PicoNet: A Network Stack for Next Generation IoT Satellite Networks

Paper # 642, 12+5 pages

Abstract

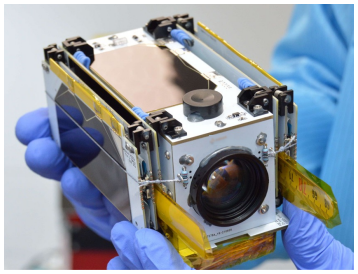
Pico-satellite (picosat) constellations aim to become the de facto connectivity solution for Internet of Things (IoT) devices. These constellations rely on a large number of small picosats and offer global plug-and-play connectivity at low data rates, without the need for Earth-based gateways. As picosat constellations scale, their traditional networking stack becomes the bottleneck because it does not incorporate constraints specific to picosat constellations – orbital motion, limited satellite power, and low complexity hardware. In this paper, we describe PicoNet – a new networking stack specifically designed for IoT-Picosat networks. PicoNet includes a new medium access approach for uplink from IoT to picosats and a new network layer that schedules downlink traffic from satellites. We empirically evaluate PicoNet using measurements from three picosats and large scale trace-driven simulations for a 173 picosat network supporting 100k devices. Our results demonstrate that PicoNet can improve the overall network throughput by 6.5× over prior state-of-the-art satellite medium access schemes.

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 [1, 5–7, 11].

Direct-to-satellite (DtS) is the prevalent connectivity model for picosat-IoT connectivity [3, 8, 33]. In DtS, IoT devices, first, receive a beacon from a picosat overhead, notifying them to uplink data to the satellite. Once they receive the beacon, the IoT devices directly transmit their data to picosats in orbit. Due to their low orbits, standard LoRa packets transmitted by IoT devices can be decoded at picosats. 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. Conversely, for picosat networks, IoT devices directly talk to picosats which are power-starved because their small size (Fig. 1) prevents them from using large solar panels and large batteries. In addition, picosats move fast with respect to IoT devices and ground stations on Earth, owing to their low orbits. Due to orbital motion, a picosat cannot connect to a ground-device for more than ten minutes at a stretch [62]. Thus, picosats also 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).



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.*

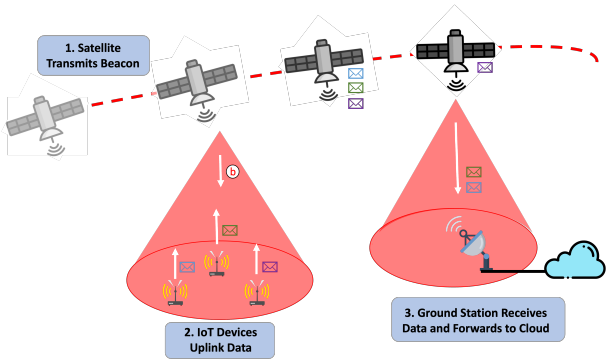


Figure 2: *Data flow in Typical IoT PicoSat Networks.*

Due to these constraints and the growing scale of picosat networks, a networking stack for IoT-picosat networks must solve the following challenges (highlighted in Fig. 3):

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 few million Km^2 containing thousands of devices.

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 omni-directional antennas, making it impossible to direct their transmissions toward a specific satellite. Consequently, when they transmit, they generate interference at multiple satellites, resulting in multiple collisions stemming from a single transmission. Finally, there is limited spectrum available (usually less than 1 MHz) for satellite-ground communications limiting frequency-domain multiplexing.

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. There has been a lot of research [17, 28, 34, 56, 62] in identifying the best set of links to activate at any time. However, picosats use omni-directional antennas due to their limited size and weight. As such, they cannot perform beamforming and perform one-to-one communication with ground stations. Instead, they establish *one-to-many* links. Specifically, when one satellite is allowed to transmit, it's signal is received at (or interferes at) *multiple* ground stations. These ground stations can no longer be used to receive signals from other satellites. Existing scheduling algorithms do not account for such one-to-many interference patterns.

In this paper, we present a new networking stack for IoT-picosat networks – PicoNet. PicoNet addresses the above challenges to improve scalability, robustness, and performance (both throughput and latency) of picosat networks. PicoNet 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, PicoNet takes a unique *constellation-scale collaborative* approach that reveals new optimization opportunities. PicoNet's design consists of three key components:

(i) Mitigating Uplink Collisions: IoT-picosat networks cannot use carrier-sense or centralized coordination to avoid collisions from IoT devices to satellites. However, we observe

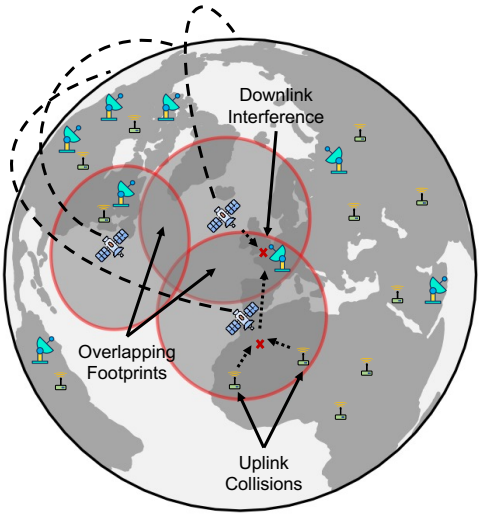


Figure 3: *IoT-picosat network – IoT devices directly send data to picosats and often experience collisions. Additionally, satellites simultaneously downlinking data to ground stations in an overlap region interfere.*

that the costs of a single transmission across different IoT devices are not equal. For example, if a device is in the range of N satellites, its transmissions prevent these satellites from receiving transmissions from other IoT devices (collision) as illustrated by Fig. 6. Based on this observation, we create a random access protocol that penalizes such heavy hitters, i.e., devices that cause interference at multiple satellites from being in an overlapping footprint region. This approach contrasts to medium access approaches used in terrestrial networks, where such schemes would be unfair to a *constant* set of devices. However, in satellite networks, the orbital motion of satellites ensures that their footprints (and the overlapping regions) move across the Earth. Therefore, the set of starved devices changes over time. Our approach minimizes collisions and improves the overall uplink throughput.

(ii) Dynamic Flow Control: Due to the variability in the data generation rate of devices across time (e.g., due to event-driven traffic), we need to regularly adapt the sending rate from IoT devices to the satellite. To achieve this objective with minimal overhead, we design a collaborative flow control scheme. With PicoNet, satellites provide feedback to all devices under their footprint for adjusting their transmission rates. The satellite determines when devices need to scale back their transmissions by detecting and measuring collisions received over time. If the number of collisions measured is significantly larger than the amount expected from periodic traffic, the satellite can infer that devices in its footprint are generating higher amounts of data than expected, and can include a 'backoff' signal in its periodic beacon message.

