Network Size Estimation for Direct-to-Satellite IoT

Pablo Ilabaca, Samuel Montejo-Sánchez, Juan Fraire, Richard Demo Souza, Sandra Céspedes

Network Size Estimation for Direct-to-Satellite IoT

Pablo Ilabaca, Samuel Montejo-Sánchez, *Senior Member, IEEE*, Juan A. Fraire, *Senior Member, IEEE*, Richard Demo Souza, *Senior Member, IEEE*, and Sandra Céspedes, *Senior Member, IEEE*

***Abstract*—The worldwide adoption of the Internet of things (IoT) depends on the massive deployment of sensor nodes and timely data collection. However, installing the required ground infrastructure in remote or inaccessible areas can be economically unattractive or unfeasible. Cost-effective nanosatellites deployed in low Earth orbits (LEO) are emerging as an alternative solution: on-board IoT gateways provide access to remote IoT devices, according to direct-to-satellite IoT (DtS-IoT) architectures. One of the main challenges of DtS-IoT is to devise communication protocols that scale to thousands of highly constrained devices served by likewise constrained orbiting gateways. In this paper, we tackle this issue by first estimating the (varying) size of the device set underneath the (mobile) nanosatellite footprint. Then, we demonstrate applicability of the estimation when used to intelligently throttle DtS-IoT access protocols. Since recent works have shown that MAC protocols improve the throughput and energy efficiency of a DtS-IoT network when a network size estimation is available, we present here a novel and computationally-efficient network size estimator in DtS-IoT: our optimistic collision information (OCI) based estimator. We evaluate OCI's effectiveness with extensive simulations of DtS-IoT scenarios. Results show that when using network size estimations, the scalability of a frame slotted Aloha-based DtS-IoT network is boosted 8-fold, serving up to 4×10^3 devices, without energy efficiency penalties. We also show the effectiveness of the OCI mechanism given realistic detection ratios and demonstrate its low computational cost implementation, making it a strong candidate for network estimation in DtS-IoT.**

***Index Terms*—Direct-to-Satellite, Internet of Things, Medium Access Control, Network Size Estimation, Satellite Communications.**

Manuscript received July 18, 2022; revised September 26, 2022; accepted November 11, 2022

P. Ilabaca is with the Department of Computer Science and Software Engineering, Concordia University, Montreal, QC, Canada, pablo.ilabaca@mail.concordia.ca.

S. Montejo-Sánchez is with Programa Institucional de Fomento a la Investigación, Desarrollo e Innovación, Universidad Tecnológica Metropolitana, Santiago, Chile, smontejo@utem.cl.

J. A. Fraire is with Univ Lyon, Inria, INSA Lyon, CITI, F-69621 Villeurbanne, France, and the Argentinian research council (CONICET), Córdoba, Argentina. He is also with the Saarland University, Germany.

R. D. Souza is with Federal University of Santa Catarina (UFSC), Florianópolis, Brazil, richard.demo@ufsc.br.

S. Céspedes is with the Department of Computer Science & Software Engineering, Concordia University, Montreal, QC, Canada. Correspondence: sandra.cespedes@concordia.ca

This work has been supported by the Project ANID Fondecyt Regular No. 1201893, the ANID Basal Project FB0008, the ANID FONDECYT Iniciación No. 11200659, FONDEQUIP-EQM180180, Project STARS STICAMSUD 21-STIC-12 Folio STIC2020003, and by CNPq Brazil (402378/2021-0, 305021/2021-4). This work started out while P. Ilabaca and S. Céspedes were with the Department of Electrical Engineering and with the NIC Chile Research Labs, Universidad de Chile, Santiago, Chile.

I. Introduction

The Internet of things (IoT) market is thriving. By 2025, the annual revenue of IoT will reach 11 trillion dollars obtained from more than 30 billion connected devices [1], [2]. Applications enabled by IoT networks include monitoring the environment, meteorological or environmental phenomena, animals or plants as well as tracking and positioning services. Applications for logistics and location are also available among many others. However, many applications require the IoT devices to be deployed in remote areas with minimal or no human presence. In cases where connectivity via terrestrial techniques might be unappealing or not possible, satellite technology emerges as the appropriate approach. Indeed, satellite IoT has recently been recognized as instrumental to ensure data collection over remote regions of the planet, enabling a truly global IoT solution [3].

The ad-hoc nature of space missions driving the deployment of satellite infrastructure typically demands a full re-engineering of the complete space and ground segments—a factor that has kept most countries out of satellite technology development. However, the CubeSat standard provided a common mechanical and electrical framework for the development of nanosatellites [4]. Since its introduction, the cost of these spacecraft has dropped dramatically, allowing new space actors to access a more "democratic" space [5]. A 1U (1 unit) CubeSat is characterized by a weight of 1 kilogram and a cubic shape of $10\times10\times10$ centimeters. CubeSat platforms of 3U, 6U, and 12U are also popular in the industry. These homogeneous characteristics reduce manufacturing, procurement, and launching costs but, at the same time, impose limiting conditions such as lower processing, energy, and storage capabilities. Thus, CubeSat-based missions tend to be heavily constrained in terms of resources. The research presented in the remainder of this work is inspired by CubeSat platforms devoted to providing direct-to-satellite IoT (DtS-IoT) services.

In the DtS-IoT architecture, actuators and sensors on the ground communicate directly with the passing-by low Earth orbit (LEO) satellite leveraging technologies such as LoRa, long range-frequency hopping spread spectrum (LR-FHSS), narrowband-IoT (NB-IoT) and the fifth generation (5G) mobile communication [6]–[8]. Since a ground-based IoT gateway is not required, this architecture is particularly appealing for efficiently servicing remote areas such as oceans, mountains, or inaccessible regions, as well as disaster zones [9].

The main challenge of DtS-IoT is to ensure good performance in the uplink shared channel as the number of nodes in coverage may be significantly higher than in traditional terrestrial networks. Indeed, link saturation is already achieved with

a few thousand nodes even when state-of-the-art wide-area IoT technologies are implemented [6]. Beyond this scalability point, packet collisions and energy inefficiencies undermine any viable DtS-IoT application. In this context, media access control (MAC) protocols for DtS-IoT are of paramount importance in massive deployments [10]. Furthermore, the network size under the footprint of the nanosatellite is a priori unknown, but an accurate network size estimation can be exploited by MAC protocols to optimize the shared uplink channel resources [11], [12].

Indeed, estimating a node population's size is a major factor for achieving scalability and energy efficiency on certain access control protocols [12], [13]. Additionally, knowing the number of devices to be served by a CubeSat guarantees the quality of service for their applications. Depending on how overwhelmed a CubeSat will be during operation, it is possible to justify whether to implement more satellites into the constellation or lower service capacity. Nonetheless, to the best of our knowledge, efficient network size estimation mechanisms operating in constrained IoT gateways deployed in continuously-moving LEO CubeSats have not yet been addressed in the literature. The problem of an unknown and massive population of nodes trying to communicate with nanosatellites in applications that satellite technology supports is recent. Consequently, the importance in the democratization of low-cost global connectivity and the lack of studies in the literature motivate the proposal of network size estimation techniques for this particular scenario.

In the past, several network size estimation mechanisms have been proposed for terrestrial wireless sensor networks (WSN) [14]. These mechanisms assume static base stations and not necessarily energy-constrained devices, making their direct application difficult in the DtS-IoT context. The work in [15] is aimed at dealing with time-evolving topologies, but the method is based on persistent node connectivity, a feature not present in DtS-IoT. Conversely, in this work we explore the design of network size estimation mechanisms specifically suited for LEO CubeSats providing DtS-IoT services to thousands of IoT nodes. Our preliminary results of the applicability of the estimator proposed in [14] to DtS-IoT scenarios are presented and analyzed in [16].

The contribution of this work is twofold. First, we propose the optimistic collision information (OCI)based estimator: a network size estimation mechanism for DtS-IoT networks. OCI is a novel scheme with reduced computational complexity, a high adaptability to different DtS-IoT scenarios in terms of detection ratios, and an increased numerical stability compared to existing estimators. We demonstrate that OCI achieves high scalability and a low estimation error. Second, we assess the practical value of leveraging the estimations from OCI. To this end, we present extensive simulation results of a DtS-IoT network governed by a frame slotted Aloha (FSA) protocol, where device transmissions are throttled by OCI estimations. We show that when OCI is employed, the network achieves a throughput close or equal to the theoretical maximum in DtS-IoT regions with more than 4×10^3 devices; furthermore, we provide theoretical and empirical complexity analyses of the network size estimation mechanisms, using a typical CubeSat architecture. The proposed estimation method runs in only 24.3% of the time required by its most relevant competitor.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III provides the description of the scenario of study and the design of the proposed network size estimator. Section IV analyzes the performance of the OCI mechanism and presents the results of exploiting size estimations in a DtS-IoT scenario. Section V delves into practical considerations while Section VII concludes the paper. Table I offers the definition of the different acronyms and Table II provides a list of variables used in this paper.

TABLE I: List of Acronyms

Acronym	Definition
5G	Fifth Generation mobile communication
BFS	Breadth-First Search
CR	Capture-Recapture
DtS-IoT	Direct-to-Satellite IoT
eNB	evolved Node B
FSA	Frame Slotted Aloha
GEO	Geosynchronous Equatorial Orbit
GPS	Global Positioning System
IoT	Internet Of Things
LEO	Low Earth Orbit
LR-FHSS	Long Range-Frequency Hopping Spread Spectrum
LTE	Long Term Evolution
MAC	Medium Acces control
MEO	Medium Earth Orbit
MTC	Machine-Type Communication
NB-IoT	NarrowBand-IoT
OCI	Optimistic Collision Information based estimator
P2P	Peer-to-Peer
PDF	Probability Distribution Function
RFID	Radio Frecuency Identification
RMSE	Root-Mean-Square Error
ROM	Read Only Memory
S&F	Store and Forward
SINR	signal-to-interference-plus-noise ratio
UE	User Equipment
WSN	Wireless Sensor Network

TABLE II: List of Variables

Variable	Definition
Φ	Array of the naive estimates ϕ
ϕ	naive estimate $\phi = s + 2c$
K	Maximum number of FSA frames considered
M	Number of iterations considered for the estimation phase
N	Number of nodes deployed in the cluster
$\vec{N}$	Array of the node populations n_k, for $k = 1, 2, ..., K$
$\vec{P}$	Array of size q formed by the polynomial coefficients
T	Arbitrary time
X	Vandermonde matrix of dimensions $j \times q + 1$ formed by the elements of Φ
c	Number of collided slots in a FSA frame
k	Frame related to a given CubeSat coverage zone $0 < k \leq K$
m	Iteration number, $0 < m \leq M$
n_k	Number of nodes present at a given CubeSat coverage zone k, $0 < n \leq N$
p	Array of polynomial coefficients in decreasing degree order
q	Degree of the polynomial
s	Number of successful slots in a FSA frame
u	Number of idle slots in a FSA frame
w	Frame length as number of slots
y	Estimate correction of the contender nodes by polynomial evaluation
$\overline{y_{(k,m)}}$	Average of m estimates of the contender nodes on a frame k

II. Background

In this section, we first briefly discuss orbit characteristics for deploying space technology, and then provide a review of previous direct and indirect network size estimation mechanisms proposed in the literature.

