UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2024-078
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 20/09/2024 in Esch-Sur-Alzette

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN *Informatique*

by

Mohammad AFHAMISIS
Born on 17 October 1989 in Shabestar, IRAN

TIME SYNCHRONIZATION AND PACKET SCHEDULING FOR SATELLITE LORAWAN NETWORKS

Dissertation defence committee

Dr Maria Rita PALATTELLA, dissertation supervisor
Principal Research and Technology Scientist, Luxembourg Institute of Science and Technology (LIST)

Dr Christian VINCENOT
Professor, Université du Luxembourg

Dr Laurent PFISTER, Chairman
Head of ENVISON UNIT, Luxembourg Institute of Science and Technology (LIST)
Professor, Université du Luxembourg

Dr Alberto Ginesi
Head of the Telecommunication Systems and Techniques, ESA-ESTEC, Nederland
Professor, University of Bologna, Italy

Dr Ana Isabel Pérez-Neira, Vice Chairman
Director of the Centre Tecnológic de Telecomunicacions de Catalunya, CTTC, Spain
Professor, Polytechnic University of Catalonia, Spain

Time Synchronization and Packet Scheduling for Satellite LoRaWAN Networks

""But all this world is like a tale we hear"
Men's evil, and their glory, disappear. "

Abolghasem Ferdowsi, Shahnameh: The Persian Book of Kings (940 – 1025)

Acknowledgements

*I would like to express my greatest gratitude to **my parents**. To **my father**, who supported me unconditionally while he was alive, your guidance and encouragement have been the cornerstone of my academic journey. Your memory continues to inspire me every day. To **my mother**, your sensational support has been invaluable. Your unwavering belief in me and your constant encouragement have been a source of strength throughout this journey. I am deeply grateful for your love and support.*

*To my love, **Elham Majidnemati**, who motivated me every day of my PhD, your encouragement and belief in me have given me the strength and robustness needed to persevere through the challenges. Your support has been a pillar of my success, and I am forever thankful for your presence in my life.*

*To my supervisor, **Maria Rita Palattella**, who helped me in each section of my PhD, your supervision has been instrumental in reaching this point. You provided the right direction in my research, supported, and motivated me every day. Without your guidance, this achievement would not have been possible. Thank you for your invaluable support and mentorship.*

*This thesis was supported by the **Design of LoRaWAN protocol optimisation over SATellite Connection for precision agriculture applications** (**LORSAT**) Project, through the **National Research Fund Luxembourg (FNR)**, under Grant **CORE/C19/IS/13705191**.*

Abstract

The integration of Satellites and Internet of Things (IoT) networks offers a promising solution for enabling connectivity in remote and hard-to-reach areas, where consistent access to the internet and reliable power sources remains a significant challenge. Among IoT technologies, the LoRa modulation scheme and LoRaWAN protocol have been chosen for their low power consumption, Long-Range communication capabilities, and cost-effectiveness, making them suitable for such environments. However, these IoT networks still require intermittent internet access, which can be efficiently provided by satellites passing over these regions multiple times per day. This introduces the need for precise satellite orbit tracking to manage network availability effectively. During each satellite pass, a large number of devices fall within the satellite's coverage area, leading to potential packet collisions and communication failures. Consequently, this situation demands advanced time synchronization and packet scheduling techniques to optimize performance and ensure reliable network operation. To address these challenges, this thesis introduces two novel methods: REACT for LoRaWAN time synchronization and SALSA for LoRaWAN packet scheduling. REACT ensures that LoRaWAN devices operations are synchronized with satellite visibility windows, improving the overall network performance. SALSA is designed to efficiently manage transmission times, thereby reducing packet collision rates and enhancing communication reliability. These methods have been rigorously evaluated using first via a simulation environment, and then, a testbed, for implementing a Proof of Concept (PoC). The results demonstrate that REACT and SALSA perfectly operate to integrate Terrestrial LoRaWAN networks with Satellites, ensuring the required reliability and scalability. These contributions provide practical solutions to existing limitations in the field of satellite IoT, paving the way for the effective implementation of integrated LoRaWAN satellite technology, totally agnostic to the possible application scenarios.

Table of Contents

List of Figures

List of Tables

1 Introduction

In recent years, the IoT has emerged as a cornerstone of technological innovation, revolutionizing various sectors through the integration of connected devices that facilitate seamless communication and data exchange. IoT's transformative potential lies in its capacity to optimize operations, enhance productivity, and significantly reduce operational costs by enabling real-time monitoring, automation, and control of complex systems and processes. What distinguishes the IoT from traditional digital technologies is its ability to provide ubiquitous, low-latency connectivity, creating a network of billions of physical devices that operate autonomously while generating and transmitting vast amounts of data. This has profound implications not only for industries but also for society at large, as IoT-driven data streams enable more informed decision-making, improved resource allocation, and the emergence of entirely new forms of digital interaction.

1.1 Outline

The growing expansion of IoT and its integration across diverse sectors has dramatically increased the number of interconnected devices, which in turn has led to a substantial escalation in the volume of data being generated and transmitted. This rise in data enhances system performance, enables better decision-making, and drives innovation across industries and other applications. However, as the number of IoT devices continues to multiply, the demands placed on existing network infrastructures also intensify. The capacity of these networks must keep pace with the burgeoning influx of devices, but terrestrial networks often fall short of providing reliable and ubiquitous coverage, particularly in rural and remote areas. These networks, although essential for

IoT connectivity, are limited by inconsistent broadband access and significant coverage gaps, which hinder the seamless operation of IoT devices in areas that lack sufficient infrastructure.

In response to these challenges, this thesis presents novel methods and algorithms designed to address the critical connectivity and time synchronization issues and scheduling method associated with the proliferation of IoT devices. These solutions aim to enhance network scalability and reliability performance by integrating IoT devices into optimization frameworks that ensure continuous and effective connectivity, even in regions with limited terrestrial network support. The proposed approaches are developed to bridge connectivity gaps, allowing IoT devices to operate efficiently across various environments. By overcoming these infrastructural limitations, the methods and algorithms discussed in this thesis are designed to support the growing number of IoT devices, ensuring scalability, improved performance, and broader societal impact as IoT continues to expand.