(iii) One-to-many Scheduling: PicoNet 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 station available to each satellite. By achieving the above goal, PicoNet is able to optimize the satellite downlink schedule for both network reliability and network robustness. We formulate PicoNet'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 PicoNet, 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 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 employ the Microsoft Research CosmicBeats Simulator [55], customizing the simulator parameters using data and models derived from our real-world setup. We conduct simulations involving constellations comprising 173 satellites, 100,000 devices, and 1,048 ground stations. Our results show that PicoNet can improve net end-to-end throughput by 6.5× compared to state-of-the-art baselines.

Our contributions in PicoNet are:

- We describe a new random access uplink technique that exploits constellation-scale contention to improve net uplink throughput.
- We design a novel downlink scheduler for satellite-ground station links that account for interference from other simultaneously transmitting satellites.
- We build a real-world research platform consisting of three picosats, two ground stations, and multiple IoT nodes.
- We leverage the measurements from real-world setup to perform large scale evaluation of PicoNet using the parameterized Microsoft CosmicBeats Simulator.

2 Background

Picosats are an emerging class of low-cost, low-complexity satellites. Around 2000 picosats have been launched in orbits by 2022 [9]. Picosats are small in size, making it easy to launch hundreds of picosats in a single launch. However, the small size also limits their hardware capabilities (see Fig. 1). These satellites operate in low earth orbits (LEO), around 450 to 650 km above Earth. Due to their low orbits, they scan different parts of the Earth across time and their communication with ground devices happens in small time windows – each contact lasts for 5-10 minutes, with few such contacts per day.

IoT Devices and DtS: IoT devices typically communicate to picosats using direct-to-satellite (DtS) model. Although power-constrained, IoT devices are able to achieve direct transmission to the satellite by using a combination low-power modulation schemes such as LoRa and duty cycling. DtS models enable ease of use and deployment. IoT device users can simply turn a device on and connect to the Internet. This connectivity is especially beneficial in areas without terrestrial connectivity like farms, forests, oceans, etc. Even in cities, DtS models enable setup-free deployment.

An alternative model for such deployments include using gateways to gather data from multiple IoT devices and relay to satellites. Such gateways require some setup costs, but can reduce power burden on IoT devices. However, gateways are only useful in small and dense deployments. In sparse or large-scale deployments, we need to deploy multiple gateways. While we focus on DtS connectivity models, the techniques presented in PicoNet are generalizable to gateway-based architectures. A gateway simply aggregates data in local zones (around one sq. Km.), but doesn't reduce the overall network demand or contention.

Data Communication: IoT picosat networks operate within the VHF to S bands (100 MHz - 2.4 GHz) and, typically communicate at low data rates (few Kbps). Low frequency data communication ensures that the wireless links between the IoT devices and satellites are not significantly affected by weather and atmospheric conditions, unlike high frequency broadband satellites such as those operated by Starlink [10]. Low frequencies also experience lower path loss, reducing the power requirement on IoT devices and picosats. Due to lower power-budget and long range requirements, LoRa is a popular choice among commercial IoT-picosat service providers like SWARM [11], FOSSA [6],Lacuna [45], and EchoStar [5]. Our testbed picosat also incorporates a LoRa gateway.

There are three types of communication in such networks: (i) uplink from IoT devices to the satellite, (ii) satellite broadcast beacons that contain metadata, and (c) downlink from the satellite to ground stations. While uplink and downlink use separate frequency bands, all uplinking devices (or all downlinking devices) operate in the same frequency range. Some constellations divide the uplink band into multiple channels, e.g., e.g., SWARM [11] picosat constellation has total 14 channels of 125 kHz. Given the small number of channels, the contention within each channel needs to be actively managed.

For downlink, traditional satellites and even cubesats (such as Planet Dove [24] for earth observation) use directional antennas at high frequencies (X-band) to establish one-to-one links with large ground stations [25]. In contrast, picosats operate at lower frequencies, are smaller in size, and are therefore forced to use omni-directional antennas [52–54], which cause large on-ground interference.

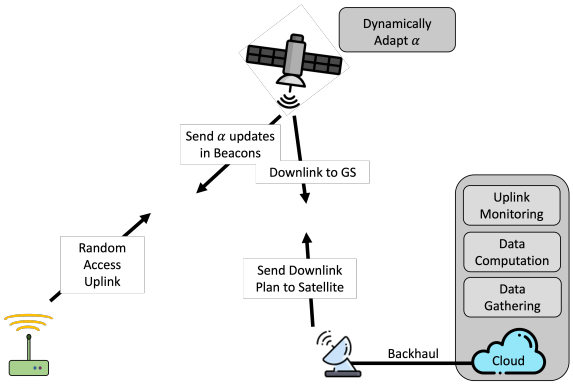


Figure 4: *Data flow for PicoNet.*

Ground Stations: Traditional satellite networks utilize a few (4-10) dedicated multi-million dollar ground stations. Recently, there have been proposals (both in academia [59, 62] and industry [15, 48]) for large-scale, distributed deployments of low complexity ground stations that can serve multiple constellations. For example, Azure orbital [48] and AWS Ground Station [15] offer ground station time by the minute to satellites. The data from these stations is available in the respective cloud. This allows satellite operators to be free from the overhead of setting up ground infrastructure. For picosatellites, such distributed architectures are increasingly taking shape, e.g., TinyGS [2] has deployed over 1000 ground stations for picosats, with multi-constellation support. Picosat ground stations use off-the-shelf hardware like omni-directional antennas, without any rotating mounts. Most TinyGS ground stations are receive-only, and are not uplink capable, i.e., they cannot provide feedback or send acks to the satellite.

3 PicoNet Overview and Goals

Our goal is to identify bottlenecks in large scale picosat-IoT constellations and design new primitives that remove these bottlenecks. Specifically, our design aims to achieve the following objectives:

- **Scale:** We aim to support a large number of satellites (hundreds satellites) and IoT devices (100k devices).
- **Performance:** We aim to maximize the end-to-end throughput and latency 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 picosatellites.