A. Space Technology and Orbits Characteristics

Satellites can be deployed in different orbits with diverse characteristics that condition the exploitation of the hosted communication payload. The three main types of satellite orbits are the following:

- Geosynchronous Equatorial Orbit (GEO): This orbit has a rotation period equal to that of the Earth. A GEO satellite appears motionless to an observer on the ground. GEO satellites have an altitude of 35,786 km, which gives them a coverage of approximately 30% of the Earth's surface. This orbit offers services such as weather data and broadcast TV [17]. However, the long distance to the Earth surface results in high latency, on the order of 120 ms for the best case scenario [10]. Furthermore, high transmission power and large antenna gains are required [17]. Also, the process of deploying a GEO satellite is the most expensive among the three orbits.
- Medium Earth Orbit (MEO): With an altitude ranging between 2,000 km and 3,586 km, MEO satellites experience a lower latency than GEO satellites and provide navigation services such as the Global Positioning System (GPS). The launching cost of MEO satellites is midway between the cost of the systems located in the other two orbits.
- Low Earth Orbit (LEO): Its altitude ranges between 300 km and 2,000 km, offering the lowest latency, on the order of tens of milliseconds [10]. Due to the smaller distance to Earth, LEO satellites require lower transmission power and launching costs. Although smaller and cheaper, LEO satellites move at $\sim$7 km/s, which may result in passes of $<$10 min over a given spot on the surface. Thus, LEO constellations are needed to provide persistent coverage. LEO satellites will play a key role in providing ubiquitous and low-cost coverage for IoT devices in the near future [8].

One of the main issues in estimating the size of a network as seen from a satellite passing by is that the ground track (the projection of the satellite trajectory over the Earth surface) does not necessarily repeat. In other words, the satellite might observe a given set of nodes in one pass, and a subset of those (plus possible others) in subsequent passes. While such a condition challenges any network size estimation techniques, orbits that periodically cover the same ground track are often used in practical space applications (*i.e.*, multiple observations of the same area). These orbits are known as Earth-repeat orbits [18], [19] and are described in detail in Section V.

B. Direct Network Size Estimation Approaches

In [15], the authors present two algorithms for estimating the size of controller/agent and *ad hoc* networks. These algorithms are adaptations of the existing *random tour* and *gossip-based aggregation* methods to fit controller/agent architectures and to be compatible with random changes in the topology. The *random tour* method is based on the transfer of a *token*, registering each new visited node to make the estimation. The *gossip-based aggregation* method is based on communication between adjacent nodes. The network is divided into *clusters*, with a controller in each cluster repeating a process in which it averages a value with the agent nodes, sends the updated value to all agents, and eventually obtains a converged value equal to $\frac{1}{N}$, where N is the number of nodes. The two algorithms proposed in [15] require communication among nodes, which is most likely not available in the DtS-IoT architecture.

In [20], the authors developed a network size estimation mechanism focused on Peer-to-Peer (P2P) networks. This mechanism "marks" neighboring nodes that are discovered when notified by other nodes of the network in a Capture-Recapture (CR) logic. This approach is based on a Breadth-First Search (BFS) rooted in the node that wants to discover the size of the network. The sum of new neighbors discovered at each iteration of the BFS is fitted to a logarithmic function whose coefficients are empirically set. Then, this function is solved using Lambert's W function [21] to estimate the network size. An error of at most 10% is reached, allowing the BFS to iterate only up to the third level. Later, the algorithm of [20] is modified to obtain a different logarithmic function, achieving a better performance in terms of accuracy versus the number of operations [22]. The improvement reaches an estimation error of at most 5% while visiting less than 10% of the network. Once again, network exploration is based on direct communication among peers, making the approaches non-feasible for DtS-IoT scenarios.

In [23], a survey of the CR approach for population estimation in computer networks is offered. In general, CR solutions show low precision and the estimation process may be unsuitable for cases when the network size changes are fast, which is expected from the DtS-IoT scenario. A more reliable and faster estimate can be obtained by storing the identity of the elements (sensor nodes) caught in the different samples, but paying the price of higher storage and computational requirements to handle the sampling history, which could be difficult for a CubeSat in terms of memory. In [24], the corner stone of the CR methods, the Lincoln-Petersen Index, is employed to estimate the size of a Wireless Sensor Network (WSN). In [25], this method is used to estimate the number of wireless access points in a closed study area, as well as the Jackknife method. However, both of these methods work well on a closed population, *i.e.* no element is entering or leaving the population during the sampling and estimate operations, which differs from the dynamic characteristic of DtS-IoT networks. In [23], CR methods for open populations are also described, such as the Jolly-Seber model. This model measures several quantities such as the number of individuals caught in each sample, the number of individuals already marked or caught for the first time in each sample, the number of marked individuals released after a sample, among others. These quantities are then used to estimate the number of individuals in the population up to the last sample, and other

parameters. However, this model would require a meticulous measurement and control of the number of nodes present in an area in the pass of a nanosatellite. Characteristics, such as the non-homogeneity of the conditions of each node, the need to store the node identities, and the fact that an orbit can take several cycles to repeat itself, make this method difficult (if at all possible) to implement in the scenario studied.

C. Indirect Network Size Estimation Approaches

Other mechanisms approach the estimation problem focusing on the observed channel load or the number of collisions (and other events) occurring during contention periods [14], [26]–[28]. To address the channel load estimation problem in a Long Term Evolution (LTE) network, the authors in [26]–[28] provide a Probability Distribution Function (PDF) of the result of the access requests to ease the management at the evolved Node B (eNB) when the simultaneous activation of a large number of Machine-Type Communications (MTC) – also called User Equipments (UEs) – occurs.

In [26], the authors derive an explicit expression to compute the PDF of the number of successful transmissions. Later, the authors in [27] compute the PDF of the number of successful preamble transmissions using combinatorial analysis. Both contributions show numerical stability issues around 200 devices, the point at which they stop generating valid results. In [28], the authors calculate the joint PDF of the successful and collided access requests within a random access opportunity, increasing the effectiveness up to 300 devices. However, the method is applied to a scenario that differs from the DtS-IoT scenario since it requires terrestrial infrastructure (*i.e.*, the eNB) capable of carrying the computations of PDF and the estimations.

In [29], the authors present an estimation technique to detect the number of active users in irregular repetition slotted ALOHA (IRSA) access protocols based on successive interference cancellation (SIC) at the receiver's side. Each device transmits multiple replicas of a packet in different slots, and the interference contribution of already identified devices is canceled, permitting the identification of new devices. But, this technique requires high processing and memory capacity at the receiver, which prevents its applicability in CubeSats [10].

The authors in [30] present the average run-based tag (ART) estimation scheme for FSA wireless networks in radio frequency identification (RFID). Collisions due to simultaneous tag responses may occur in the same way as in WSNs. ART is based on the average run length of $1's$ in the bit string received at the reader (*i.e.*, receiver), considering 1 as the bit value of one or multiple transmissions received in a slot and 0 as the value of no transmission. These authors also provide a set of optimal parameters that save some calculations for wide ranges of node populations, but complex and costly numerical solutions are still needed at the end of each frame during the estimation stage. This makes ART inadequate for a CubeSat implementation.

Another scheme that uses the scalable minimum mean square error (sMMSE) is proposed as an estimation method for the RFID tag population and it actually outperforms the ART method in both the achieved estimation error and the estimation time [31]. This approach bases its operation on the iterative adaptation of the FSA frame length to the node population to be estimated. Once the frame size is adapted, the estimation is calculated by numerically solving a minimum square error equation. Given the superior performance of sMMSE over ART and the possibility to adapt it to the DtS-IoT scenario studied, this scheme is considered one of the benchmarks of our proposal.

The other benchmark selected for its performance and simplicity is the collision set size estimator proposed by Zanella for the FSA protocol in RFID systems [14]. A collision set is the group of nodes contending for a time slot in a given contention window of FSA. A collision occurs if two or more nodes in the set transmit concurrently. The receiver can perceive these collisions, keeping track of the number c of collisions that have occurred, as well as the number of successful transmissions s and idle slots i. These three numbers together compose a time slot observation. Then, Zanella applies a maximum likelihood estimator to these values, which returns the number n of transmitting nodes that maximizes the conditional probability of having that observation at the end of the frame. The number of transmissions in each time slot is considered to be an independent Poisson random variable.

Table III presents a summary with a qualitative assessment of the main estimation methods surveyed here and their applicability to DtS-IoT scenarios. In addition, Section III provides a detailed description of the selected benchmark schemes and the adaptations required for their application in the DtS-IoT scenario.

III. Network Size Estimation Mechanism for DtS-IoT

In this section, we describe the proposed network estimation mechanism employed in a DtS-IoT network, considering the capabilities and limitations of an IoT gateway deployed on a CubeSat. We first detail the study scenario and assumptions and then describe OCI, a simple yet accurate network estimation mechanism.

A. System Model and Assumptions

The network scenario under study follows a DtS IoT architecture, with a CubeSat nanosatellite acting as the IoT gateway. The CubeSat, whose coverage footprint passes over a cluster of terrestrial IoT nodes, travels around the earth using a LEO orbit. The orbit in which the satellite is deployed has a height and speed of approximately 500 km and 7.5 km/s, respectively [32]. The satellite completes a turn around the world in 90 minutes.