1.2 Problem Statement

Deploying IoT devices in rural and remote areas is challenging [1], [2], where traditional terrestrial network infrastructure is either unavailable, too costly to implement, or lacks reliable power sources. The IoT devices in these regions are typically battery-powered with limited computational capabilities, and they perform sporadic or periodic transmissions with low throughput, primarily sending sensor measurement values and device states. While these devices are mostly used for monitoring mostly for agriculture, forest monitoring and similar applications, therefore they assumed to be the almost geographically-fixed IoT devices. Considering mobility in the IoT devices is not included in the scope of this thesis. Given these constraints, it is necessary to explore alternative network backhauling infrastructures that can provide consistent connectivity under such challenging conditions.

This is where Satellite IoT becomes essential. Satellite IoT can bridge the connectivity gap by providing backhaul for IoT networks in regions where terrestrial infrastructure is either unavailable or cost-prohibitive to deploy [3]. The integration of Terrestrial Networks (TN) and satellite networks is crucial for extending the reach and reliability of

IoT applications, ensuring that remote and rural areas are not left behind in the digital transformation [4]. The 3GPP has been actively standardizing satellite IoT, generally on of the Non Terrestrial Networks (NTN), to support this integration, particularly focusing on Low Earth Orbit (LEO) satellites, which offer lower latency, higher throughput, and better link budget than traditional geostationary satellites. This integration is expected to play a pivotal role in the 6G ecosystem, enhancing global coverage and service continuity for IoT applications [5]–[8].

To effectively integrate satellite networks with existing IoT networks, advanced solutions are required to manage the unique challenges posed by satellite communication environment. Satellites pass over rural areas several times per day, necessitating precise knowledge of their orbits to manage availability effectively. During these satellite passes, the large number of devices within the satellite's visibility window leads to high probabilities of packet collisions. This scenario underscores the need for advanced packet scheduling and time synchronization algorithms.

Among the Low Power Wide Area Network (LPWAN) IoT technologies, Long Range Wide Area Network (LoRaWAN)[9] based on the LoRa modulation [10] is one of remarkable IoT protocols available. LoRaWAN provides low power, long range and low cost connectivity, suitable for the remote areas. In this thesis, the combination of LoRa/LoRaWAN has been selected as the base technology to overcome these challenges.

1.3 Thesis Contributions

In this thesis, two novel methods have been introduced to address these challenges in the Satellite LoRaWAN networks: REACT - a low cost and specification-independent LoRaWAN time synchronization algorithm and SALSA - a LoRaWAN packet Scheduling Algorithm for LoRa to LEO satellite. REACT ensures that sensor operations are synchronized with satellite visibility windows and following SALSA allocated time schedules, improving the overall network performance. SALSA is designed to efficiently manage packet transmission times, thereby reducing packet collision rates and enhancing communication reliability.

These methods were rigorously evaluated using my custom-developed simulators and Testbed. The results demonstrate that REACT and SALSA significantly improve the in-

teroperability and integration of Satellite and Terrestrial LoRaWAN networks, ensuring the required performance. These contributions provide practical solutions to existing limitations, paving the way for the effective implementation of integrated IoT-satellite technology agnostic to the application.

1.4 Thesis Structure

The remaining of the thesis is structured as follows. In Chapter 2, an introduction to the existing IoT technologies and their characteristics has been provided, by comparing them, and selecting the best fitting IoT protocol.

In Chapter 3, a brief summary of satellites and their different orbits is given to have better understanding of the satellite environment. This includes also the challenges and benefits of the Satellite IoT technologies, in correspondences to their use cases and market challenges. In Chapters 4 and 5, the REACT and SALSA methods are described in detail, including their design and theoretical foundation. In Chapter 6, SALSA simulator and LORSAT testbed have been presented and described, as the tools to evaluate the REACT and SALSA methods in Chapter 7.

Finally, Chapter 8 concludes the thesis, summarizing the problems, the solutions, and their main innovative features, and outlining future work and application scenarios of the proposed solutions.

2 | Internet of Things Technologies

Low Power Wide Area Networks (LPWAN) encompass a category of wireless communication technologies specifically designed to enable long-distance data transmission while minimizing power consumption for connected devices. Protocols such as LoRaWAN, NB-IoT, and Sigfox exemplify LPWAN technologies, offering solutions particularly suited for scenarios that demand prolonged battery life, broad-range connectivity, and support for large numbers of devices across vast geographic areas [11].

Narrowband Internet of Things (**NB-IoT**) [12] is a cellular LPWAN technology operating on licensed spectrum, designed to enable efficient communication for a multitude of low-power devices. It offers good coverage, deep penetration, and reliable connectivity, making it ideal for applications such as smart metering, asset tracking, and smart city solutions.

Conversely, **LoRaWAN** [9] is a non-cellular LPWAN technology that operates in unlicensed spectrum, renowned for its long-range communication capabilities and low power consumption. LoRaWAN is particularly suitable for applications requiring long battery life and extended range, such as smart agriculture, environmental monitoring, and industrial automation.

In recent years, most efforts and attention in the LPWAN space have been directed towards LoRa/LoRaWAN and NB-IoT [13]. Therefore, the concentration is on these technologies, which lead LPWAN deployments and innovations. Their distinct characteristics and capabilities make them well-suited for a wide range of IoT applications, and their integration into 6G networks positions them as key components in the advancement of next-generation wireless communication technologies.

The following sections will provide detailed descriptions of these two prominent LP-WAN technologies.

2.1 LoRaWAN

LoRaWAN is a LPWAN communication protocol designed to support IoT applications, ensuring long-range connectivity with minimal power consumption. The protocol builds upon LoRa, a proprietary radio communication technology initially developed by Cycleo, a company based in Grenoble, France, and patented in 2014. Following its development, Cycleo was acquired by Semtech, which has since continued to advance LoRa technology.

LoRaWAN, recognized as an official standard by the International Telecommunication Union (ITU) under ITU-T Y.4480, defines the network architecture and communication protocol for LPWAN applications. The ongoing development of the LoRaWAN protocol is managed by the LoRa Alliance, a global non-profit organization that includes Semtech as a founding member. Together, LoRa and LoRaWAN enable efficient, long-range wireless communication for battery-powered devices in regional, national, and global networks. The protocol is optimized for key IoT requirements such as bi-directional communication, end-to-end security, mobility, and localization, offering data rates between 0.3 kbit/s and 50 kbit/s per channel.