Fig. 4 shows an overview of PicoNet's design. As shown, PicoNet's uplink algorithm (Sec. 4.2) runs on each IoT device and coordinates medium access. The rate of upload, i.e., the flow rate, is mediated by our flow control algorithm (Sec. 4.3) that communicates information through a picosat's periodic beacon transmissions. The centralized scheduler, running on the cloud, identifies optimal schedule for downlink transmissions from picosats (Sec. 5). The schedule is communicated to the picosats through ground stations. The downlink transmissions follow this schedule and can be received at one or more ground stations. The ground stations pass all the received data to the cloud, where it is gathered and processed.

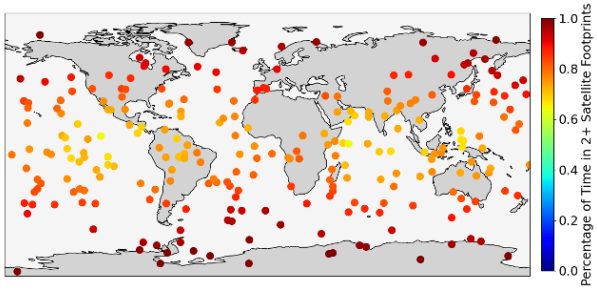


Figure 5: *IoT device percentage of time under 2+ footprints for 173 satellites based on geographic location. IoT device spends 83% of time on average in an overlapping region.*

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 availability, and segregated operation of picosat-borne 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 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, within the channels.

4.1 Observations and Design Choices

Past work [14, 20, 63] 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 in the range of multiple satellites, none of those satellites can receive a collision-free packet in that slot. We demonstrate such multi-satellite contention in Fig. 3, which is evident when we analyze uplink at a constellation-scale.

We analyze a commercially operational picosat constellation – the SWARM constellation [11] with 173 satellites. We place IoT devices uniformly across the globe and observe the

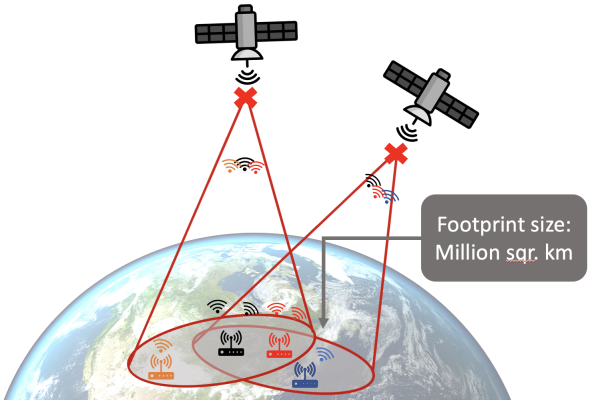


Figure 6: *Large IoT Satellite footprints result in more uplink collisions caused by devices in overlapping footprints.*

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. 5. 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 cycle, even though they scan different parts of the Earth during the cycle. At a given time, devices may simultaneously be in up to 33 (median: 5) footprints. The time spent in overlapping footprints is not continuous due to satellite's orbital motion and Earth's diurnal motion. Over time, an IoT device sporadically enters and exits overlapping footprints.

Based on this analysis, our core strategy is to penalize these heavy hitters from an overlapping region, i.e., devices creating collision at multiple satellites. Such penalization is sub-optimal in terrestrial networks, because penalizing devices in overlapping footprints leads to permanent starvation for a subset of devices and is unfair to such devices. However, in satellite networks, the orbital motion causes natural variation in footprints over time at the scale of minutes, leading to the penalty shifting across space and time.

4.2 Medium Access Scheme

Our goal is ensuring seamless integration of our solution into current picosat networks. Therefore, we design our medium access scheme such that it does not require any hardware changes. Currently, in a picosat-IoT network, when an IoT device has data to transmit, it searches for an overhead satellite by listening on the predefined satellite beacon channel. Satellites transmit very short beacons at regular intervals similar to LoRaWAN Class B protocol [13]. After receiving the beacon, the device then transmits its data using a random-access protocol like ALOHA [52–54, 63]. Our solution introduces two modifications. First, our satellite beacon includes metadata estimating the number of devices within the satellite footprint. Second, each device d now transmits data at time step t after the beacon with a given transmission probability p_d. We design p_d to account for potential multi-satellite contention. These two modifications serve as the foundation for our distributed medium access scheme.

PicoNet utilizes network metadata to determine the geographic distribution of IoT devices. It utilizes this information to estimate the number of devices within the footprint of each picosat at any given time. Given the deterministic orbital motion of a satellite, it is easy to update that info on the satellite from PicoNet's central coordinator. The satellite embeds this info in the beacon based on its location. An IoT device uses this information to compute p_d. 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 (over estimate). 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 flow control as described below.

4.3 Flow Control

An IoT network encompasses two types of traffic: a) periodic, where sensors report data at predefined intervals [61], and b) event-driven, where data reporting is triggered by specific events like forest fire detection, rain monitoring, or tracking asset movements, among others. While the first type could be predicted in advance, the latter is highly unpredictable and often takes precedence [18]. 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 a 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.

In this particular context, PicoNet 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. 4). 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. 5, 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, PicoNet dynamically adjusts α to guarantee fairness for devices spending more time in densely overlapping areas (see Fig. 13).

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 [18, 37]. How does one detect transmissions and collisions below the noise floor? PicoNet 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 [12]. CAD detects the presence of LoRa symbols on a channel with minimal power consumption [35], even when the complete LoRa packet cannot be decoded. PicoNet 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, PicoNet can estimate collision events. PicoNet uses CAD to observe unusually long channel inactivity as well. While PicoNet 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 transmission 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. 4). 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 trend 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 heavily 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 [42]. 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 [17, 28, 34, 56, 62]. 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 in Fig. 7. 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 [2], 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 phenomenon 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. PicoNet should choose these ground stations, such that packet losses from the satellite to the chosen ground stations are uncorrelated. Hence, PicoNet's downlink scheduler must:

(1) *Optimize the downlink throughput by maximizing 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 [17, 28, 34, 56, 62]. 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 [38–40]. 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 min 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 multiple constellations of satellites, where each constellation is operating at a different frequency band.

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 $\{g_1, g_2, \ldots, g_M\}$.