For network size estimation purposes, the communication between the CubeSat and IoT nodes follows an FSA logic. The FSA communication windows are illustrated in Fig. 1; each window is divided for downlink and uplink transmissions. In the downlink, a beacon serves the satellite to announce the beginning of the frame, which corresponds to an uplink period divided into w time slots intended for contention. The beacon

TABLE III: State of the Art

Reference	Strategy	Suitability for Dts-IoT
[15]	Random tour and gossip-based aggregation algorithms rely on the interaction between adjacent nodes of the network.	Unsuitable, requires interaction between nodes which is not assured in the DtS-IoT architecture .
[20] & [22]	BFS approach based on CR logic for P2P networks. Nodes discover the network size by interacting with neighboring nodes	Unsuitable, nodes require P2P communication with other network's nodes.
[23]–[25]	Survey of CR approaches for population estimation in closed and open computer networks [23], and CR approaches for closed wireless networks [24], [25].	Unsuitable for closed and open populations due to the rapid network size changes, the node identity storage requirement, and meticulous control of all nodes present in a region.
[26]–[28]	Estimation mechanism according to two PDFs of the number of successful and collided preambles during contention periods of LTE networks. Estimation effectiveness for up to 300 devices.	Unsuitable. The calculations to compute the PDF and the estimation are unfeasible considering CubeSat's limitation. Furthermore, the effectiveness is desired for bigger node population.
[29]	Scheme based on the irregular repeat slotted ALOHA (IRSA) protocol. User devices transmit multiple replicas of their packets in random slots. Their interference contribution is canceled by applying (SIC) techniques at the receiver, allowing the identification of new transmitter devices.	Unsuitable for CubeSats since SIC techniques require high processing and memory capacity at the receiver. Furthermore, the high expected channel congestion impairs the operation of this technique if a frame of adequate size is not used, which risks being excessively large.
[30]	Estimation of the tag population size of RFID networks based on the average run size of successful and collided slots in an FSA frame.	Unsuitable for a CubeSat serving multiple regions due to the high processing cost involved in performing complex numerical calculations at the beginning and end of each frame.
[31]	Dynamically and iteratively adapting the FSA frame length to the RFID network size according to the channel load. Network size is estimated using a minimum mean square error function, whose result is obtained through numerical solutions.	Not directly applicable. This scheme requires modifications to bear the cost of adapting the frame for each coverage region. Frame lengths risk being excessively large (*e.g.* 4096 slots for 1500 nodes with the proposed estimator).
[14]	Collision set size estimator based on a Poisson simplification of the maximum likelihood estimator of contender nodes given the observation of the channel load in RFID systems.	Suitable. The channel observation is feasible in a DtS-IoT scenario, and the complexity of the calculations is viable on a CubeSat-type nanosatellite.

is an invite to the IoT nodes against which the only possible response is a transmission attempt, *i.e.*, nodes do not remain idle. Consequently, during the estimation process, every node competes in each available frame for which it receives a beacon. We assume a cluster is stable for the duration of a frame, as depicted in Fig. 1.

Time synchronization between nodes and the satellite is assumed to properly operate the FSA logic. Although the synchronization aspect is out of the scope of this work, recent studies show it is feasible to maintain synchronization in DtS-IoT under certain constellation designs [33]. During the estimation process, the information sent by the IoT nodes may be successfully decoded by the CubeSat or not. The failures in the decoding process can be due to a) collisions, *i.e.* simultaneous reception of signals from different nodes, or to b) insufficient received signal power due to path loss and other communication channel conditions. When the power of the received signal is less than the sensitivity of the receiver, successful decoding is not possible; but note that very weak signals will be undetected by the receiver on the satellite. In such cases, the CubeSat will erroneously assume that time slot to be idle. Moreover, in this work, we call effective detection the valid detection of actual transmissions from the IoT nodes, while erasure happens when the CubeSat erroneously detects an idle time slot. The latter resembles an on-off channel model [34], [35].

Although all nodes under the footprint choose a slot upon beacon reception, there can be slots that remain idle if no node chooses that slot for a transmission attempt. Consequently, the satellite perceives three outcomes for a given slot: successful, collided, or idle, and can track the number of failures and successes within a frame.

We consider the activity of the CubeSat divided into two phases: a) an estimation phase and b) an operational phase. During the estimation phase, the CubeSat travels its orbit, announcing the beginning of new frames and decoding the received node transmissions at each time slot. This work aims to estimate the number of contender nodes n_k, using the information acquired during the contention of each frame k. The estimated number of contender nodes is noted y_k. We consider each frame k to be related to a specific region of the globe, and thus n_k corresponds to the number of nodes present in that region. For this, we assume the application of Earth-repeat orbits, further discussed in Section V. Multiple passes over the same region may be needed to estimate n_k. In that case, we consider each pass an iteration of the process, and the estimate of frame/region k at iteration m is denoted $y_{(k,m)}$. M denotes the total number of iterations for the estimation phase. This means that after M passes, the CubeSat advances to the operational phase, using the estimates to throttle the communication with the ground nodes.

The estimation phase may need to be repeated periodically to serve two purposes: 1) to update the network's estimated size at each zone so as to account for dynamics in the network population and 2) to adjust communication parameters according to the changing requirements of the DtS-IoT network. However, determining an optimal periodicity for the updates is out of the scope of this work.

B. Network Size Estimator Design

Following the mathematical development presented by Zanella in [14], let $v = \langle c, s, u \rangle$ be one observation of the set $V = \langle C, S, U \rangle$ which stores the number of collided, successful, and idle slots in the frame. The number of observations of the set V is numerically equal to the number of passes of the nanosatellite over the same zone considered in the estimation. Then, the conditional probability of observing v given that n

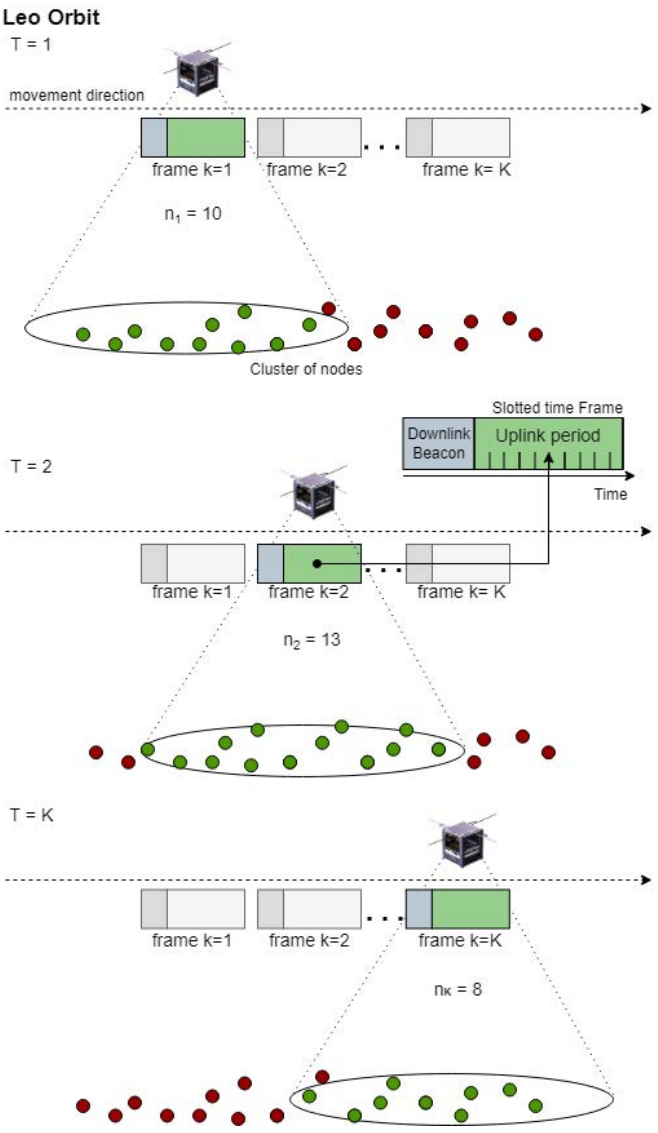


Fig. 1: Characteristics of the DtS communication.

nodes transmit can be expressed as

$$P_n(v) = P[V = v|n] = \frac{n! \sum_{j=0}^{c} \binom{c}{j} (-1)^{c-j} \sum_{l=0}^{c-j} \binom{c-j}{l} \frac{j^{n-s-l}}{(n-s-l)!}}{w^n \left[\binom{w}{s}\binom{w-s}{c}\right]^{-1}}, \quad (1)$$

where w indicates the length of the FSA frame in terms of slots. The development of this probability mass distribution is further explained in [36].

The classical maximum likelihood estimator is the one that returns the value of n that maximizes the probability shown in (1). However, such a calculation has several problems, such as the numerical instability of the computation of binomial terms, along with a high computational cost, which makes the maximum likelihood estimator inappropriate for implementation in constrained devices and much less in a nanosatellite. A simplification, suggested in [37], is to consider the number of transmissions in each slot as an independent Poisson random variable of mean $\mu = \frac{n}{w}$. The conditional probability, $P[V = v|\mu]$, of observing v given μ can then be expressed as

$$P_\mu(v) = \mu^s e^{-\mu w}(e^\mu - 1 - \mu)^c. \quad (2)$$

Now, to find the value that maximizes that expression, the derivative of (2) in μ is set to zero, which admits only one non-negative solution

$$\frac{\mu w - s}{c} = \frac{\mu(e^\mu - 1)}{e^\mu - 1 - \mu}, \quad (3)$$

which can be determined by the bisection search method [38].

The simplification made to get the expression of equation (2) allows the computational and energy cost to be adequate for a low-end device. However, the approximation restrains the mechanism to Poisson-type traffic. Consequently, our proposed estimation mechanism shares with [37] the design objective of using computationally simple calculations but overcoming the Poisson-type traffic restriction over the network traffic. We introduce our proposed estimator in the following section.

C. Optimistic Collision Information-Based Estimator (OCI)

Considering that n_k nodes are within the CubeSat footprint at region k, if every node transmits in the frame choosing a slot with probability $\frac{1}{w}$, we argue that a simple yet naive way of estimating n_k is by the consideration $\phi = s + 2c$, where ϕ is the estimation of n_k, assuming no more than two nodes are involved in every collision, and that the only way a transmission fails is due to collisions. This assumption is sufficiently accurate as long as $n_k \leq w$; otherwise, the probability of triple, quadruple, and higher order collisions becomes larger, and thus ϕ becomes an underestimation.

To assess the effectiveness of the naive estimator, we simulated the communication between the CubeSat and nodes using MATLAB. In this simulation, nodes are randomly distributed within the nanosatellite footprint, and a total of K regions are simulated with varying node population sizes. A 100% effective detection is considered; that is, the outcome of a transmission attempt can only be a success or a collision. We employ a step-size of 10 to account for network sizes n_k from 10 nodes to 2000 nodes , reaching a total of $K = 200$ different regions with different values of node population. For these simulations, $k = 1, 2, ..., K$ is a variable designating fictitious regions of increasing size.. The frame length is set to $w = 128$ as in [14]. At each step, the contention of n_k nodes over the w slots of an FSA frame is simulated, and the results in terms of c, s, and u are stored. The naive estimates $\phi = s + 2c$ are calculated at each step and stored in the array Φ of length K. The orange curve shown in Fig. 2 reflects the elements of Φ when plotted as a function of the number of nodes contending for a slot. The curve converges to a value equal to twice the number of slots in a frame due to high channel congestion for greater network sizes. However, setting a longer frame length, with $w = 512$, shows a better result, as depicted by the blue curve in Fig. 2. As long as triple and higher order collisions are not considered, this naive estimation always converges to $2w$, limiting its estimation capabilities for large network sizes.