2.1.1. LoRaWAN Network Architecture

LoRaWAN operates as a star-of-stars network, facilitating end-to-end bi-directional communication between LoRaWAN end devices and LoRaWAN servers. The system architecture, illustrated in Fig. 2.1, comprises three primary components: (i) devices, (ii) gateways, and (iii) network servers.

End Devices (EDs) are devices equipped with sensing and actuating capabilities. They sense the physical environment, collect data (e.g., temperature, humidity), and send it through wireless LoRa links to the gateways. EDs cannot communicate directly with each other because the standard only supports a single-hop network topology. Sensing capabilities, computing power, and battery life are key metrics for selecting the most suitable ED for a specific application.

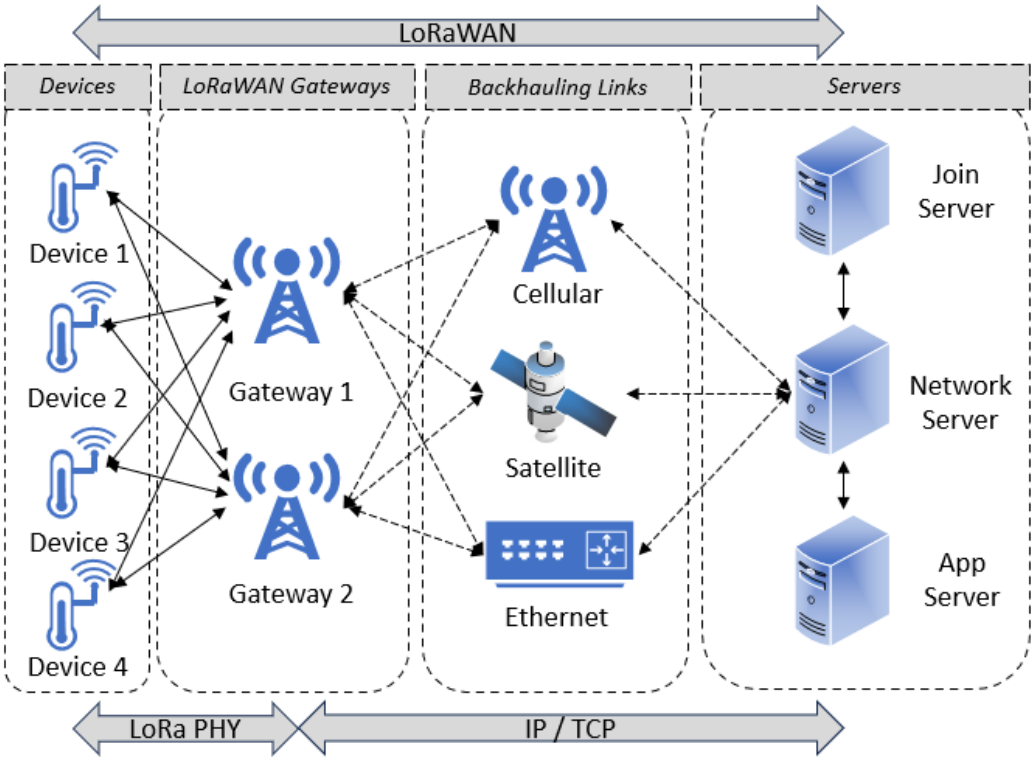


Figure 2.1: LoRaWAN Network Architecture.

Multiple **LoRaWAN Gateways (LGWs)** connect to a **LoRaWAN Network Server (LNS)** via a reliable backhaul link (Ethernet, cellular 4G, satellite, etc.). Each LGW acts as a network packet forwarder between the EDs and the LNS. Consequently, it cannot decode data frames received over LoRa links but encapsulates them into IP packets addressed to the LNS. Depending on system design criteria, LGWs can be fixed or mobile (e.g., mounted on drones or LEO satellites).

The system architecture includes a **LNS** that manages the **LGWs** and the entire system and handles the LoRaWAN protocol management within the network. The LNS collects and decodes uplink traffic (generated by the EDs and forwarded by the LGWs) and sends downlink packets and Medium Access Control (MAC) commands to the EDs through the LGWs. Various LNSs are available, with the open-source Chirpstack [14] server being one of the most adopted for private deployments. In nearly every country, there is at least one TelCo Operator offering a public LNS (and LPWAN coverage with their own LGWs). The Things Network [15] and LorIoT [16] are public and collaborative

Network Operators with freemium pricing plans, providing basic products and services free of charge while charging for premium features.

Finally, the LNS connects with several LoRaWAN Application Servers that give end users access to the entire system. These LNSs have interfaces allowing easy connection with Application Servers to collect, store, and visualize IoT data. Examples of well-known public Application Servers include Cayenne, Amazon Cloud Services, Things-Board, Ubidots, and etc.

In the following sections, the LoRa PHY protocol and the LoRaWAN MAC protocol (Fig. 2.2) are described, which regulate communication among the different components of the LoRaWAN network architecture.

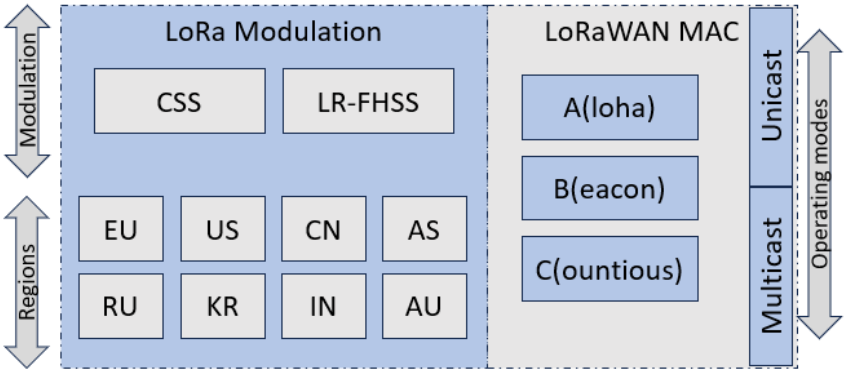


Figure 2.2: LoRa PHY and LoRaWAN MAC layers.