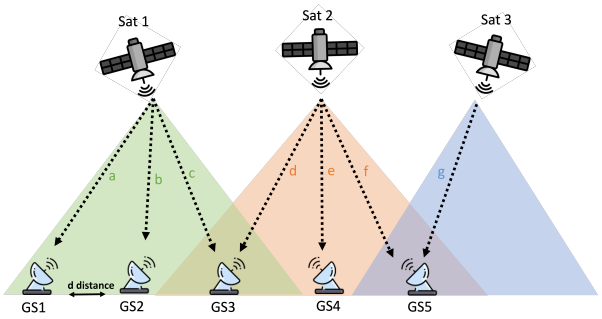
Vertices: Fig. 7(a) shows the physical configuration of a downlink network at time instance t, and Fig. 7(b) shows the corresponding graph formulation for the network at t. As shown, the vertices $v_i \in V$ in the graph correspond to feasible links (s_i, g_j) that can be used for downlink communication. 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 their own corresponding vertex.

Edges: Given the graph with vertices V, scheduling is equivalent to choosing vertices (or satellite-ground station links). However, as we noted, PicoNet'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. PicoNet accounts for three conflict types:

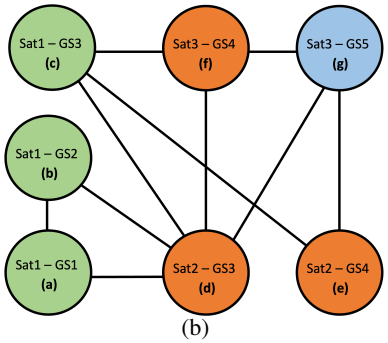
(1) *Ground Station Conflict:* A ground station g_j, can receive from only 1 satellite at any given time. Hence, there will be an edge between all vertices that share the same ground station. That is, all vertex pairs (v_x, v_y) where $v_x = (s_i, g_j)$ and $v_y = (s_k, g_j)$ for all $j \in \{1, 2, \ldots M\}$, 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. 7(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. 7(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 in order 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.



(a)



(b)

Figure 7: *Centralized Downlink Scheduling: (a) Physical configuration of picosats and available ground stations, (b) Graph formulation that accounts for interference.*

Figure 8: *Our ground station deployment.*

Note that this conflict is relevant only to satellites belonging to the same constellation, since satellites in different constellations operate at different frequencies. Concretely, to ensure that PicoNet's scheduler only picks interference-free transmissions, 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 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:* PicoNet'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 to receive from a single satellite to ensure reliability. However, for reliability, we want the 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 PicoNet's algorithm chooses multiple ground stations for a satellite that are all closely located, then these transient packet losses are likely to occur simultaneously across all the chosen ground stations. To address this, PicoNet adds edges between vertices corresponding to the same satellite, if the two ground stations are closer than some 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. 7, where GS1 and GS2 being close leads to an edge between vertices (a) and (b).

5.3 PicoNet's Scheduling Algorithm

Given graph $G_t(V, E)$, PicoNet'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 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 PicoNet in three key ways.

Link Capacities: Different satellite-ground station links have different capacities and achievable data rates. PicoNet'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 to each vertex $v_i \in V$. 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 [29].

Weight Updates: This formulation does not account for fairness, yet. The algorithm may keep scheduling the same set of vertices (links) repeatedly and starve certain links. To address this, we modify the weights as $w_i(t) = r_i(t) \times LTS_i(t)$ for each vertex $v_i \in V$. $r_i(t)$ is the RSSI for link corresponding to v_i at time t. $LTS_i(t)$ is the **L**ast **T**ime **S**lot at which the satellite s_i corresponding to the link in v_i transmitted to any ground station (not just the ground station corresponding to v_i). At the start of the scheduling algorithm, the value of $LTS_i(t = 1)$ for all vertices is set to 1. As the algorithm schedules links in each time slot, the weights get updated. Specifically, at time slot t, if a satellite was scheduled to *any* ground station, then all vertices in the graph corresponding to that satellite have their $LTS_i(t+1)$ values reset to 1. Else, for satellites that did not transmit to any ground station, their vertex weights update as follows: $LTS_i(t+1) = LTS_i(t) + 1$. These updated weights are then used by PicoNet to schedule the set of transmitting links for the next time slot.

Reliability Guarantees: PicoNet's scheduling algorithm above, tries to schedule multiple 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 propose 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 heuristic looks for opportunities where it can remove certain scheduled links and in its place, add additional links for the satellites that had only one assigned ground station in the original schedule. It is possible that even through this heuristic, PicoNet is unable to meet the reliability guarantee for every satellite. In these cases, PicoNet simply drops satellites from the schedule for whom it is unable to find at least two receive ground stations. While this approach 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.

In Appendix A, we present the overall flow of the scheduler (Alg. 1), and our greedy heuristic for ensuring reliability in our schedule in Alg. 2.

6 Research Platform

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

6.1 Real-World IoT Satellite Testbed

We partner with a commercial picosat-IoT 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 for picosat communication. Our picosat is designed and configured to be consistent with the commercial IoT 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 as well. On the ground, we use two bidirectional communication-capable ground stations located in western Europe (shown in Fig. 8) with satellite tracking capability and directional antennas for beamforming. The testbed also includes IoT devices on the ground that use SX162 LoRa radios with omni-directional antennas to communicate directly with our picosat.

6.2 Trace Driven Simulations

We leverage the state-of-the-art, open source, Microsoft CosmicBeats Simulator [55], parameterized using measurements from our real-world setup, to conduct large-scale evaluation. We describe details of our trace-driven simulation below:

Orbital Dynamics and Link: We calculate the orbital movements of satellites using publicly available TLEs obtained from CelesTrak [4]. We use real-world measurements from our satellite and ground device setup to create a wireless link model that computes satellite-ground device link quality and data rate.

Power Modeling: We build a satellite power model based on measurements from our picosat in-orbit and couple with its power allocation algorithm. Power allocation prioritizes critical tasks (e.g., battery heating, flight control) over communication services like downlink transmission and uplink reception. We also model power generation based on sunlight exposure in orbit.

Satellites: Our simulation contains 173 satellites based on the TLEs of the largest IoT satellite constellation, SWARM [11]. The frequency and transmission bandwidth used for all the satellites is 401.7 MHz and 125 kHz respectfully. These values are identical to the transmission frequencies used by the picosats in our real-world deployment.

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 generate rate of every IoT device. In total, we simulate 24 hours of the network. The simulation takes approx. 8 real-hours to run.

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 [2]. This network has 1000+ ground stations and can support multiple constellations.