A way to solve this issue, without knowing the order of the collisions, is to create a function that relates each point of the

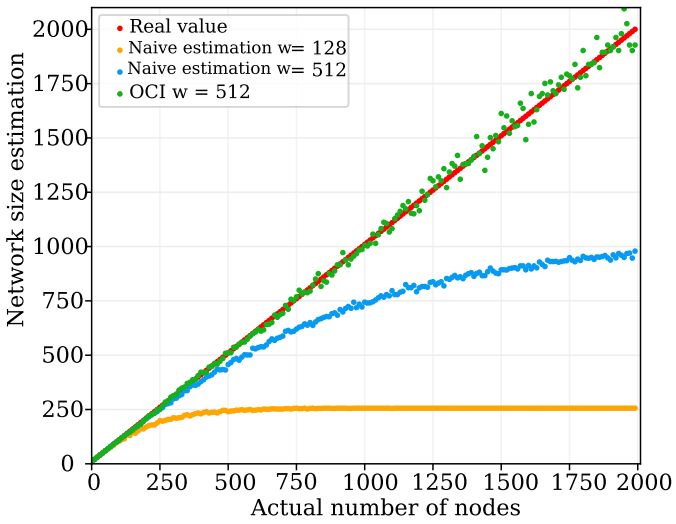


Fig. 2: Network size estimation with a naive approach (orange and blue curves) and with OCI estimator (green curve) for which the correction coefficients are [7.024e-09,-1.056e-05,0.006,-0.036,41.705]

naive estimation curve to the corresponding real value. For this to work correctly, bijectivity must be ensured between the naive estimation and the real value of nodes. Using a polynomial fit solves this matter as long as the frame length w is long enough in terms of slots so that the naive estimation has no repeated values for the simulated domain, thus ensuring injectivity. In a practical scenario, this function would be provided to the CubeSat, in such a way that it only uses the naive estimate method and then calculates the image using the given function, keeping the estimation process as simple as possible. Accordingly, we propose to use a first polynomial fit of order seven to smooth the naive estimation curve, ensuring injectivity. Then, we calculate the coefficients required to relate each point of the smooth curve to the real number of nodes by using a second polynomial fit; the latter maps the elements of the naive estimates ϕ to the value of the real number of nodes. The order of the second polynomial fit has been chosen to be high enough with the purpose that the correction oscillates less around the true values (*i.e.*, the red curve in Fig. 2). In the simulation, the order of the second polynomial fit is set to 4 by inspection; however, the impact of this parameter is further discussed in Section V.

The coefficients of a polynomial of degree q that best fit the estimation $y(x)$, $y(x) = p_0x^q + p_1x^{q-1} + \cdots + p_qx^0$, in a least-squares sense, can be calculated from

$$\mathbf{X} \cdot \tilde{\mathbf{P}} = \tilde{\mathbf{N}} \tag{4}$$

where $\mathbf{X}$ represents the $K \times q+1$ Vandermonde matrix formed from the elements of the naive estimations array Φ, on a geometric progression. $\mathbf{P}$ is a row vector of length $q+1$ containing the polynomial coefficients in descending powers to be calculated, and $\mathbf{N}$ is a K length vector containing the real value of nodes n_k. Thus, the detailed composition of each of these elements is

$$\begin{pmatrix} \phi_1^q & \phi_1^{q-1} & \cdots & 1 \\ \phi_2^q & \phi_2^{q-1} & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ \phi_K^q & \phi_K^{q-1} & \cdots & 1 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ \vdots \\ p_q \end{pmatrix} = \begin{pmatrix} n_1 \\ n_2 \\ \vdots \\ n_K \end{pmatrix} \tag{5}$$

In order to calculate the coefficients in $\mathbf{P}$, the matrix $\mathbf{X}$ is decomposed on a orthogonal-triangular manner, also known as QR decomposition, such that $\mathbf{X} = \mathbf{Q} \cdot \mathbf{R}$, where the $\mathbf{R}$ factor is an K-by-$(q+1)$ upper triangular matrix and $\mathbf{Q}$ is an K-by-K orthogonal unitary matrix, such that $\mathbf{Q}^\top \cdot \mathbf{Q} = \mathbf{I}$. Then, $\mathbf{P}$ is calculated as

$$\mathbf{P} = \mathbf{R} \setminus (\mathbf{Q}^\top \cdot \mathbf{N}). \tag{6}$$

This process is valid if the polynomial degree q is less than the length of $\mathbf{X}$. Once the coefficients are determined, they can be stored in CubeSat's ROM memory. Due to the processing and energy costs of these calculations, a ground-based station is responsible for the polynomial regression. Architectures with down-link and up-link channels have been developed for nanosatellites such as CubeSats [39]–[43], allowing these low-cost devices to download and store data from an earth station. A practical example of the latter is Swarm Technologies, which uses nanosatellite constellations for IoT communication using a store and forward design [44].

These ground-based stations are also responsible for studying the conditions of the channel and readjusting the coefficients in case of changes. The recalculation of the coefficients is done by integrating the new channel characteristics to the simulation, followed by the polynomial regression. This work represents these channel characteristics using the effective detection percentage.

The CubeSat's detailed process at the end of each frame is depicted in Algorithm 1. Step 1 is the calculation of the naive estimation ϕ using input parameters s and c. After declaring y in step 2, steps 3 to 5 calculate y as the evaluation of ϕ by the polynomial whose coefficients are present in vector $\mathbf{P}$, in decreasing degree order. The calculation corresponds to Horner's method of evaluating a polynomial, which has proven to be optimal in terms of arithmetic operations [45] as it only requires some additions and multiplications equal to the order of the polynomial. Step 6 returns the estimate $\overline{y_{(k,m)}}$, which is the equally-weighted average of the estimate of m passes if $M > 1$, or y if $M = 1$. The design variable M (*i.e.*, the number of satellite passes devoted to the estimation stage) that governs the iterations m, is finite and should be vastly smaller than the duration of the operation stage of the network. In step 7, the CubeSat overwrites $\overline{y_{(k,m-1)}}$ with the new estimate $\overline{y_{(k,m)}}$.

The result of applying Algorithm 1 to the naive estimates associated with $w = 512$ and $M = 1$ are illustrated by the green dots in Fig. 2.

We call our proposed estimation mechanism the Optimistic Collision Information (OCI) based estimator. The initial OCI's estimation curve in Fig. 2 has a noisy behavior, which can be refined by carrying out the estimation in multiple passes (i.e $m > 1$ in algorithm 1) and progressively averaging the estimations of each pass of the CubeSat over the same zone.

Algorithm 1: OCI algorithm

Input: s, c: number of successful and colliding slots within FSA frame k on iteration m
Stored Data:
$p[q+1]$: array of size $q+1$ of the correction coefficients in decreasing degree order
M: the number of satellite passes for the estimation
$\overline{y_{(k,m-1)}}$: estimated number of contenders for frame k at the previous iteration (only if $M \geq 2$)
Output: $\overline{y_{(k,m)}}$: estimated number of contenders for frame k at iteration m

$\phi = s + 2c$;
$y = 0$;
for *a = 1 to (q+1)* **do**
  $y = y \cdot \phi + p[a]$;
end
Return $\overline{y_{(k,m)}} = \overline{y_{(k,m-1)}} \cdot \frac{m-1}{m} + y \cdot \frac{1}{m}$;
Replace in memory: $\overline{y_{(k,m-1)}}$ by $\overline{y_{(k,m)}}$

D. Additional Benchmark Scheme

In [31], sMMSE is presented as a two-phase algorithm: a) an adaptation phase where the FSA frame size is dynamically and iteratively increased to be adapted to the number of devices and b) the estimation phase with a minimum square error function. The adaptation phase consists of increasing the frame length if the percentage of slots where transmission was received is larger than a predefined threshold. The process is repeated until the response rate RR is lower than the threshold, meaning that the frame length is adapted to the node population size to be estimated. An optimal set of parameters is provided in [31] so that the frame is doubled when 40% of its slots are occupied. The starting frame size of the algorithm is 128 slots in [31], resulting, as an example, in a length of 4096 slots for a 1500 node population, representing a considerable resource efficiency disadvantage in comparison to other schemes with a fixed frame size. Another downside is the number of iterations needed to adapt the frame, which implies an additional adaptation cost. To adapt sMMSE to the requirements of our scenario, the frame is started at an initial length of 512 slots, which reduces the number of iterations. The explanation for this choice is found in Section III.D.

The estimation phase starts right at the end of the adaptation phase. This phase consists of calculating the estimated number of nodes $\hat{n}$ that minimize the distance between the observed tag responses v_f and the expected value V_f, according to

$$H_{sMMSE}(v_f) = \arg \min_{n \in N} P_n(v_f), \tag{7}$$

which, using a binomial distribution of the probability of node occupancy and Chebyshev's inequality [31], is expressed as

$$\begin{aligned} P_n(v_f) &= ||v_f - V_f||^2 \\ &= (RR - 1 + \psi^n)^2 + (1 - RR - \psi^n)^2, \end{aligned} \tag{8}$$

where RR represents the response ratio, $\psi = \left(1 - \frac{1}{kL}\right)$, k and L are parameters such that kL represents the FSA window size, and n is the number of devices to be estimated.

IV. Performance Evaluation

In this section, we provide an analysis of the performance of OCI. We selected the Zanella (henceforth *Zan*) collision set size [14] and the sMMSE estimators [31], discussed in Sections II and III, for comparison purposes. Furthermore, we evaluate the impact of using network size estimators to provide feedback to the MAC protocol in the scenario of study.

A. Network Size estimator Performance

To measure the performance of the estimators, we calculate the Root-Mean-Square Error (RMSE), commonly used to rate how a regression model fits a dataset. For this, we compare the elements of the array of the real number of nodes N present in the satellite footprint to the ones of the array of the estimated size with the equation

$$\varepsilon_{RMSE} = \sqrt{\sum_{h=1}^{K} \frac{(\overline{y_{(h,M)}} - n_h)^2)}{K}} \tag{9}$$

RMSE is heavily penalized when an estimate deviates far from the actual observed value. For this reason, it is used in cases where significant errors are undesirable. Zan and sMMSE estimations are calculated for benchmarking purposes to compare to the performance of OCI. In previous work, we have adapted and evaluated the Zan estimator to DtS-IoT scenarios [16]. All simulations are developed using MATLAB R2020a, with a fresh implementation of OCI and sMMSE, and using Zan's code as provided by its author [46]. To ensure fair comparisons, sMMSE's frame size is set at 512 as its starting point.