2.1.2. LoRa PHY Protocol

The Long Range (LoRa) technology, proprietary of Semtech Corporation, defines the specification of the Physical (PHY) Layer. LoRa is a Spread Spectrum modulation technique stem from the Chirp Spread Spectrum (CSS) concept. CSS is a wide-band modulation technique where the carrier wave varies cyclically and linearly over a time, between two values $f_{min} = -\frac{BW}{2}$ and $f_{max} = +\frac{BW}{2}$, defined by the signal Bandwidth, BW.

The resulting signal is called chirp. When the frequency variation increases, it is referred as up-chirp, either wise as down-chirp. LoRa modulation adopts up-chirp most of the time; therefore, this can be considered as its standard behavior, unless differently indicated.

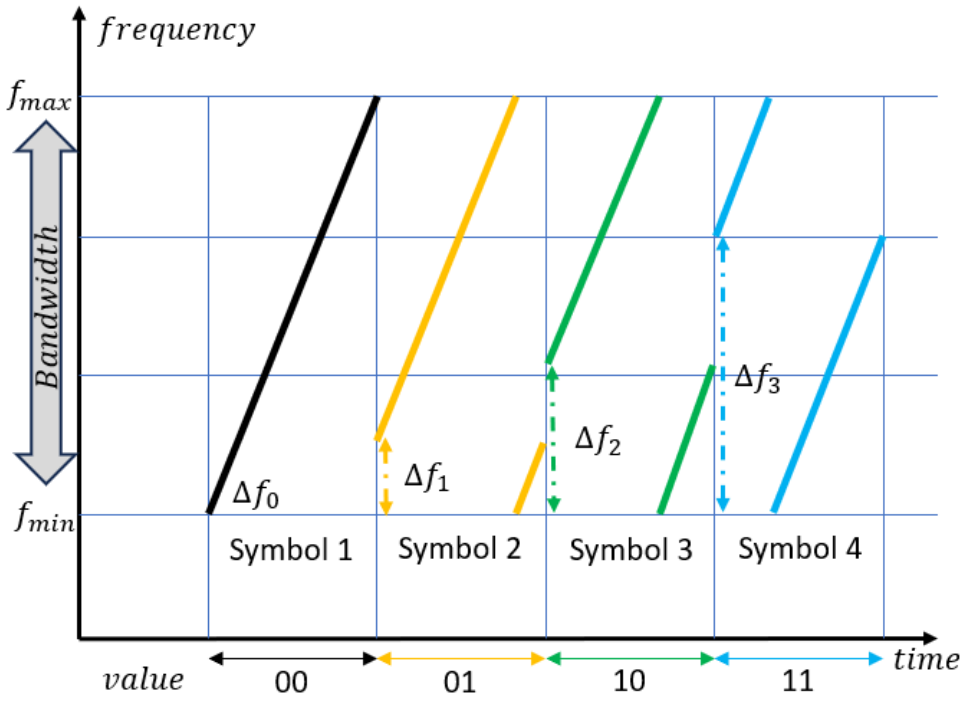


Figure 2.3: CSS Modulation. Example of un-modulated signal ($\Delta f = 0$), and modulated signal ($\Delta f > 0$).

A chirp, defined by its initial and final frequency (f_i ; f_f) does not necessarily start at the lowest value of the bandwidth (f_{min}). Actually, f_i can be any of the different frequencies within the bandwidth, and can be defined as $f_{min} + \Delta f$. The special case with $\Delta f = 0$, then $f_i = f_{min}$ and $f_f = f_{max}$ corresponds to unmodulated signal. Different values of Δf allow representing different values of the signal. When $\Delta f > 0$, the carrier wave starts the modulation at a given frequency $f_i = f_{min} + \Delta f$ and increases until it reaches f_{max}. Then, it wraps down to f_{min}, and continues increasing again until it completes a loop, by reaching the initial value $f_f = f_i$.

Fixed a number N of frequency shift Δf, with LoRa modulation it is possible to represent N_S different symbols. The number of bits per symbol, needed to identify the specific value represented by the signal, is called *Spreading Factor*, SF. According to the standard, the SF values can be selected from 7 to 12. This translates in the possibility of transmitting from 128 to 4096 bits of information per symbol.

$$N_S = 2^{SF} \tag{2.1}$$

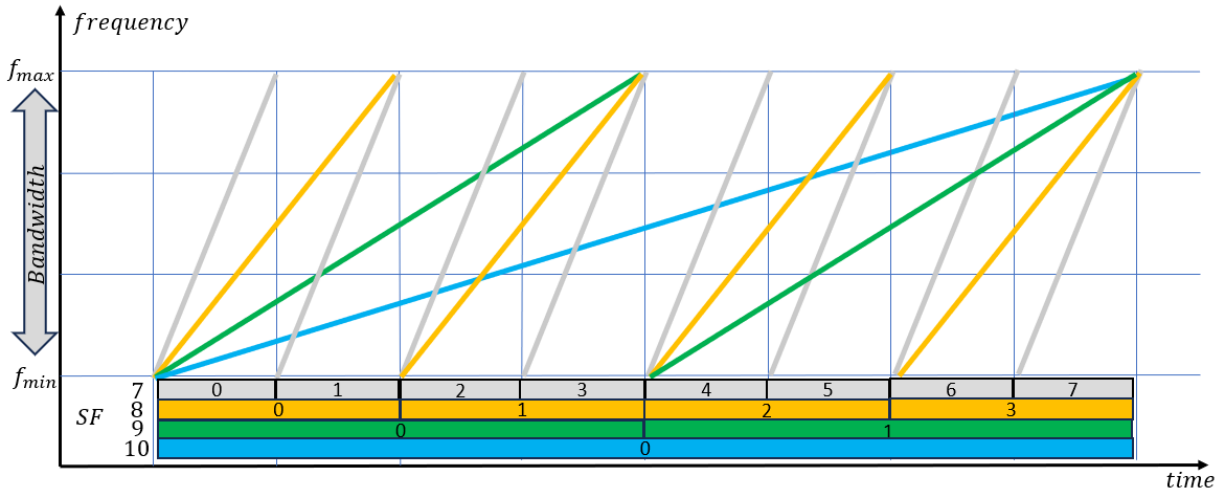


Figure 2.4: Relationship between SF, symbol rate, and ToA.