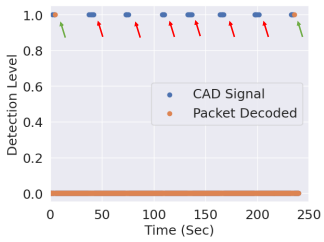
7 Real-world Experiments

7.1 Collision Estimation

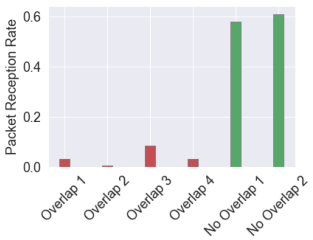
We tested PicoNet's ability to estimate collisions in LoRa using the Channel Activity Detection (CAD) mechanism. In PicoNet, this happens on the satellites, but since we cannot modify the satellites post launch, we used another satellite radio on the ground, with transmissions happening at the satellite (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 of satellite passes in Fig. 9a. 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

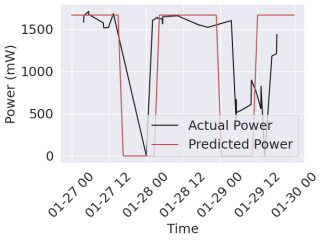
A key claim in PicoNet is that the transmission, made by a device from the overlapping footprints of multiple satellites, will cause collisions at multiple satellites. We validate this



(a) *Collision Est. using CAD.*



(b) *Uplink in overlap.*



(c) *Power generation over time.*

Figure 9: *Real-World Experiments: (a) Long distance collision detection, (b) Time varying satellite-ground station link quality measurements, (c) Power modeling*

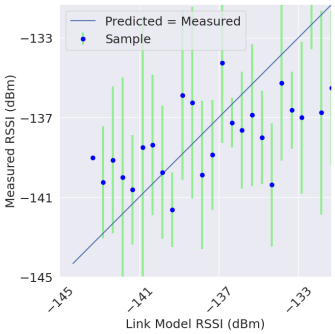


Figure 10: *Predicted vs Measured Link Quality.*

claim through a real-world experiment. We setup two IoT devices transmitting uplink to our satellite. We specifically target passes where two of our satellites create an overlap over our IoT device deployment. During these passes, each device synchronously transmits to both satellites. In Fig. 9b, 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 transmitting uplink to a single satellite. The packet reception rate on both the satellites when devices transmitting from the overlap is significantly lower due to collisions. It establishes that a device in overlapping footprints causes interference to all satellites creating that overlap. Because of LoRa's ability to decode some 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. 9c 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 PicoNet 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 our simulator. Fig. 10 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 closely resembles the real-world measured downlink quality.

8 Large Scale Trace-Driven Evaluation

Having validated our key claims and simulator fidelity with real-world experiments, we report large scale results using trace-driven simulations.

Baselines: We use two baselines for *uplink* medium access – **(a) Uplink Transmission Probability Function (UTPF):** Prior work [63] explores modifying the transmission probability of devices based on the number of devices in a single satellite footprint. In this approach, devices have a transmission probability function (TPF) based on the number of devices from the most recent satellite beacon. This is a common variant of the ALOHA random access protocol that is considered as the state-of-the-art medium access uplink protocol for IoT PicoSat networks. **(b) Fixed Probability ALOHA (FP-ALOHA):** Uplink medium access protocols in picosat-IoT networks mostly rely on ALOHA. Therefore, 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** [62] – 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 omni-directional antennas.

8.1 Throughput

PicoNet's primary goal is optimizing IoT-picosat network throughput. We measure throughput through an 8 hour simulation in our 173 satellite-100k device network and plot results in Fig. 11.

Uplink Throughput: Fig. 11a reports throughput in bps across varying data generation patterns for each uplink

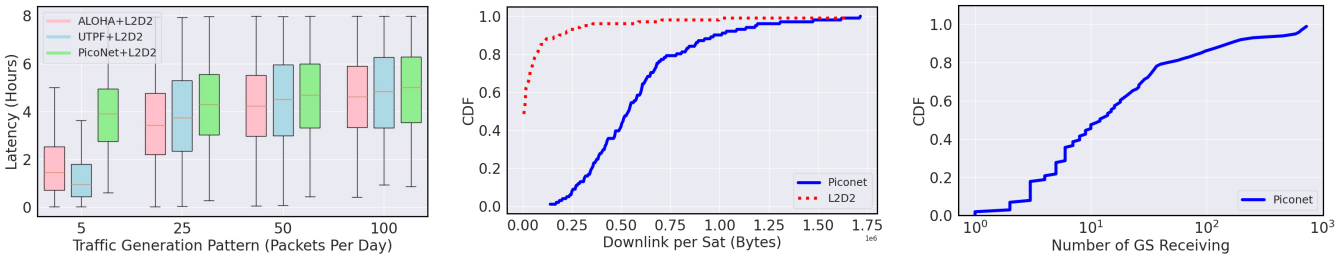
Algorithm	5 Packets	25 Packets	50 Packets	100 Packets
FP-ALOHA (PicoNet inspired)	2631	2477	2381	2389
UTPF	2872	295	273	277
PicoNet	**3580**	**3495**	**3522**	**3467**

(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
PicoNet	**1073**	**1378**	**1388**	**1383**

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

Figure 11: *PicoNet's system performance (a) Comparison of PicoNet uplink throughput to baseline random access schemes (b) Comparison of PicoNet end-to-end throughput to baselines*



(a) *End to End Latency.* (b) *Downlink Throughput* (c) *Downlink Robustness*

Figure 12: *PicoNet's system performance: (a) Uplink Latency (b) Downlink Throughput and (c) Downlink Robustness settings.*

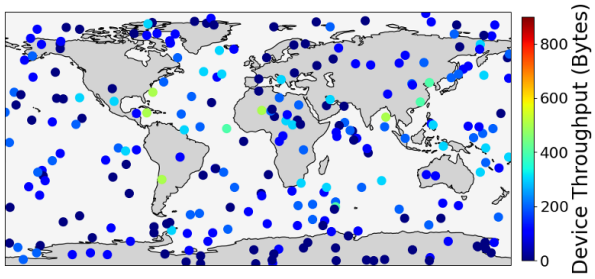


Figure 13: *Distribution of device uplink using PicoNet. Uniform distribution indicates uplink fairness among devices.*

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 the network transitions to the basic daily per-device quota offered by IoT-picosat companies, which is around 50 packets/day (5000 bytes) [21], 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 PicoNet's simplified concept, begins to perform well with higher traffic. However, PicoNet 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. 12b plots the per-satellite downlink throughput of PicoNet's scheduler and the corresponding baselines. In this experiment, we generate (infinite) data directly on the satellite to understand how our downlink scheduler performs independent of our uplink. The mean (90th percentile) throughput for PicoNet is 538,000 (969,800) bytes while the mean (90th percentile) throughput for L2D2 is 7,000 (181,600) bytes. PicoNet's downlink schedule achieves greater throughput than broadband satellite scheduling such as L2D2 because it accounts for interference that stems from satellites transmitting to ground stations in an overlapping region. Moreover, PicoNet's one-to-many downlink scheduling algorithm helps schedule which ground stations need to be actively listening, allowing the network to have greater gains in throughput by leveraging simultaneous downlink through multiple ground stations.