To evaluate the performance, the first set of simulations considers n_k nodes distributed randomly within the nanosatellite footprint, ranging from 10 to 2000 with steps of 10, for a total of $K = 200$ different node populations. At each size-step, n_k nodes contend for a random slot in the FSA frame, and the estimations of Zan and OCI are calculated accordingly, using the record of successes and collisions, s and c. Effective detection of 100% is considered. Furthermore, for each $k = 1, 2, ..., K$, we consider multiple satellite passes M, ranging from 1 to 200. For each pair (k, M), the estimation results of OCI and Zan are denoted as $\overline{y_{(k,M)}}$ and $\overline{z_{(k,M)}}$, respectively. Note that the estimate $\overline{y_{(k,M)}}$ is equal to the average of all estimates $y_{(k,m)}$ for $m = 1, 2, ..., M$. The matrices $\bar{\mathbf{Y}}$ and $\bar{\mathbf{Z}}$, expressed in (10), are constructed using the aforementioned OCI and Zan estimates, respectively.

$$\overline{y_{(k,m)}} = \sum_{h=1}^{m} \frac{y_{(k,h)}}{m}$$

$$\overline{z_{(k,m)}} = \sum_{h=1}^{m} \frac{z_{(k,h)}}{m}$$

$$\bar{\mathbf{Y}} = \begin{bmatrix} y_{(1,1)} & \cdots & y_{(k,1)} & \cdots & y_{(K,1)} \\ \vdots & \ddots & \vdots & & \vdots \\ \overline{y_{(1,m)}} & \cdots & \overline{y_{(k,m)}} & \cdots & \overline{y_{(K,m)}} \\ \vdots & & \vdots & \ddots & \vdots \\ \overline{y_{(1,M)}} & \cdots & \overline{y_{(k,M)}} & \cdots & \overline{y_{(K,M)}} \end{bmatrix} \quad (10)$$

$$\bar{\mathbf{Z}} = \begin{bmatrix} z_{(1,1)} & \cdots & z_{(k,1)} & \cdots & z_{(K,1)} \\ \vdots & \ddots & \vdots & & \vdots \\ \overline{z_{(1,m)}} & \cdots & \overline{z_{(k,m)}} & \cdots & \overline{z_{(K,m)}} \\ \vdots & & \vdots & \ddots & \vdots \\ \overline{z_{(1,M)}} & \cdots & \overline{z_{(k,M)}} & \cdots & \overline{z_{(K,M)}} \end{bmatrix}$$

Each row of $\bar{\mathbf{Y}}$ and $\bar{\mathbf{Z}}$ compose a different estimation phase of m passes. We calculate the RMSE between rows and the real values of nodes that were set up in the simulations, resulting in the 200 points plotted for each mechanism in Fig. 3. The adaptation phase of sMMSE is not considered for the count of satellite passes. This means that the first satellite pass is considered to be carried out when the adaptation has already been made in all regions. Because of this, the different estimates of each pass result from the numerical solution of (7) with a fixed frame size but a different response rate (RR) at the end of the communication window.The simulation is then repeated, considering three different effective detection percentages: 100%, 95%, and 75%.

Results in Fig. 3 show that OCI outperforms Zan and sMMSE in scenarios with more realistic effective detection rates, lower than 100%. Zan's RMSE increases when the effective detection percentage decreases, since the Poisson approximation of the traffic is no longer valid. The Zan mechanism underestimates the number of nodes, given that a lower effective detection percentage means that a portion of nodes does not manifest their presence in the frame. Something similar happens with sMMSE as the response rate (RR) is decreasing due to reduced detections, and thus sMMSE underestimates the number of nodes. On the contrary, OCI's correction coefficients implicitly have this information, which explains why the resulting RMSE remains at a low value with lower effective detection rates. Additionally, OCI's RMSE decreases as the effective detection ratio is reduced. When the collision probability decreases, the probability of multiple collisions (triple and further) also decreases, rendering OCI's approximation more accurate. On average, Zan's RMSE is 4 times higher than OCI's considering a 95% effective detection scenario, and 38.7 times higher on a 75% effective detection scenario. Zan's RMSE is close to sMMSE's when imperfect detection ratios are considered.

When a higher number of satellite passes is considered for the estimation phase, the RMSE decreases progressively, but at a decreasing rate for the three estimators. Nevertheless, none of the RMSEs reached a point of convergence, as shown in Fig. 3 where the RMSE lowers asymptotically. Overall, the results on RMSE are good for all the estimators since, when confronted with a large number of nodes, the error between the estimate and the real number is slight.

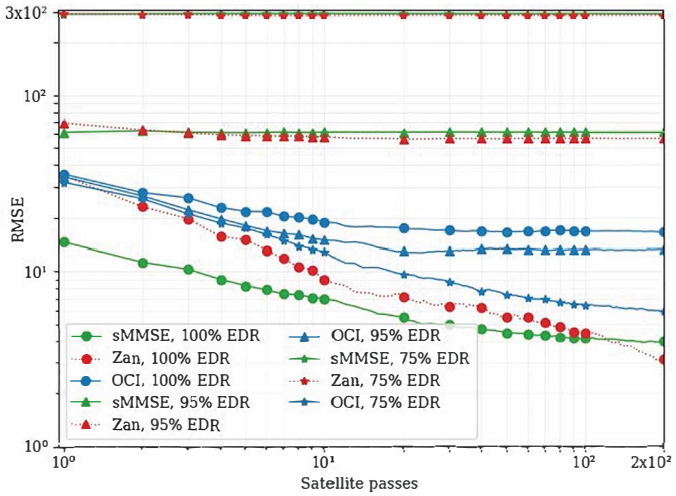


Fig. 3: RMSE (log scale) obtained from the simulations of the three estimators under different effective detection rates (EDR) and up to 200 satellite passes in the estimation phase. Initial frame length $w = 512$ slots for Zan, sMMSE (assuming the adaptation phase is finished), and OCI.

These results show the adaptability of the OCI mechanism over different scenarios, maintaining an RMSE lower than Zan and sMMSE when there is a nonzero percentage of signal loss. Space-ground channels are more prone to packet errors and low signal-to-interference-plus-noise ratio (SINR) values than ground IoT links. With this in mind, OCI's effectiveness under imperfect detection ratios makes it a better candidate for DtS-IoT.

An important aspect to compare is the efficiency of channel resources. In an environment such as the DtS-IoT, it is desirable to minimize the number of wasted channel resources, as they may negatively impact the network's performance. Fig. 4 shows a comparison between the frame length of each scheme. Note that the low RMSE in sMMSE's estimation goes along with low channel resource efficiency, as it has to increase the frame length to keep an occupancy rate lower than 40%. Furthermore, note that we have omitted the adaptation phase of sMSSE in Fig. 3; however, when the two phases are considered, the FSA frame adaptation phase of sMMSE requires four satellite passes to reach a 512-frame length as the starting point for the estimation. In such a case, the green curve of sMMSE in the figure would shift to the right, rendering a performance similar to Zan's. In addition, considering the two phases, sMMSE could only provide an estimate at the 5^{th} pass, whereas OCI and Zan can start the estimation process from the first pass.

B. FSA with Network Size Feedback

In [13], the Slotted Aloha Game is proposed to control access to the medium in a satellite network. The proposal focuses on determining the transmission probabilities of a group of

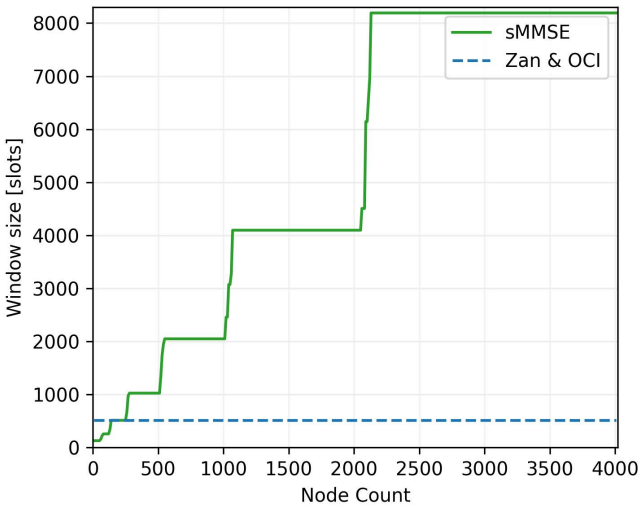
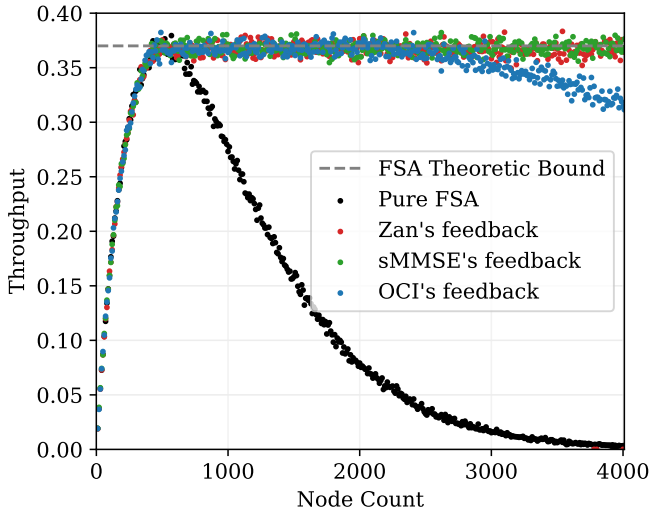


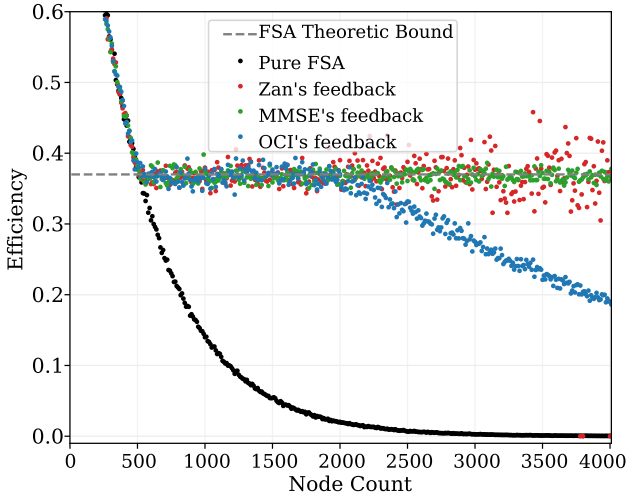
Fig. 4: Frame length comparison between sMMSE, Zan, and OCI versus the number of estimated nodes.

independent terminals (nodes) that share the communication channel and transmit via the FSA protocol. The probability of transmission of each node depends on the number of active nodes, in such a way that if this number is less than or equal to the number of slots in the frame, they are allowed to transmit with a probability equal to 1. However, if the number of active nodes exceeds the number of slots, the probability of transmitting decreases in such a way that, on average, only a number of nodes equal to the size of the frame transmit. The evaluation of the Slotted Aloha Game shows to allow the communication of a large number of nodes, maintaining an approximately constant throughput, with a low access delay and a low power consumption [13]. Similarly to [12], the authors assume that the number of active nodes is perfectly known.

In this section, we use the network size estimates to set the threshold employed by an FSA protocol following the aforementioned Slotted Aloha Game. Two metrics are calculated to evaluate the performance: throughput and normalized energy efficiency. Both pure FSA and FSA using the Slotted Aloha Game approach fed with network size feedback are implemented in the simulation for comparison purposes. First, a scenario with 100% effective detection is simulated, where Fig. 5a shows the throughput calculated as the number of successful transmissions in a frame divided by the time length of the frame (*i.e.* $w = 512$), considering one pass of the satellite for the estimation ($M = 1$). Then, Fig. 5b shows the normalized energy efficiency, which is calculated considering that when a sensor node transmits, it consumes one arbitrary unit of energy as in [13]; then, the energy consumed by successful transmissions is divided by the total energy consumed during the entire process for transmissions. The results represent the averaged results of 30 repetitions for each estimator. The results show the performance for twice the number of nodes that OCI was designed for. This is to appraise the scalability of the designed estimation mechanism, and the viability of letting the node population grow without re-adjusting the polynomial coefficients and frame length.