The SF also determines the rate at which the carrier wave changes from f_i to f_f over time. This frequency-change rate is defined *Chirp Rate*, or *Symbol Rate*, R_s.

$$R_s = \frac{BW}{2^{SF}} \tag{2.2}$$

Since the SF also represents the number of bits per symbol, the Symbol Rate translates in the following *Bit Rate*, R_b.

$$R_b = SF \times \frac{BW}{2^{SF}} \tag{2.3}$$

Increasing the SF translates into a linear increment on the amount of bits represented by each symbol which would lead to increased bit rate. However, it will also decrease the Chirp rate exponentially, and the bit Rate consequently. In fact, the carrier wave will require more time to complete a loop from f_i to f_f to represent any symbol. This translates in a longer Time on Air (ToA) to transmit the signal, and higher energy consumption. A slower modulation is more resilient to interference, attenuation, and noise. Thus, it allows the receiver demodulating correctly the received signal, even from longer distances. The opposite is also true. Reducing the SF diminishes the amount of different values that a Symbol can represent. Thus, it would decrease the bit rate. On the contrary, it increases much more significantly the Symbol Rate, which implies higher bit rate, faster modulation, shorter ToA, and lower energy consumption. Also, a signal

with lower SF would be more affected by noise, attenuation, and interference. This makes a low SF more suitable for short distance and faster communication.

The Bit Rate R_b specifies the amount of *raw* bits that can be transmitted over time. Of those *raw* bits, some are used to represent informational bits, namely the *net* bits. While the remaining bits are kept for error detection and correction, the *redundant* bits. The ratio between *net* bits and *raw* bits is known as Coding Rate (CR).

$$CR = \frac{net_{bits}}{raw_{bits}} \tag{2.4}$$

For each 4 net bits, LoRa modulation allows using between 5 and 8 raw bits. The default CR proposed by the standard is $4:5$. For each 4 net bits, 1 $redundant$ bit is added resulting in 5 raw bits. However, LoRa supports more resilient transmissions by increasing the amount of $redundant$ bits, at the cost of reducing the net bit Rate.

2.1.3. LR-FHSS PHY Protocol

Standard LoRa PHY encounters reliability problems when numerous EDs are concentrated in a densely populated area. Additionally, its finite link budget restricts its effectiveness for long-distance communications, such as direct satellite-IoT connections. To address these issues, Semtech has developed an advanced LoRa PHY layer called Long-Range Frequency Hopping Spread Spectrum (**LR-FHSS**), also referred to as LoRa-Extended [17].

The LR-FHSS protocol utilizes the same channels as regular LoRa PHY but divides the LoRa bandwidth into several smaller sub-bands called hops. In LR-FHSS, ED and LGW synchronize using a pseudo-random sequence that designates the specific hops for data transmission. This sequence is shared via a header, with two or three replicas transmitted on randomly selected hops.

After synchronization, the ED transmits the payload, which is segmented into smaller portions known as fragments. Each fragment is sent over a different hop according to the initially exchanged pseudo-random sequence. This process results in a longer ToA for transmitting the same payload compared to LoRa PHY (see Figure 2.5), enhancing resilience to collisions. Consequently, signal detectability and the link budget improve.

It is important to note that LR-FHSS is applicable only to uplink transmissions, with downlink traffic continuing to follow the regular LoRa PHY.

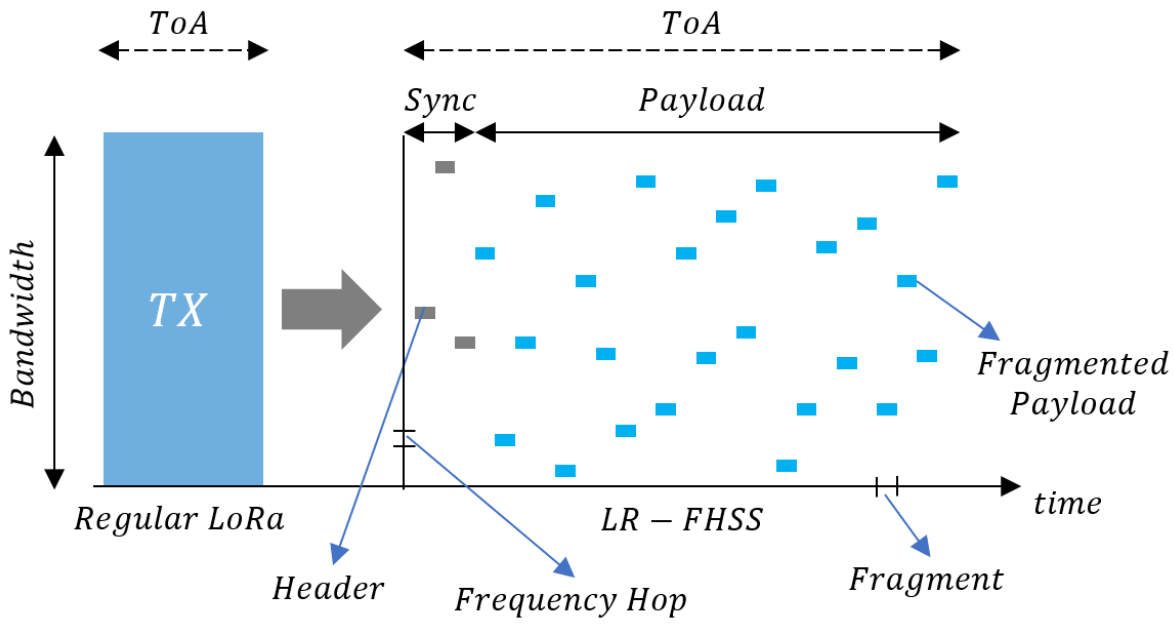


Figure 2.5: LoRa PHY vs LR-FHSS

2.1.4. LoRa Regional Regulations

LoRa is defined to operate in the unlicensed Industrial, Science, and Medicine (ISM) frequencies spectrum. ISM frequency spectrum is available worldwide, in different frequencies bands, e.g. Asia 443 MHz; Europe, Russia, India, parts of Africa 863-870 MHz; US, Latin-America 902-928 MHz; Australia 915-928 MHz; Canada 779-787 MHz, and China 470-510 MHz, 779-787 MHz.