End-to-End Throughput: Fig. 11b reports the full end to end network throughput in bps across the varying traffic generation patterns applying both uplink and downlink network stack settings. PicoNet outperforms network stack 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. 11a. Among all approaches PicoNet maintains consistently high performance across a variety of traffic generation schemes. As a result, PicoNet maximizes data from IoT devices to the cloud by providing a scalable solution that tackles the challenges of throughput in IoT picosat networks from an architectural level. This result also implies that PicoNet can support a larger scale of device deployments, compared to other baselines.

8.2 Uplink Fairness

Given that PicoNet forces devices in overlapping regions to transmit more conservatively, we evaluate the uplink fairness of our random access scheme. Fig. 13 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 PicoNet penalizes devices in the overlap, fairness is possible due to the continuously moving satellite footprints. Moreover, PicoNet's flow control mechanism and α tuning based fairness strategy further prevents

starvation of devices not only in overlapping regions but also in high-overlap-prone geographic regions towards the North and South poles.

8.3 Downlink Robustness

PicoNet's downlink schedule algorithm is designed to generate a schedule that is also robust to packet loss, attempting to ensure that data is downlinked to at least $K \geq 2$ ground stations. Fig. 12c 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 like L2D2 which only transmit using a point-to-point link schedule, PicoNet significantly improves downlink robustness by leveraging the advantage of the large coverage area of IoT picosat footprints.

8.4 Latency

In Fig. 12a, we plot a boxplot of latency (data reception time at a ground station to data generation time) across different baselines for varying traffic patterns. We observe that although PicoNet has the highest overall latency reported, it is mainly because of the survivor bias. Latency calculations take into account the packets that successfully reach the ground station. Since PicoNet achieves higher throughput, more packets successfully make their way to the ground station throughout the simulation duration, contributing to the higher aggregate latency figure. Conversely, lower throughput results in fewer packets reaching the ground station in the case of other methods, ultimately leading to lower aggregate latency numbers.

9 Related Work

Satellite Networking: Previous work in the satellite networking domain has mostly focused on satellite ground station architectures [23, 41, 62], RF link prediction [16, 26, 66], and orbital and power modeling [19, 27, 31, 50], as well as analyzing large-scale constellations [43, 44, 57]. However, the majority of this prior work focuses on broadband satellites, which have 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 [20, 30, 32, 36, 51, 65]. Additionally, IoT satellite networking needs to handle uplink data to the satellite, which is not relevant for certain contexts of broadband satellites, such as Earth observation satellites [60, 62]. Some effort has been made to explore the effects of LEO IoT satellites on shared spectrum with terrestrial communication systems [47, 64] and to examine the effects of data communication under high satellite path loss and Doppler shift [22, 32]. However, PicoNet distinguishes itself by providing a novel architectural solution that will better enable IoT picosat networks at scale.

Uplink Protocols: Due to current hardware constraints on both IoT devices and picosats, sophisticated network medium access control techniques that require the channel cannot be implemented. Therefore, data uplink to picosats can only be done using random access MAC protocols [14]. 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 [63]. In contrast to terrestrial networks, where ALOHA with overlapping cells has been explored to a much more limited extent [49], the satellite network case is largely different because the cells are constantly moving. PicoNet's unique uplink mechanism tunes the device uplink probability based on the movement and overlap of satellite footprints in the network and varies the transmission probability based on the orbital patterns of the satellite topology. By reducing collisions that stem from inter-satellite interference, significant gains in uplink throughput at the architectural level can be achieved. PicoNet's approach also provides a MAC layer solution that can be easily augmented and integrated with various popular PHY layer approaches. For example, recent work [14, 46, 58] 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 [17, 34, 56]. 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 [28, 62]. In contrast, PicoNet's scheduling approach is tailored for picosats and addresses interference in the satellite network resulting from omnidirectional antennas used for downlink.

10 Conclusion

We present PicoNet-a network stack for a new class of IoT networks served by picosatellites. Our work leverages satellite-specific insight and involves cross-stack innovations – new medium access protocol, a new network scheduling algorithm, and a flow control scheme. We evaluate PicoNet using a combination of real-world measurements and trace-driven simulations. We hope our measurements, insights, and simulation frameworks will enable other researchers in the community to identify and solve new challenges in this space.

References

[1] Spacex buys out satellite iot startup swarm technologies. https://www.satellitetoday.com/business/2021/08/09/spacex-buys-out-satellite-iot-startup-swarm-technologies/.

[2] Tinygs – the open source global satellite network.

[3] Benefits of direct-to-satellite iot and use cases to avoid, Feb 2021.

[4] Celestrak. https://celestrak.org/, 2023.

[5] Echostar. https://www.echostar.com/, 2023.

[6] Fossa systems. https://fossa.systems/, 2023.

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

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

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

[10] Starlink. https://www.starlink.com/, 2023.

[11] Swarm. https://swarm.space/, 2023.

[12] Sx1302cssxxxgw1. https://www.semtech.com/products/wireless-rf/lora-core/sx1302cssxxxgw1, 2023.

[13] LoRaWAN Alliance. Lorawan 1.1 specification, Oct 2017.

[14] Guido Alvarez, Juan A. Fraire, Khaled Abdelfadeel Hassan, Sandra Cespedes, and Dirk Pesch. Uplink transmission policies for lora-based direct-to-satellite iot. *IEEE access: practical innovations, open solutions*, 10:72687–72701, 2022.

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

[16] Radu Arsinte. Effective methods to analyze satellite link quality using the built-in features of the dvb-s card. 01 2006.

[17] Jeremy Castaing. Scheduling Downloads for Multi-Satellite, Multi-Ground Station Missions. Small Satellite Conference, 2014.

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

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

[20] Houcine Chougrani, Steven Kisseleff, Wallace A Martins, and Symeon Chatzinotas. Nb-iot random access for nonterrestrial networks: Preamble detection and uplink synchronization. *IEEE Internet of Things Journal*, 9(16):14913–14927, 2021.

[21] Devin Coldewey. Swarm prices out its orbital iot network's hardware and services, Sep 2020.

