire, Khaled Abdelfadeel Hassan, Sandra Cespedes, and Dirk Pesch. 2022. Uplink Transmission Policies for LoRa-Based Direct-to-Satellite IoT. *IEEE access: practical innovations, open solutions* 10 (2022), 72687–72701. https://doi.org/10.1109/ACCESS.2022.3189647

[16] Amazon Inc. 2024. AWS Ground Station . https://aws.amazon.com/ground-station/.

[17] Radu Arsinte. 2006. Effective Methods to Analyze Satellite Link Quality Using the Built-in Features of the DVB-S Card. (01 2006).

[18] Jeremy Castaing. 2014. Scheduling Downloads for Multi-Satellite, Multi-Ground Station Missions *(Small Satellite Conference)*.

[19] Tusher Chakraborty, Heping Shi, Zerina Kapetanovic, Bodhi Priyantha, Deepak Vasisht, Binh Vu, Parag Pandit, Prasad Pillai, Yaswant Chabria, Andrew Nelson, et al. 2022. Whisper: IoT in the TV White Space Spectrum. In *19th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2022*. USENIX Association, 401–417.

[20] Chiu Chun Chan, Bassel Al Homssi, and Akram Al-Hourani. 2022. Performance Evaluation of Random Access Methods for IoT-over-Satellite. *Remote Sensing* 14, 17 (2022). https://doi.org/10.3390/rs14174232

[21] B.H. Cho and F.C.Y. Lee. 1988. Modeling and analysis of spacecraft power systems. *IEEE Transactions on Power Electronics* 3, 1 (1988), 44–54. https://doi.org/10.1109/63.4330

[22] Houcine Chougrani, Steven Kisseleff, Wallace A Martins, and Symeon Chatzinotas. 2021. NB-IoT Random Access for Nonterrestrial Networks: Preamble Detection and Uplink Synchronization. *IEEE Internet of Things Journal* 9, 16 (2021), 14913–14927.

[23] Devin Coldewey. 2020. Swarm prices out its orbital IoT network's hardware and services. https://tinyurl.com/2zjd7sjp

[24] Taniya Das. 2018. Performance Analysis of Slotted Aloha Protocol. https://doi.org/10.13140/RG.2.2.35711.97445/1

[25] Iñigo del Portillo, Bruce Cameron, and Edward Crawley. 2018. Ground segment architectures for large LEO constellations with feeder links in EHF-bands. In *2018 IEEE Aerospace Conference*. 1–14. https://doi.org/10.1109/AERO.2018.8396576

[26] Bradley Denby and Brandon Lucia. 2020. Orbital edge computing: Nanosatellite constellations as a new class of computer system. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. 939–954.

[27] Menachem Manis Domb Alon and Guy Leshem. 2022. Satellite to Ground Station, Attenuation Prediction for 2.4–72 GHz Using LTSM, an Artificial Recurrent Neural Network Technology. *Electronics* 11, 4 (2022). https://doi.org/10.3390/electronics11040541

[28] Tom Etchells and Lucy Berthoud. 2019. Developing a Power Modelling Tool for CubeSats. Annual AIAA/USU Conference on Small Satellites, SSC 2019 ; Conference date: 03-08-2019 Through 08-08-2019.

[29] Huilong Fan, Zhan Yang, Shimin Wu, Xi Zhang, Jun Long, and Limin Liu. 2021. An efficient satellite resource cooperative scheduling method on Spatial Information Networks. https://www.mdpi.com/2227-7390/9/24/3293

[30] Uriel Feige and Daniel Reichman. 2015. Recoverable values for independent sets. *Random Structures & Algorithms* 46, 1 (2015), 142–159.

[31] Lara Fernandez, Joan Adria Ruiz-De-Azua, Anna Calveras, and Adriano Camps. 2020. Assessing LoRa for satellite-to-earth communications considering the impact of ionospheric scintillation. *IEEE access* 8 (2020), 165570–165582.

[32] Roberto Flores, Burhani Makame Burhani, and Elena Fantino. 2021. A method for accurate and efficient propagation of satellite orbits: A case study for a Molniya orbit. *Alexandria Engineering Journal* 60, 2 (Apr 2021), 2661–2676. https://doi.org/10.1016/j.aej.2020.12.056

[33] Juan A Fraire, Sandra Céspedes, and Nicola Accettura. 2019. Direct-to-satellite IoT-a survey of the state of the art and future research perspectives: Backhauling the IoT through LEO satellites. In *Ad-Hoc, Mobile, and Wireless Networks: 18th International Conference on Ad-Hoc Networks and Wireless, ADHOC-NOW 2019, Luxembourg, Luxembourg, October 1–3, 2019, Proceedings 18*. Springer, 241–258.