For instance, in Europe, ETSI regulates for each sub-band of the 863-870 MHz band, the maximum energy allowed to irradiate in any direction, namely the Effective Isotropic Radiated Power (EIRP) and the amount of time (ToA) that the irradiation takes in a proportion of transmission, i.e, the duty cycle, DC (see Table 2.1).

These regulations have direct impact on the efficiency of the transmission. It imposes an upper-bound to the amount of time a transmission could take. The LNS can decide how to make best use of that time for a specific transmission, as much as it does not overpass the imposed limitation. It can increase the SF to make transmissions longer

Table 2.1: ETSI SUB BANDS 863-870 MHz

Name	Band (MHz)	Limitations
G	863 - 870	EIRP < 25 mW - DC $< 0.1\%$
G1	868 - 868.6	EIRP < 25 mW - DC $< 1\%$
G2	868.7 - 869.20	EIRP < 25 mW - DC $< 0.1\%$
G3	869.4 - 869.65	EIRP < 500 mW - DC $< 10\%$
G4	869.7 - 870	EIRP < 25 mW - DC $< 1\%$

and reachable from far, or decrease the SF to make transmissions shorter to save energy, or transmit more data during the same amount of time.

To ensure interoperability between transmitter and receiver, a set of communication parameters (better known as channel), should be configured and agreed. In the case of LoRa, the LoRa-Alliance has defined channels for each region, in a document called LoRaWAN Regional Parameters [10]. Those definitions include allocations of bandwidth, central frequency, and communication direction (uplink/downlink). The combination of SF and bandwidth is called Data Rate (**DR**) and it varies from 0 up to 5, depending on the region, and the frequency plan. Table 2.2 provides an example of DR in the EU 863-870 band. Network Operators might define their own set of channels and regulations, at the condition to be always compliant with local authorities.

Table 2.2: Data Rate in EU863-870 band

Data Rate	Configuration	PHY bit rate [bit/s]
0	SF 12 BW 125 kHz	250
1	SF 11 BW 125 kHz	440
2	SF 10 BW 125 kHz	980
3	SF 9 BW 125 kHz	1760
4	SF 8 BW 125 kHz	3125
5	SF 7 BW 125 kHz	5470
6	SF 7 BW 250 kHz	11000

2.1.5. LoRa Frame

Fig. 2.6 illustrates the regular LoRaWAN Uplink Frame format, transmitted by EDs to LGWs in their transmission range [9]. The Uplink frame includes: (i) a `Preamble`, (ii) a Packet Header `PHDR`, (iii) a Packet Header error Detector `PHDR_CRC`, (iv) a PHY Payload, and (v) a Data error detector `CRC`. Downlink Frames sent by the LNS to the ED differentiates from the Uplink Frames for not including a CRC field. The LoRaWAN MAC Frame is encapsulated into the PHY Payload, and is composed of: a Message Header, `MHDR`, a MAC Payload, and a Message Integrity Code, `MIC`.

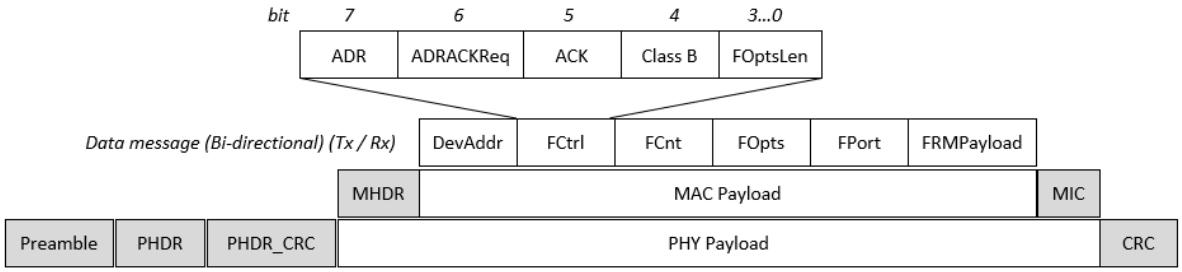


Figure 2.6: LoRa PHY and LoRaWAN MAC Uplink Frame Format.

When an ED wants to join a LoRaWAN network, first has to send a *join request*, and then wait for the *join accept* message. The data message format for such bi-directional communication is illustrated in Fig. 2.6. It includes six fields: (i) the Device Address in the network, `DevAddr`, (ii) the Frame Control byte, `FCtrl`, (iii) the 2-octets Frame counter, `FCnt`, (iv) a Frame Optional field, `FOpts`, (v) the Port Field `FPort` specifying the application port number, and (vi) the Frame Payload, `FRMPayload`. While all the details related to those fields can be found in the standard [18], hereafter only the field `FCtrl` is described, being relevant for the set up of the LoRaWAN Communication modes.

The `FCtrl` field is 1 byte long, and it includes five sub-fields in uplink: (i) `ADR` specifying if the Adaptive Data Rate Algorithm is supported, (ii) `ADRRACKReq` Acknowledge about the minimum Data Rate requested by the ED, (iii) `ACK` set for the confirmed messages, (iv) `Class B`, a Reserved for Future Usage (`RFU`) field used for switching between class A and B, and (v) `FOptsLen` specifying the actual length of `FOpts` field.

To send a MAC command, such as a joint request message, or a message for configuring the communication parameters, the ED can use two different approaches: (i) send it as a piggyback in the `FOpts` without encryption or (ii) send it in the `FRMPayload` as encrypted command with `FPort = 0`. In the first case, the maximum length is equal to 15 octets, while in the second case it is equal to the `FRMPayload` size. A Command Identifier, `CID`, allows to identify the command type. Requests and related Answers are matched using the same `CID` number. These MAC commands can be utilized to configure the ED remotely, such as class selection, and internal clock.

In other side, the LR-FHSS implements several modifications on the `preamble` and `Header` side of the uplink packets. As depicted in the Figure. 2.5, there are `SyncWord`, `Header`, and `HeaderCRC` instead of the preamble and header of the regular LoRa packet. It is worthy to mention that LR-FHSS has been defined only by uplink communications, and the downlink packet will remain same as regular format of LoRa PHY packets.

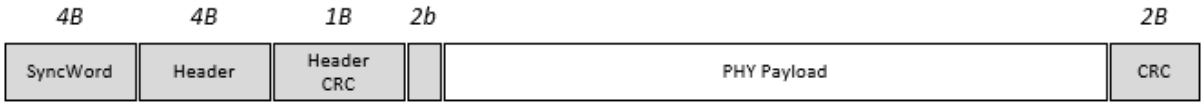


Figure 2.7: LR-FHSS Packet Format.