[22] Matteo Conti, Stefano Andrenacci, Nicola Maturo, Symeon Chatzinotas, and Alessandro Vanelli-Coralli. Doppler impact analysis for nb-iot and satellite systems integration. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2020.

[23] Iñigo del Portillo, Bruce Cameron, and Edward Crawley. Ground segment architectures for large leo constellations with feeder links in ehf-bands. In *2018 IEEE Aerospace Conference*, pages 1–14, 2018.

[24] Kiruthika Devaraj, Ryan Kingsbury, Matt Ligon, Joseph Breu, Vivek Vittaldev, Bryan Klofas, Patrick Yeon, and Kyle Colton. Dove High Speed Downlink System. Small Satellite Conference, 2017.

[25] Kiruthika Devaraj, Matt Ligon, Eric Blossom, Joseph Breu, Bryan Klofas, Kyle Colton, and Ryan Kingsbury. Planet High Speed Radio: Crossing Gbps from a 3U Cubesat. Small Satellite Conference, 2019.

[26] Menachem Manis Domb Alon and Guy Leshem. Satellite to ground station, attenuation prediction for 2.4–72 ghz using ltsm, an artificial recurrent neural network technology. *Electronics*, 11(4), 2022.

[27] Tom Etchells and Lucy Berthoud. Developing a power modelling tool for cubesats. August 2019. Annual AIAA/USU Conference on Small Satellites, SSC 2019 ; Conference date: 03-08-2019 Through 08-08-2019.

[28] Huilong Fan, Zhan Yang, Shimin Wu, Xi Zhang, Jun Long, and Limin Liu. An efficient satellite resource cooperative scheduling method on spatial information networks, Dec 2021.

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

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

[31] Roberto Flores, Burhani Makame Burhani, and Elena Fantino. A method for accurate and efficient propagation of satellite orbits: A case study for a molniya orbit. *Alexandria Engineering Journal*, 60(2):2661–2676, Apr 2021.

[32] Juan A Fraire, Sandra Céspedes, and Nicola Accettura. 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*, pages 241–258. Springer, 2019.

[33] Juan A. Fraire, Santiago Henn, Fabio Dovis, Roberto Garello, and Giorgio Taricco. Sparse satellite constellation design for lora-based direct-to-satellite internet of things. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pages 1–6, 2020.

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

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

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

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

[38] International Telecommunications Union. ITU P.838: Specific attenuation model for rain for use in prediction methods. Technical report, 2019.

[39] International Telecommunications Union. ITU P.839 : Rain height model for prediction methods . Technical report, 2019.

[40] International Telecommunications Union. ITU P.840: Attenuation due to clouds and fog. Technical report, 2019.

[41] William Ivancic. Architecture study of space-based satellite networks for nasa missions. *NASA/TM*, 2003.

[42] V. Jacobson. Congestion avoidance and control. In *Symposium Proceedings on Communications Architectures and Protocols*, SIGCOMM '88, page 314–329, New York, NY, USA, 1988. Association for Computing Machinery.

[43] Pauline Jakob, Seiichi Shimizu, Shoji Yoshikawa, and Koki Ho. Optimal satellite constellation spare strategy using multi-echelon inventory control. *Journal of spacecraft and rockets*, page 1–13, May 2019.

[44] Sijing Ji, Di Zhou, Min Sheng, Dong Chen, Liang Liu, and Zhu Han. Mega satellite constellations analysis regarding handover: Can constellation scale continue growing? In *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, pages 4697–4702, 2022.

[45] Herber Kramer. Lacuna constellation, 2020.

[46] Alireza Maleki, Ha H. Nguyen, Ebrahim Bedeer, and Robert Barton. D2d-aided lorawan lr-fhss in direct-to-satellite iot networks. *arXiv*, 2022.

[47] Mario Marchese, Aya Moheddine, and Fabio Patrone. Iot and uav integration in 5g hybrid terrestrial-satellite networks. *Sensors*, 19(17):3704, 2019.

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

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

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

[51] Zhicheng Qu, Gengxin Zhang, Haotong Cao, and Jidong Xie. Leo satellite constellation for internet of things. *IEEE access*, 5:18391–18401, 2017.

[52] FCC Report. Company profile: Hiber inc. https://fcc.report/company/Hiber-Inc, 2022.

[53] FCC Report. Company profile: Swarm technologies inc. https://fcc.report/IBFS/Company/Swarm-Technologies-Inc, 2022.

[54] FCC Report. Company profile: Myriota pty ltd. https://fcc.report/company/Myriota-Pty-Ltd, 2023.

[55] Microsoft Research. 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. https://github.com/microsoft/CosmicBeats-Simulator, 2023.

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

[57] Tianyu Sun, Min Hu, and Chaoming Yun. Low-orbit large-scale communication satellite constellation configuration performance assessment. *International Journal of Aerospace Engineering*, 2022:1–8, Mar 2022.

[58] Muhammad Asad Ullah, Konstantin Mikhaylov, and Hirley Alves. Analysis and simulation of lorawan lr-fhss for direct-to-satellite scenario. *IEEE Wireless Communications Letters*, 11(3):548–552, 2022.

[59] Deepak Vasisht and Ranveer Chandra. A distributed and hybrid ground station network for low earth orbit satellites. ACM HotNets, 2020.

[60] Deepak Vasisht and Ranveer Chandra. A distributed and hybrid ground station network for low earth orbit satellites. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, pages 190–196, 2020.

[61] Deepak Vasisht, Zerina Kapetanovic, Jong-ho Won, Xinxin Jin, Ranveer Chandra, Ashish Kapoor, Sudipta N. Sinha, Madhusudhan Sudarshan, and Sean Stratman. Farmbeats: An iot platform for data-driven agriculture. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation*, NSDI'17, page 515–528, USA, 2017. USENIX Association.

[62] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. L2d2: Low latency distributed downlink for leo satellites. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, SIGCOMM '21, page 151–164, New York, NY, USA, 2021. Association for Computing Machinery.

[63] Kai Vogelgesang, Juan A. Fraire, and Holger Hermanns. Uplink transmission probability functions for lora-based direct-to-satellite iot: A case study. In *2021 IEEE Global Communications Conference (GLOBECOM)*, page 01–06. IEEE, Dec 2021.

[64] Peng Wang, Jiaxin Zhang, Xing Zhang, Zhi Yan, Barry G Evans, and Wenbo Wang. Convergence of satellite and terrestrial networks: A comprehensive survey. *IEEE access*, 8:5550–5588, 2019.