(a) Throughput comparison with 100% effective detection



(b) Energy efficiency comparison with 75% effective detection

Fig. 5: Comparison of FSA and FSA with size estimation feedback, considering the average of 10 passes and a 512 slots window size for Zan and OCI

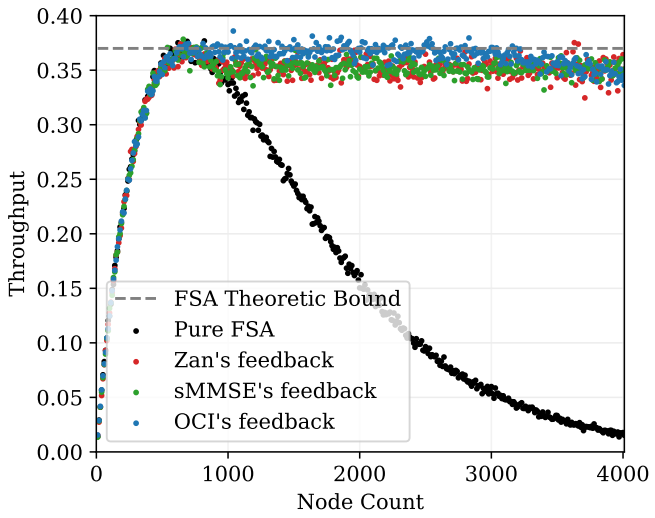
Analyzing the 100% effective detection scenario, in Figs. 5a and 5b, it can be seen that Pure FSA's throughput decreases once the number of nodes contending exceeds the number of slots in the frame and its energy efficiency decreases with the number of nodes. When either of the three estimators' feedback is applied, these two metrics have the same behavior until the network's size exceeds the frame length. From that point, throughput and energy efficiency oscillate close to 0.37. However, with a number of nodes higher than 2000, OCI starts to show slightly worse performance than sMMSE and Zan, while the latter shows discontinuities when the network's size is close to 4000 nodes. These results allow us to affirm that the feedback from the three estimators benefit the MAC protocol performance even if the number of nodes for which

its parameters were adjusted is exceeded. Zan and sMMSE show the best result for this scenario.

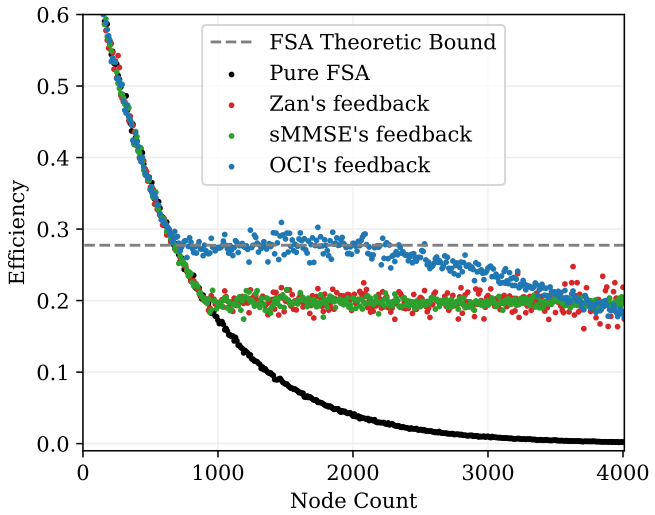
On the one hand, the discontinuities that Zan presents (seen by the red dots at abscissa 0), which translate into 0 throughput, and the slightly worse performance of OCI are explained by the number of nodes for which the estimators' parameters were chosen. The algorithm designed by Zanella in [14] returns an estimation equal to infinity once the number of nodes increases past the point where all the slots in a frame collide. On the other hand, OCI's polynomial coefficients are adjusted for a setting with a maximum of 2000 nodes, for which a frame length of $w = 512$ slots is suitable. Once the number of nodes increases considerably beyond the original maximum, the estimator progressively losses estimation capability until it converges to an estimate of 2415 nodes for a population size of 4000 nodes. Nevertheless, the throughput is maintained close to the theoretic bound by allowing 21.2% of the present nodes to transmit [13], which is the percentage that $w = 512$ represents to a 4000 node population. To improve the performance for larger network sizes, the frame length should be increased, and the coefficients for OCI's correction recalculated for the new network size. However, depending on the characteristics of the application, it might be unsuitable to adopt a number of slots past a certain length. sMMSE's estimation needs first an adaptation period which explains why its behavior is maintained close to the theoretic bound even for large node population sizes.

We also carried out a simulation with up to 200 satellite passes; the results showed that a single satellite pass devoted to the estimation process is enough to improve the performance, and thus a plot of these results is not shown in this manuscript. Making use of more passes narrows the resulting points in the graph but does not bring any substantial improvements.

To account for more realistic scenarios, we simulated with a 75% effective detection at the satellite. Fig. 6a shows a slightly worse performance, in terms of throughput, of FSA with Zan's and sMMSE's feedback compared with using OCI's feedback. Due to the 25% of ineffective detection, the frame is less congested, which translates to higher probabilities of successful messages. This latter dampens the effects of the underestimation of Zan and sMMSE and explains the difference in performance. When comparing energy efficiency, Fig. 6b shows a bigger difference in favor of OCI's feedback since the better estimate allows better control on node transmissions, mitigating energy waste. After the node count exceeds 2000, the number for which OCI was designed, the performance decreases towards matching Zan's and sMMSE's feedback performance at 4000 nodes. Overall, the results of Figs. 5a and 6a show that the peak throughput is achieved at a network size equal to the frame length. Nevertheless, when using OCI, the throughput is maintained at a high level for large network sizes of up to 4000 nodes. Hence, the supported network size is eight times the number of nodes where the throughput started to decrease for the Pure FSA scheme.



(a) Throughput with 75% effective detection



(b) Energy efficiency with 75% effective detection

Fig. 6: Comparison of FSA and FSA with size estimation feedback, considering the average of 10 passes and windows of 512 slots for Zan and OCI

V. Practical Considerations

In this section, we discuss relevant practical aspects involved in the estimators' implementation, such as *(i)* the applicability according to the traffic arrival process; *(ii)* the hardware requirements relative to computational costs and memory capacity; and *(iii)* considerations related to the projection of the satellite trajectory over the Earth surface.

A. Arrival Process

One advantage of the OCI estimator is its independence from the nature of the data arrival process at the satellite. For the Zan estimator to be applicable in restricted IoT gateways, it is necessary to use the simplification in which the number of transmissions in each slot of the frame is considered to be an independent Poisson random variable. Such a restriction is not present in OCI. However, the OCI estimator is limited to be predefined for a precise frame length used during the estimation process. To mitigate this limitation,

when the number of slots in the contention window is expected to change, a lookup table can be stored in CubeSat's ROM, containing the coefficients needed for the algorithm adjusted to different window sizes.

B. Computational Cost

Computational cost is an area where differences in estimators can be observed. In the case of the Zan estimator [14], it increases with the frame length w, according to $\log_2 w$. Instead, OCI's computational cost is independent of the frame length but grows with the order of the polynomial function used for the correction process depending on the polynomial evaluation method. For instance, Horner's method [45] has a complexity $O(q)$, where q represents the order. sMMSE's complexity depends on the numerical method used to solve its equation. To this end, and to validate the complexity analysis, OCI and Zan estimators were empirically tested on a Raspberry Pi Zero W board, which is currently used as the on-board computer (OBC) of CubeSat implementations such as in [47]. The hardware characteristics of the emulated OBC are specified in Table V. The test is based on calculating the average time needed to compute 10^5 repetitions of estimations using both methods and considering a contention window $w = 512$ slots. The experiment concluded with both estimators being suitable for a CubeSat in terms of computational cost: OCI shows a time requirement of $8.13 \cdot 10^{-5}$ s while Zan shows a requirement of $3.35 \cdot 10^{-4}$ s. The reported times demonstrate the advantage of OCI in terms of computations required over Zan, as OCI needed only 24.3% of the time demanded by Zan's method while achieving very similar or better performance. However, the sMMSE simulations could not be run on the same hardware platform, so to fairly compare the three estimators in terms of computation time, the same experiments were run on a computer. Now, the average time used to compute 10^5 repetitions of estimations was calculated using Python 3.10.7. The results, summarized in Table IV, show $1.22 \cdot 10^{-3}$ s on average for sMMSE, $5.48 \cdot 10^{-6}$ s for Zan, and $7.59 \cdot 10^{-7}$ s for OCI. For this test, only the estimation phase was considered for sMMSE, using $k = 128$ as done in [31] and different values for RR and L calculated by independent simulations of the first phase. These results confirm OCI as the fastest scheme since OCI needed only 13.9% of the time required by Zan's method and 0.06% of the time required by the sMMSE method.

TABLE IV: Computational Costs

Scheme	Avg. Compute Time	Ratio
sMMSE	$1.22 \cdot 10^{-3}$ s	100 %
Zan	$5.48 \cdot 10^{-6}$ s	0.45%
OCI	$7.59 \cdot 10^{-7}$ s	0.06%

To further compare Zan and OCI in terms of time requirements, we carried out a simulation varying the frame length from 128 to 2048 slots with steps of 128. The results are shown in Fig. 7 where time is normalized to the maximum value reached by the Zan estimator when a frame of 2048 slots is used. As can be seen, Zan's computation cost increases along with the frame length, whereas OCI's time requirement

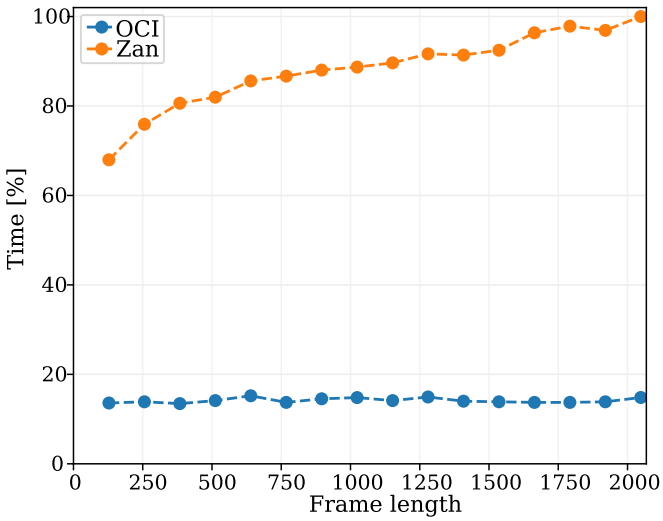


Fig. 7: Normalized computation time required for both estimators varying the frame length.