[34] Juan A. Fraire, Santiago Henn, Fabio Dovis, Roberto Garello, and Giorgio Taricco. 2020. Sparse Satellite Constellation Design for LoRa-based Direct-to-Satellite Internet of Things. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9348042

[35] C. Fuchs and F. Moll. 2015. Ground station network optimization for space-to-ground optical communication links. *IEEE/OSA Journal of Optical Communications and Networking* (2015).

[36] Amalinda Gamage, Jansen Christian Liando, Chaojie Gu, Rui Tan, and Mo Li. 2020. Lmac: Efficient carrier-sense multiple access for lora. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–13.

[37] Alessandro Guidotti, Alessandro Vanelli-Coralli, Alberto Mengali, and Stefano Cioni. 2020. Non-terrestrial networks: Link budget analysis. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 1–7.

[38] Jetmir Haxhibeqiri, Ingrid Moerman, and Jeroen Hoebeke. 2018. Low overhead scheduling of LoRa transmissions for improved scalability. *IEEE Internet of Things Journal* 6, 2 (2018), 3097–3109.

[39] International Telecommunications Union. 2019. *ITU P.838: Specific attenuation model for rain for use in prediction methods*. Technical Report.

[40] International Telecommunications Union. 2019. *ITU P.839 : Rain height model for prediction methods* . Technical Report.

[41] International Telecommunications Union. 2019. *ITU P.840: Attenuation due to clouds and fog*. Technical Report.

[42] William Ivancic. 2003. Architecture Study of Space-Based Satellite Networks for NASA Missions. *NASA/TM* (2003).

[43] V. Jacobson. 1988. Congestion Avoidance and Control. In *Symposium Proceedings on Communications Architectures and Protocols* (Stanford,

California, USA) *(SIGCOMM '88)*. Association for Computing Machinery, New York, NY, USA, 314–329. https://doi.org/10.1145/52324.52356

[44] Pauline Jakob, Seiichi Shimizu, Shoji Yoshikawa, and Koki Ho. 2019. Optimal Satellite Constellation Spare Strategy Using Multi-Echelon Inventory Control. *Journal of spacecraft and rockets* (May 2019), 1–13. https://doi.org/10.2514/1.A34387

[45] Sijing Ji, Di Zhou, Min Sheng, Dong Chen, Liang Liu, and Zhu Han. 2022. Mega Satellite Constellations Analysis Regarding Handover: Can Constellation Scale Continue Growing?. In *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. 4697–4702. https://doi.org/10.1109/GLOBECOM48099.2022.10001556

[46] Suraj Jog, Jiaming Wang, Junfeng Guan, Thomas Moon, Haitham Hassanieh, and Romit Roy Choudhury. 2019. Many-to-Many beam alignment in millimeter wave networks. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. 783–800.

[47] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. 2020. Exploring the" Internet from space" with Hypatia. In *Proceedings of the ACM Internet Measurement conference*. 214–229.

[48] Herber Kramer. 2020. Lacuna Constellation. https://www.eoportal.org/satellite-missions/lacuna-constellation

[49] Zeqi Lai, Hewu Li, Yangtao Deng, Qian Wu, Jun Liu, Yuanjie Li, Jihao Li, Lixin Liu, Weisen Liu, and Jianping Wu. 2023. {StarryNet}: Empowering Researchers to Evaluate Futuristic Integrated Space and Terrestrial Networks. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 1309–1324.

[50] Diego Maldonado, Juan A Fraire, Pablo Ilabaca, Hervé Rivano, and Sandra Céspedes. 2023. Network Size Estimation for LoRa-Based Direct-to-Satellite IoT. In *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*. IEEE, 1–6.

[51] Alireza Maleki, Ha H. Nguyen, Ebrahim Bedeer, and Robert Barton. 2022. D2D-aided LoRaWAN LR-FHSS in Direct-to-Satellite IoT Networks. *arXiv* (2022). https://doi.org/10.48550/arxiv.2212.04331

[52] Microsoft. 2024. Azure Orbital. https://azure.microsoft.com/en-us/services/orbital/.

[53] Gam D. Nguyen, Jeffrey E. Wieselthier, and Anthony Ephremides. 2010. Random access in wireless networks with overlapping cells. *IEEE Transactions on Information Theory* 56, 6 (Jun 2010), 2887–2892. https://doi.org/10.1109/TIT.2010.2046194

[54] Pablo Ilabaca Parra, Samuel Montejo-Sánchez, Juan A Fraire, Richard Demo Souza, and Sandra Céspedes. 2022. Network size estimation for direct-to-satellite iot. *IEEE Internet of Things Journal* 10, 7 (2022), 6111–6125.