[65] Alexander M Zadorozhny, Alexander A Doroshkin, Vasily N Gorev, Alexander V Melkov, Anton A Mitrokhin, Vitaliy Yu Prokopyev, and Yuri M Prokopyev. First flight-testing of lora modulation in satellite radio communications in low-earth orbit. *IEEE Access*, 10:100006–100023, 2022.

[66] Bircan Çalışır and Ayhan Akbal. A new rf satellite link analyzing and antenna effect on satellite communication. *Tehnički glasnik*, 16:550–556, 09 2022.

A Downlink Scheduling Algorithms

In this appendix, we present the algorithms in our centralized downlink scheduler. Algorithm 1 shows the overall scheduling flow of PicoNet. As discussed in Section 5, the scheduling algorithm is modeled as a Maximum Weighted Independent Set problem on the graph constructed at time step t. We have discussed how the weights of the vertices get initialized and updated in Section 5, and this is shown in Algorithm 1 as well.

Algorithm 1 PicoNet Downlink Scheduling

Input:
$G_t(V,E) \leftarrow$ Graph Constructed for time step t
$r_i(t)$, $\forall\, v_i \in V \leftarrow r_i(t)$ is RSSI for link in v_i at time t
$T \leftarrow$ Number of time slots
Output:
Schedule $F = \{F_1, F_2, \ldots, F_T\}$, where $F_i \subset V, \forall i$
Algorithm:
$LTS_i(t=1) = 1 \;\; \forall v_i \in Vt$
for $t \in \{1, \ldots, T\}$ **do**
  Vertex Weights W_t: $w_i(t) = r_i(t) \times LTS_i(t)$, $\forall\, v_i \in V$
  $F_t \leftarrow$ WeightedMaxIndependentSet(G_t, W_t)
  $F_t \leftarrow$ ReliabilityGuarantee(F_t)
  for $v_i \in V$ **do**
    if Satellite in v_i transmitted to anyone at t **then**
      $LTS_i(t+1) = 1$
    else
      $LTS_i(t+1) = LTS_i(t+1) + 1$

A.1 PicoNet's Reliability Guarantee Algorithm

As mentioned in Section 5, the Maximum Weighted Independent Set formulation for PicoNet's scheduler is a best-effort algorithm. To ensure reliability, we want to assign each satellite $K \geq 2$ receive ground stations. Towards this end, Alg. 2 shows the greedy heuristic implemented by PicoNet to ensure reliability. The input to Alg. 2 is the schedule F_t computed by the MWIS algorithm at time step t. The output of Alg. 2 is a tweaked version of the schedule, that is, $F_t^{'}$, where the goal is to remove some scheduled links in F_t, and in its place, add new links such that every satellite can be assigned at least 2 ground stations in the final schedule $F_t^{'}$.

Let $S = \{s_1, \ldots, s_n\}$ be the subset of satellites that were scheduled to transmit at time slot t by the MWIS algorithm. Also, let the set of vertices chosen by the MWIS algorithm corresponding to the schedule in F_t by $V_{MW} = \{v_1, \ldots, v_q\}$, where $V_{WM} \subset V$. Now divide S into two mutually exclusive subsets. First, $S_I \subset S$, which consists of set of satellites that were assigned to only 1 ground station in original schedule F_t. Second, $S_M = S - S_I$, which consists of set of satellites that were assigned greater than or equal to 2 ground stations in the original schedule F_t.

Now for reliability, we require that for the satellites in S_I, we add at least one more of its links to the schedule. Concretely, let's say for a satellite s_i, the set of vertices in the graph that correspond to satellite s_i is V_{s_i}. So for each satellite $s_i \in S_I$, we want to add at least one more currently unscheduled vertex from the set V_{s_i}. However, note that the vertices scheduled in F_t form a *maximal* independent set, as shown in [29]. This means that no new currently unscheduled vertex can be added to the schedule, *without removing some of the currently scheduled vertices in* V_{MW}.

One approach to achieve reliability is that, there will be some satellites in S_M that are scheduled to more than 3 ground stations. So we could potentially divert 1 ground station from serving those satellites in S_M to serving the satellites in S_I. However, if one observes the graph construction and the edges carefully, it becomes evident that this is not possible. This is because when a satellite $s_m \in S_M$ has, say 4 ground stations assigned to it, and if we were to try to divert one of the ground stations to a satellite $s_i \in S_I$, this diverted ground station is still receiving signal from s_m since the s_m picosat transmissions are omnidirectional. Hence the diverted ground station will still be unviable to receive from s_i due to interference from s_m satellite.

Hence, our approach to achieve reliability is the following. We will try to remove some of the satellites scheduled in S_I, such that we can ensure that the remaining satellites in S_I can get their guaranteed 2 receive ground stations. We leverage a greedy heuristic to do this as shown in Alg. 2.

Algorithm 2 PicoNet's RELIABILITYGUARANTEE

Input:
$G_t(V,E) \leftarrow$ Graph Constructed for time step t
$F_t \leftarrow$ Scheduled vertices as computed by MWIS without reliability
$S = \{s_1, \ldots, s_n\} \leftarrow$ Set of satellites scheduled in F_t
$V_{s_i} \leftarrow$ Set of vertices corresponding to satellite s_i in $G_t(V,E)$
Output:
$F_t^{'} \leftarrow$ Tweaked Schedule with reliability guarantee
Algorithm:
$S_I \leftarrow$ Satellites that were assigned only 1 GS, $S_I \subset S$
$S_M \leftarrow S - S_I \leftarrow$ Satellites assigned multiple receive GS
for $p \in S_I$ **do**
    $V_M =$ All vertices corresponding to satellites in S_M
    $V_{M\_F_t} = V_M \cap F_t \leftarrow$ Vertices corresponding to sats in S_M in schedule F_t
    $N[V_{M\_F_t}] =$ Vertices in $V_{M\_F_t}$ along with all their neighbors in set V_{S_I}
    $G_t^{ind} = G_t(V_S - N[V_{M\_F_t}])$ where $G(V^{'})$ is induced subgraph on vertex set $V^{'}$
    if there exists vertex u in G_t^{ind} s.t. $u \in V_p$ **then**
        if u does not have edge to vertex $V_p \cap F_t$ **then**
            $S_M \leftarrow S_M \cup p$
            $F_t \leftarrow F_t \cup u$
    else $F_t \leftarrow F_t - (V_p \cap F_t)$
$F_t^{'} = F_t$