TABLE V: Main characteristic of the emulated CubeSat on-board computer

Component	Characteristic
Processor	Broadcom BCM2835 ARM1176
Architecture	ARMv6 (32bit)
Speed	1.0 GHz
RAM	512 MB
OS	Raspbian 10 (buster) Linux 5.10.17

remains at the same values except for small random variations. Furthermore, the Zan estimator is proven to be more time efficient than the best-performing estimators for RFID tags up to the year of its publication [14], which means that OCI also outperforms methods like [48] in terms of computational cost. In [49], a look-up table is proposed to store all the possible estimations for different combinations of FSA frame results (only two among s, c and u due to the constraint of $s + c + u = w$) of Vogt's method [48] to improve time delays. For OCI, a look-up table could also be implemented. This would imply negligible delays but there is a need for storing the polynomial evaluations of all possible frames, which results with a fixed w.

We also compared the memory needed for both estimators. Since the satellite travels at approximately 7.5 km/s in its orbit, it goes around the world in 90 minutes, i.e., 5400 seconds. It is common in the literature that the length of a slot in a communication protocol such as FSA lasts an amount of time equal to the time of packet transmission. In the existing literature, applications with packet lengths of around 1000 bits can be found [50]; the packet transmission times vary depending on the IoT application. If the transmission of a packet takes for example 5 ms [51], then a frame takes $512 \cdot 5 = 2560$ ms, that is 2.56 seconds. Then, with that frame length, there could be 2109 contention windows, *i.e.* positions, in a turn around the world. In the first satellite pass, OCI needs to store two integers of 3 digits maximum (the number of successes and collisions among the frame slots) and the

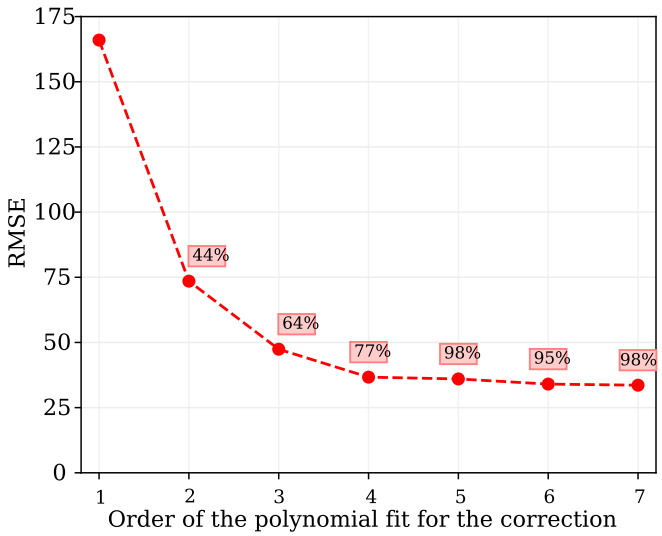


Fig. 8: RMSE of OCI as a function of the order of the polynomial fit used for the correction.

estimation made with these numbers, for each position. The simplest way of storing a number requires the use of 10 bits for a 3-digit number; thus $2 \cdot 10 \cdot 2109 = 42.180$ bits are needed in the first pass, plus the estimate. According to the experiments, each position's size estimate is an integer of up to 4 digits, which means $14 \cdot 2109 = 29,526$ bits are also needed in the first pass. OCI does not require storing the results of an FSA frame in long-term memory, but only the estimates and the progressive averaging of these results, which is a shared feature with Zan.

For OCI to work properly, the set of coefficients required for the correction is to be stored in memory. These coefficients are numbers with up to 15 digits that may require up to 64 bits each, which then implies that $64 \cdot C$ bits are required, with C being the polynomial order. Fig. 8 shows the impact that the order of the polynomial correction has on the RMSE of OCI under the same experiment carried out in Section III. In Fig. 8 each label represents the error percentage of each point with respect to the previous order. It can be seen that a coefficient of order 4 is sufficient to have a low RMSE compared to lower orders. If a 4 order is chosen, then 256 bits would be needed.

C. Repeating Orbits

As mentioned in Section II, one important issue in estimating the size of the network beneath a passing by satellite is that the ground track does not necessarily repeat. In the literature, orbits that repeat the exact same ground track are known as Earth-repeat orbits [18], [19]. Such Earth-repeat orbits exploit the nodal precession effect (a phenomenon provoked by the non-spherical nature of the Earth) to shift the orbital plane such that the ground track matches a previous one [52]. In other words, the nodal precession is used to balance out the offset of the revolution of Earth. Thus, the same ground track can be covered after a controlled number of orbital revolutions in a given time interval [18]. Although the efficient determination of Earth-repeat orbits demands specific computation and orbit maintenance techniques [53], it unlocks the possibility of learning from reiterative passes over the same region of the network. The longitudinal rotation after a certain period of time of a planet is

$$\Delta L_1 = -2\pi \frac{T}{T_E}, \tag{11}$$

while the quantization of the effect of the nodal precession can be expressed as

$$\Delta L_2 = -\frac{3\pi J_2 R_e^2 cos(i)}{a^2(1-e^2)^2}, \tag{12}$$

where T_E a sidereal day, J_2 is the Earth's second dynamic form factor, R_e is the Earth's radius, i is the orbital inclination, a is the orbit's semi-major axis, e is the orbital eccentricity, and T represents the elapsed time, which can be expressed as a function of the standard gravitational parameter for the Earth μ, like $T = 2\pi\sqrt{\frac{a^3}{\mu}}$.

Therefore, a repeat orbit sees these two effects canceling each other after a set of orbital revolutions j and sidereal days k, such that $j|\Delta L_1 + \Delta L_2| = k2\pi$. Then, the required condition for the orbit to repeat can be written as

$$j \left| -2\pi \frac{2\pi\sqrt{\frac{a^3}{\mu}}}{T_E} - \frac{3\pi J_2 R_e^2 cos(i)}{a^2(1-e^2)^2} \right| = k2\pi. \tag{13}$$

Replacing the variables with the parameters of the Earth and using similar parameters to those used in a known Chilean CubeSat named Suchai [54], [55], it is possible to identify different repeat orbits, as shown in Fig. 9. In this figure, dots of different colors represent a set of orbital inclination and orbital height parameters that holds the equation (13). Darker colors represent shorter repetition periods, and bigger markers indicate higher accuracy in the resolution of the equation (13). It should be noted that the parameters of the Suchai nanosatellite are misaligned since its orbit is not repetitive.

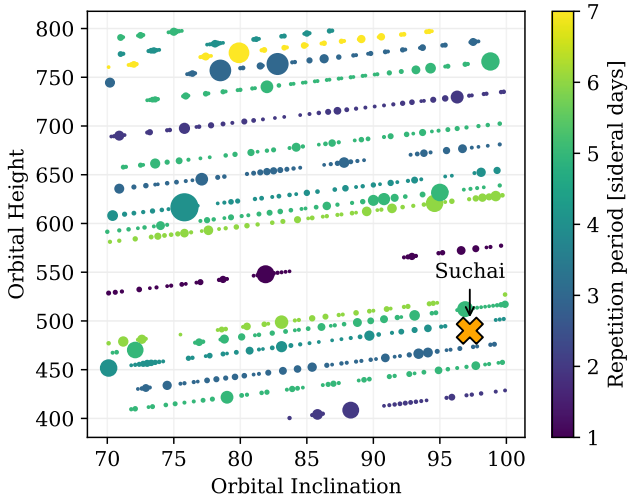


Fig. 9: Parameters of orbits that hold equation (13) for circular orbits with varying height and inclination. The current parameters for the Chilean Suchai satellite are highlighted.

Up to this point, the estimation process has been considered to be held by a single CubeSat. However, several studies on store and forward technology and its application to CubeSats have been done over the years [39], [40], [42]. With store and forward, a CubeSat can store its estimate on a ground terminal which can later be sent to another CubeSat of the same constellation. In this way, the estimation process done over multiple passes of a single CubeSat can be carried out by a single pass of multiple CubeSats.

VI. Conclusions

The massive and worldwide deployments of IoT networks face connectivity challenges related to the scalability, coverage, resilience, and ubiquity of these implementations. The Direct-to-Satellite IoT paradigm allows cost-effective solutions such as CubeSat nanosatellites in Low-Earth Orbit that serve as orbiting gateways to remote IoT nodes. However, achieving efficient use of temporal and spectral resources to overcome the exponential connectivity demands remains one of the main challenges of DtS-IoT. This efficient allocation of transmission resources is strongly favored when MAC protocols are aware of the number of contending nodes, but to the best of our knowledge, network size estimation mechanisms operating in IoT gateways deployed in continuously-moving LEO CubeSats have not yet been addressed in the literature. To tackle this issue, in this work we proposed OCI, a novel and computationally efficient network size estimator for DtS-IoT.

The results of the experiments carried out showed that OCI has low estimation error and computational costs, and that the estimate can be used to assist a MAC protocol, achieving a stable throughput serving up to 4000 nodes without energy efficiency penalties. Channel impairments, such as erasures, can be integrated into OCI's correction coefficients to adjust the estimates to the specific scenario for which it will be implemented. This provides OCI with adaptability in low detection ratio scenarios absent in the state-of-the-art estimation mechanisms used for benchmarking, further positioning OCI as a better candidate for DtS-IoT.

Regarding future work, we intend to design new approaches that favor an equitable distribution of channel resources in terms of medium access. Given the nature of a DtS-IoT network with Frame Slotted Aloha, the work in progress aims to provide terrestrial nodes with the capability to decide in which frame to transmit, given the estimated potential competition in each zone obtained with OCI estimations.

References

[1] J. Greig, "IoT industry projected to have economic impact of $11 trillion by 2025 - TechRepublic," https://www.techrepublic.com/article/iot-industry-projected-to-have-economic-impact-of-11-trillion-by-2025/, 2019, accessed: 2021-03-19.

[2] L. S. Vailshery, "Global IoT and non-IoT connections 2010-2025 Statista," https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/, 2021, accessed: 2021-04-19.

[3] M. Centenaro, C. E. Costa, F. Granelli, C. Sacchi, and L. Vangelista, "A Survey on Technologies, Standards and Open Challenges in Satellite IoT," *IEEE Communications Surveys & Tutorials*, 2021.

[4] I. F. Akyildiz and A. Kak, "The internet of space things/cubesats," *IEEE Network*, vol. 33, no. 5, pp. 212–218, 2019.

[5] M. A. Diaz, J. C. Zagal, C. Falcon, M. Stepanova, J. A. Valdivia, M. Martinez-Ledesma, J. Diaz-Peña, F. R. Jaramillo, N. Romanova, E. Pacheco, M. Milla, M. Orchard, J. Silva, and F. P. Mena, "New opportunities offered by cubesats for space research in latin america: The suchai project case," *Advances in Space Research*, vol. 58, pp. 2134–2147, 11 2016.