2.1.6. LoRaWAN MAC Protocol

LoRaWAN is based on a (**MAC**) layer communication, introduced by the LoRa Alliance [9] to manage the end to end network. LoRaWAN adopts a simple channel management scheme, based on pure ALOHA random access [19], which ensures low power consumption and low complexity, resulting in a cost-effective solution.

As shown in Fig. 2.8, LoRaWAN defines three main channel access strategies, one mandatory (Class A), and two optional (Class B and Class C), with the aim of meeting the requirements of different application scenarios.

Class A(loha). Defined in the standard as mandatory, it is implemented by all the LoRaWAN EDs, according to the protocol specifications [9]. It adopts simple ALOHA scheme resulting in the simplest and most energy efficient mode among the three classes. The ED can sleep most of the time and wake up only when it has some data

ready to transmit. The duration of the uplink transmission window depends on the frame size especially the user payload, and the LoRa PHY parameters.

The main drawback of the Class A consists in its asynchronous configuration that introduces high collision probability and random downlink (DL) availability. The downlink communication is triggered by the ED that opens two short downlink receive windows, after having transmitted a frame.

Typically the first receive window, RX_1, is opened for $1\ s$ (D_{RX_1}), and it can last for maximum $30\ ms$ by modifying the configuration, enough for receiving a preamble of a LoRa frame. If a frame is well received during RX_1, then the node goes immediately after in sleep mode. If the ED does not hear any preamble, then it opens a second DL receive window, RX_2, after $2\ s$ from the end of the UL window (D_{RX_2}). Despite its default value, the duration of D_{RX_1} can be modified by the LNS, using a MAC command through a downlink packet.

Class B(eacon): As an improvement of Class A, Class B presents predictable DL windows with the ability of reducing the probability of collisions between the data frames. While the UL phase of the Class B remains unchanged as UL of the Class A, more receive windows to download data and settings (i.e., MAC commands) from the LGWs can be opened. A Class B device initially operates in mode A, till it is switched to mode B, via application layer of the ED or via exchange of MAC commands from the application server.

When operating in Class A, the sub-field `RFU` of the `FCtrl` field in the UL frame is set to 0; for operating in Class B, it must be set to 1. When receiving an UL frame with `FCtrl (RFU=1)`, the LGW has to send *beacons* to the ED for synchronising the UL and DL timing. Once the internal clock of EDs is synchronized with the base clock of the GWs, the GWs transmit periodically beacons, every *Beacon Period*. The latter is set by default to 128 seconds. When an ED does not receive any beacon for 120 minutes, the ED must go back to Class A, and inform the LGW and the network of the class change by setting `FCtrl (RFU = 0)` in the next UL frame.

As shown in the Fig. 2.9, the time elapsing between two beacons includes a *Beacon Reserved* period, a *Beacon Window*, and a *Beacon Guard* period. The *Beacon Reserved* period ensures the transmission of the beacon before opening the beacon window. It has a fixed

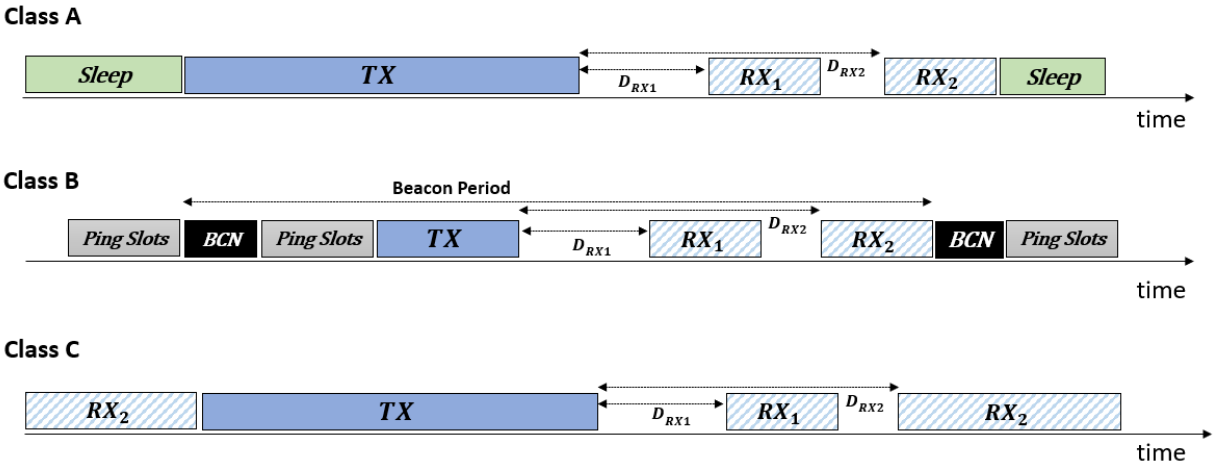


Figure 2.8: LoRaWAN End device operating classes A, B and C. Uplink and Downlink configurations.

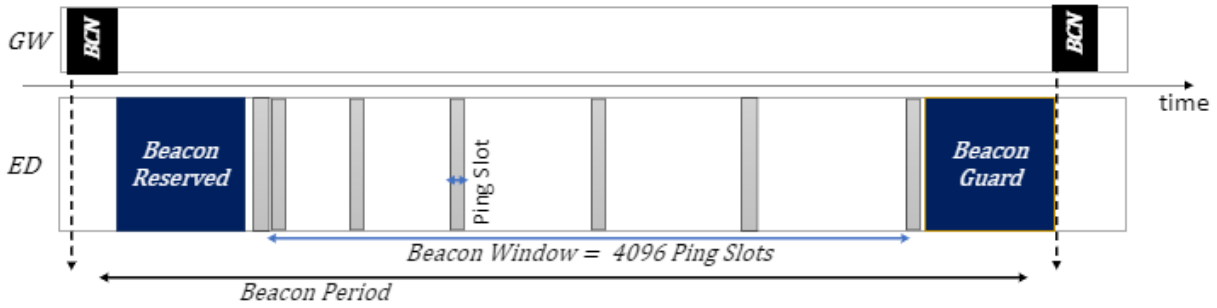


Figure 2.9: Class B downlink configuration

duration equal to 2.120 seconds. Analogously, the *Beacon Guard* period, 3 seconds long, ensures the beacon window is closed before the transmission of the next beacon. The *Beacon Window*, having a duration of 122.880 seconds is divided into ping slots, with a maximum number of 4096 slots. Each ping slot has a duration of about 30 milliseconds. It has to be noticed that not all of the ping slots can be used by the ED in every Beacon Period. By using a slot randomization method, the available number of operational ping slots is limited to the maximum number of 128 slots. This allow reducing the systematic collisions and capture effect. In such slot randomization method, a random key and a *Ping Offset* will be produced at each beacon by ED and LNS using the AES encryption technique. By using the generated key, the set of reserved ping slots are known for server, LGW and the ED.