[55] Cristina Puente, Maria Ana Sáenz-Nuño, Augusto Villa-Monte, and José Angel Olivas. 2021. Satellite orbit prediction using big data and soft computing techniques to avoid space collisions. *Mathematics* 9, 17 (Aug 2021), 2040. https://doi.org/10.3390/math9172040

[56] Zhicheng Qu, Gengxin Zhang, Haotong Cao, and Jidong Xie. 2017. LEO satellite constellation for Internet of Things. *IEEE access* 5 (2017), 18391–18401.

[57] FCC Report. 2022. *Company Profile: Hiber Inc.*

[58] FCC Report. 2022. *Company Profile: Swarm Technologies Inc.*

[59] FCC Report. 2023. *Company Profile: Myriota Pty Ltd.*

[60] Microsoft Research. 2023. *CosmicBeats-Simulator: A space simulation platform that caters to individuals with diverse research interests, including networking, AI, computing, and more. Unlike traditional simulators tied to specific research applications, our design allows for seamless integration of various space-related research verticals.*

[61] Vaibhav Singh, Tusher Chakraborty, Suraj Jog, Om Chabra, Deepak Vasisht, and Ranveer Chandra. 2024. Exploiting Satellite Doppler for Reliable and Faster Data Download in IoT Satellite Networks. *GetMobile: Mobile Computing and Communications* 27, 4 (2024), 11–14.

[62] Vaibhav Singh, Tusher Chakraborty, Suraj Jog, Om Chabra, Deepak Vasisht, and Ranveer Chandra. 2024. Spectrumize: Spectrum-efficient Satellite Networks for the Internet of Things. In *USENIX Symposium on Networked Systems Design and Implementation, Santa Clara, CA*.

[63] Vaibhav Singh, Akarsh Prabhakara, Diana Zhang, Osman Yağan, and Swarun Kumar. 2021. A community-driven approach to democratize access to satellite ground stations. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 1–14.

[64] Sara Spangelo, James Cutler, Kyle Gilson, and Amy Cohn. 2015. Optimization-based scheduling for the single-satellite, multi-ground station communication problem. *Computers & Operations Research* (2015).

[65] Tianyu Sun, Min Hu, and Chaoming Yun. 2022. Low-Orbit Large-Scale Communication Satellite Constellation Configuration Performance Assessment. *International Journal of Aerospace Engineering* 2022 (Mar 2022), 1–8. https://doi.org/10.1155/2022/4918912

[66] SWARM Technologies. 2020. *Amended Filing on a Satellite Space Stations filing.*

[67] Muhammad Asad Ullah, Konstantin Mikhaylov, and Hirley Alves. 2022. Analysis and Simulation of LoRaWAN LR-FHSS for Direct-to-Satellite Scenario. *IEEE Wireless Communications Letters* 11, 3 (2022), 548–552. https://doi.org/10.1109/LWC.2021.3135984

[68] Deepak Vasisht, Zerina Kapetanovic, Jong-ho Won, Xinxin Jin, Ranveer Chandra, Ashish Kapoor, Sudipta N. Sinha, Madhusudhan Sudarshan, and Sean Stratman. 2017. Farmbeats: An IoT Platform for Data-Driven Agriculture. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation* (Boston, MA, USA) *(NSDI'17)*. USENIX Association, USA, 515–528.

[69] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. 2021. L2D2: Low Latency Distributed Downlink for LEO Satellites. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference* (Virtual Event, USA) *(SIGCOMM '21)*. Association for Computing Machinery, New York, NY, USA, 151–164. https://doi.org/10.1145/3452296.3472932

[70] Kai Vogelgesang, Juan A. Fraire, and Holger Hermanns. 2021. Uplink Transmission Probability Functions for LoRa-Based Direct-to-Satellite IoT: A Case Study. In *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 01–06. https://doi.org/10.1109/GLOBECOM46510.2021.9685152

[71] Alexander M Zadorozhny, Alexander A Doroshkin, Vasily N Gorev, Alexander V Melkov, Anton A Mitrokhin, Vitaliy Yu Prokopyev, and Yuri M Prokopyev. 2022. First Flight-Testing of LoRa Modulation in Satellite Radio Communications in Low-Earth Orbit. *IEEE Access* 10 (2022), 100006–100023.

[72] Bircan Çalışır and Ayhan Akbal. 2022. A New RF Satellite Link Analyzing and Antenna Effect on Satellite Communication. *Tehnički glasnik* 16 (09 2022), 550–556. https://doi.org/10.31803/tg-20220405153200