[6] G. Boquet, P. Tuset-Peiró, F. Adelantado, T. Watteyne, and X. Vilajosana, "LR-FHSS: Overview and Performance Analysis," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 30–36, 2021.

[7] S. Cluzel, L. Franck, J. Radzik, S. Cazalens, M. Dervin, C. Baudoin, and D. Dragomirescu, "3GPP NB-IOT coverage extension using LEO satellites," *IEEE Vehicular Technology Conference*, vol. 2018-June, pp. 1–5, 2018.

[8] I. Leyva-Mayorga, B. Soret, M. Röper, D. Wübben, B. Matthiesen, A. Dekorsy, and P. Popovski, "LEO Small-Satellite Constellations for 5G and Beyond-5G Communications," *IEEE Access*, vol. 8, pp. 184 955–184 964, 2020.

[9] J. A. Fraire, S. Céspedes, and N. Accettura, "Direct-to-satellite iot - a survey of the state of the art and future research perspectives," in *Ad-Hoc, Mobile, and Wireless Networks*, M. R. Palattella, S. Scanzio, and S. Coleri Ergen, Eds. Cham: Springer International Publishing, 2019, pp. 241–258.

[10] T. Ferrer, S. Céspedes, and A. Becerra, "Review and evaluation of mac protocols for satellite IOT systems using nanosatellites," *Sensors*, vol. 19, no. 8, pp. 1–29, 2019.

[11] C. Buratti, A. Conti, D. Dardari, and R. Verdone, "An overview on wireless sensor networks technology and evolution," *Sensors*, vol. 9, no. 9, pp. 6869–6896, 2009.

[12] K. Vogelgesang, J. A. Fraire, and H. Hermanns, "Uplink transmission probability functions for LoRa-based direct-to-satellite IoT: A case study," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 01–06.

[13] B. Zhao, G. Ren, and H. Zhang, "Slotted aloha game for medium access control in satellite networks," in *2019 IEEE/CIC International Conference on Communications in China (ICCC)*, 2019, pp. 518–522.

[14] A. Zanella, "Estimating collision set size in Framed Slotted Aloha wireless networks and RFID systems," *IEEE Communications Letters*, vol. 16, no. 3, pp. 300–303, 2012.

[15] R. Ali, S. S. Lor, and M. Rio, "Two algorithms for network size estimation for master/slave ad hoc networks," *2009 IEEE 3rd International Symposium on Advanced Networks and Telecommunication Systems, ANTS 2009*, pp. 23–25, 2009.

[16] P. Ilabaca, S. Céspedes, and S. Montejo-Sánchez, "Network size estimation in direct-to-satellite iot," in *2021 IEEE 8th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, 2021, pp. 107–111.

[17] SES, "GEO, MEO, and LEO. How orbital altitude impacts network performance in satellite data services," https://www.satellitetoday.com/content-collection/ses-hub-geo-meo-and-leo/, 2020, accessed: 2021-04-04.

[18] O. Montenbruck, E. Gill, and F. Lutze, "Satellite orbits: models, methods, and applications," *Appl. Mech. Rev.*, vol. 55, no. 2, pp. B27–B28, 2002.

[19] M. Lara, "Searching for repeating ground track orbits: a systematic approach," *The Journal of the Astronautical Sciences*, vol. 47, no. 3, pp. 177–188, 1999.

[20] J. Bustos-Jimenez, N. Bersano, S. E. Schaeffer, J. M. Piquer, A. Iosup, and A. Ciuffoletti, *Estimating The Size Of Peer-To-Peer Networks Using Lambert's W Function*. Boston, MA: Springer US, 2008, pp. 61–72.

[21] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the lambertw function," *Advances in Computational Mathematics*, vol. 5, no. 1, pp. 329–359, Dec 1996.

[22] J. Bustos-Jiménez, M. Hidalgo, and E. Schaeffer, "Estimating the size of natural networks using local information," *Proceedings - International Conference of the Chilean Computer Science Society, SCCC*, no. November, pp. 307–313, 2010.

[23] N. Accettura, G. Neglia, and L. A. Grieco, "The Capture-Recapture approach for population estimation in computer networks," *Computer Networks*, vol. 89, pp. 107–122, 2015.

[24] S. L. Peng, S. S. Li, X. K. Liao, Y. X. Peng, and N. Xiao, "Estimation of a Population Size in Large-Scale Wireless Sensor Networks," *Journal of Computer Science and Technology*, vol. 24, no. 5, pp. 987–997, 2009.

[25] A. Achtzehn, L. Simić, M. Petrova, and P. Mähönen, "IEEE 802.11 Wi-Fi access point density estimation with capture-recapture models," *2015 International Conference on Computing, Networking and Communications, ICNC 2015*, pp. 153–159, 2015.

[26] C. H. Wei, G. Bianchi, and R. G. Cheng, "Modeling and analysis of random access channels with bursty arrivals in OFDMA wireless networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 4, pp. 1940–1953, 2015.
[27] S. Duan, V. Shah-Mansouri, Z. Wang, and V. W. Wong, "D-ACB: Adaptive Congestion Control Algorithm for Bursty M2M Traffic in LTE Networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9847–9861, 2016.
[28] L. Tello-Oquendo, V. Pla, I. Leyva-Mayorga, J. Martinez-Bauset, V. Casares-Giner, and L. Guijarro, "Efficient random access channel evaluation and load estimation in LTE-A with massive MTC," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1998–2002, 2019.
[29] J. Sun, R. Liu, and E. Paolini, "Detecting the Number of Active Users in IRSA Access Protocols," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1972–1976.
[30] M. Shahzad and A. X. Liu, "Fast and Accurate Estimation of RFID Tags," *IEEE/ACM Transactions on Networking*, vol. 23, no. 1, pp. 241–254, 2015.
[31] L. Arjona, H. Landaluce, A. Perallos, and E. Onieva, "Scalable RFID Tag Estimator With Enhanced Accuracy and Low Estimation Time," *IEEE Signal Processing Letters*, vol. 24, no. 7, pp. 982–986, 2017.
[32] S. Cakaj, B. Kamo, A. Lala, and A. Rakipi, "The coverage analysis for low earth orbiting satellites at low elevation," *International Journal of Advanced Computer Science and Applications*, vol. 5, no. 6, 2014. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2014.050602
[33] J. A. Fraire, S. Henn, F. Dovis, R. Garello, and G. Taricco, "Sparse Satellite Constellation Design for LoRa-based Direct-to-Satellite Internet of Things," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.
[34] F. Tondo, S. Montejo Sánchez, M. Pellenz, S. Céspedes, and R. Souza, "Direct-to-Satellite IoT Slotted Aloha Systems with Multiple Satellites and Unequal Erasure Probabilities," *Sensors*, vol. 21, p. 7099, 10 2021.
[35] A. Munari, F. Clazzer, G. Liva, and M. Heindlmaier, "Multiple-relay slotted aloha: Performance analysis and bounds," *IEEE Transactions on Communications*, vol. 69, no. 3, pp. 1578–1594, 2021.
[36] J. Riordan, *An Introduction to Combinatorial Analysis*. Princeton University Press, 1978.
[37] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, vol. 2006, pp. 322–333, 2006.
[38] H. A. Eiselt and C.-L. Sandblom, *Linear programming and its applications*. Springer Science & Business Media, 2007.
[39] S. N. M. Rahim, J. Johari, S. A. Ence Ab Rahim, and M. H. Jusoh, "Estimation of communication link on Ground Sensor Terminal (GST) system for nanosatellite (UiTMSAT-1) store-and-forward mission," *ICSET 2018 - 2018 IEEE 8th International Conference on System Engineering and Technology, Proceedings*, no. October, pp. 108–111, 2019.
[40] D. Abbasi and M. Abolghasemi, "Store & forward communication payload design for leo satellite systems," *Majlesi Journal of Electrical Engineering*, vol. 10, pp. 7–18, 09 2016.
[41] A. Addaim, A. Kherras, and E. Zantou, "Design and analysis of store-and-forward data collection network using low-cost leo small satellite and intelligent terminals," *Journal of Aerospace Computing Information and Communication - J AEROSP COMPUT INF COMMUN*, vol. 5, pp. 1–11, 02 2008.
[42] A. Addaim, A. Kherras, and B. Zantou, "Design of store and forward data collection low-cost nanosatellite," in *2007 IEEE Aerospace Conference*, 2007, pp. 1–10.
[43] Q. Verspieren, T. Obata, and S. Nakasuka, "Innovative Approach to Data Gathering in Remote Areas Using Constellations of Store & Forward Communication Cubesats," in *31st International Symposium on Space Technology and Science (ISTS)*, 2017, pp. 1–6.
[44] "Swarm - Low cost, global satellite connectivity for IoT," accessed: 2022-07-9. [Online]. Available: https://swarm.space/
[45] V. Y. Pan, "METHODS OF COMPUTING VALUES OF POLYNOMIALS," *Russian Mathematical Surveys*, vol. 21, no. 1, pp. 105–136, feb 1966.
[46] dgt Dei Gruppo Telecomunicazioni, "MATLAB source code," http://dgt.dei.unipd.it/pages/read/54/, accessed: 2020-10-01.
[47] ECE3SAT, "OBC: On Board Computer - ECE3SAT," http://www.ece3sat.com/cubesatmodules/obc/, 2018, accessed: 2021-07-10.
[48] H. Vogt, "Efficient Object Identification with Passive RFID Tags," in *Pervasive Computing*, F. Mattern and M. Naghshineh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 98–113.
[49] A. T. H. Bui, C. T. Nguyen, T. C. Thang, and A. T. Pham, "A Comprehensive Distributed Queue-Based Random Access Framework for mMTC in LTE/LTE-A Networks with Mixed-Type Traffic," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 12 107–12 120, 2019.
[50] J. Su, G. Ren, Q. Wang, and H. Zhang, "Randomly Pre-coded Packets based Random Access Scheme for IoT-Oriented Satellite Networks," *IEEE Access*, vol. 8, 2020.
[51] D. Li, S. Wu, Y. Wang, J. Jiao, and Q. Zhang, "Age-Optimal HARQ Design for Freshness-Critical Satellite-IoT Systems," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2066–2076, 2020.
[52] S. D. Vtipil and B. Newman, "Determining an earth observation repeat ground track orbit for an optimization methodology," *Journal of Spacecraft and Rockets*, vol. 49, no. 1, pp. 157–164, 2012.
[53] Y. He, M. Xu, X. Jia, and R. Armellin, "High-precision repeat-groundtrack orbit design and maintenance for earth observation missions," *Celestial Mechanics and Dynamical Astronomy*, vol. 128, no. 2, pp. 275–294, 2017.
[54] N2YO, "SUCHAI Satellite details 2017-036Z NORAD 42788," https://www.n2yo.com/satellite/?s=42788, 2017, accessed: 2021-06-1.
[55] H.-A. G. Chris Peat, "SUCHAI - Orbit," https://www.heavens-above.com/orbit.aspx?satid=42788, 2017, accessed: 2021-06-1.