EDs may miss the beacons sent from the LGW for coverage and interference issues. In this case, the *beacon-less* operation mode is activated: the ED goes into internal timing

reference when it does not receive any beacon. At this state, the ED adds drift to its internal clocking reference to gradually widen ping slots duration to have more time to have the chance of receiving a frame preamble. If this situation persists, the ED will switches to a Class A mode after 120 minutes since the last received beacon. Then it will update the `Class B=0` in the `FCtrl` field of the MAC frame sent to the LNS.

Class B is a better choice for applications requesting confirmed messages, and or downlink traffic from the network/application server. By using appropriate scheduling techniques, it is possible to use the ping slots in efficient way, while reducing the increased energy consumption of such communication mode.

Class C(ontinuous): Unlike Class B, Class C does not implement beacons, but similarly to Class A, it opens DL receive windows, and keeps them opened till a new transmission is triggered. The ED never goes in sleep mode. Thus, this access scheme reduces the latency of DL frames, while in contrast it dramatically increases the energy consumption. Class C is suitable for applications requiring bi-directional communications, and and in particular timely actuating operations (through DL traffic sent by the LGW to the EDs). A Class C ED cannot switch to the Class B. Note that it is not necessary to update `FCtrl` byte for switching from Class A to Class C. This information is exchanged between the ED and the network/application server during the joint procedure.

2.2 NB-IoT Standard

NB-IoT is a LPWAN standard that has been standardized by the 3rd Generation Partnership Project (3GPP) as part of the broader LTE and 5G specifications. It was designed specifically to address the needs of IoT applications that require low data rates, wide coverage, and long battery life, making it a key player in cellular IoT communications.

The NB-IoT devices have **low power consumption** requirements, allowing for long battery life. These networks can cover large geographical areas, making it suitable for applications that require **wide coverage**. Also, NB-IoT provides a **secure communication**, inheriting from 5G standard, protecting data transmission from unauthorized access.

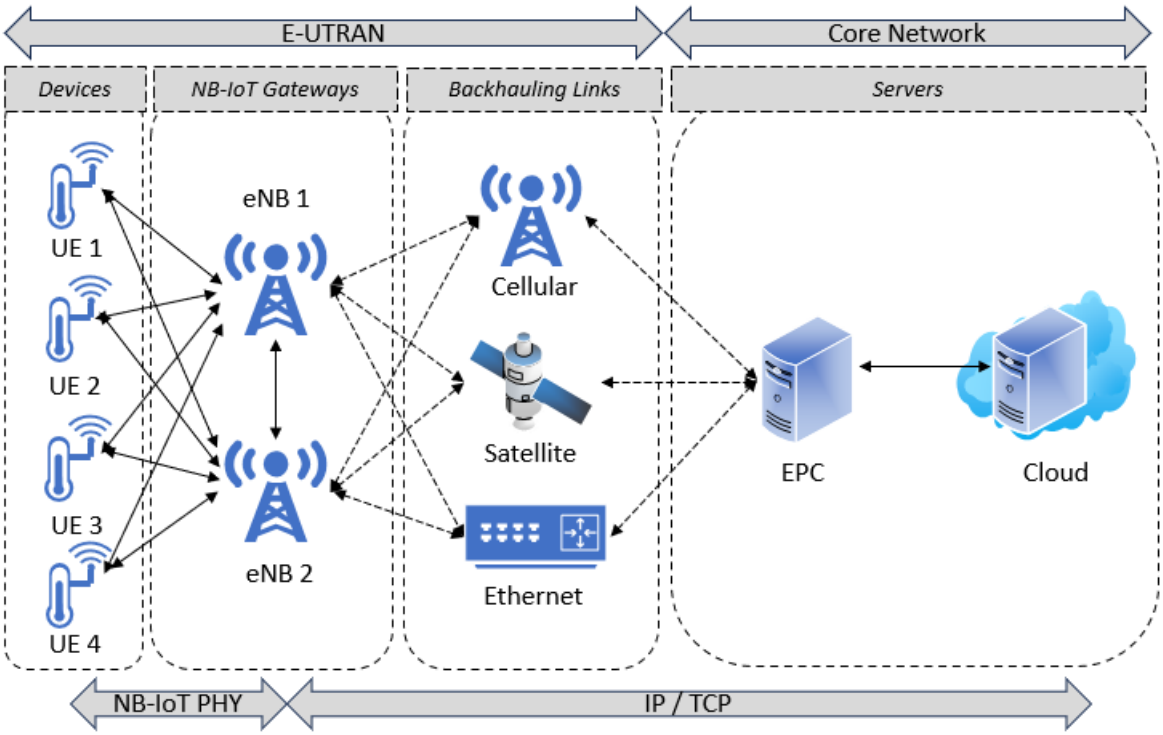


Figure 2.10: NB-IoT Network Architecture

On the other side, NB-IoT offers **limited bandwidth**, which may not be suitable for applications requiring high data rates. Complexity of NB-IoT makes it a technology that may have **higher latency** compared to other wireless technologies, affecting real-time applications. Like other Wireless technologies, NB-IoT networks may experience **interference**, affecting the reliability of communication, in dense urban areas.

2.2.1. Network Architecture

As shown in Figure. 2.10, the NB-IoT network architecture consists of three main components:

- **Devices**: These are the end devices or sensors that collect data and communicate with the network, also called as User Equipment (**UE**).
- NB-IoT **Base Station**: Also known as eNodeB (**eNB**), the base station connects the devices to the core network using the air interface.

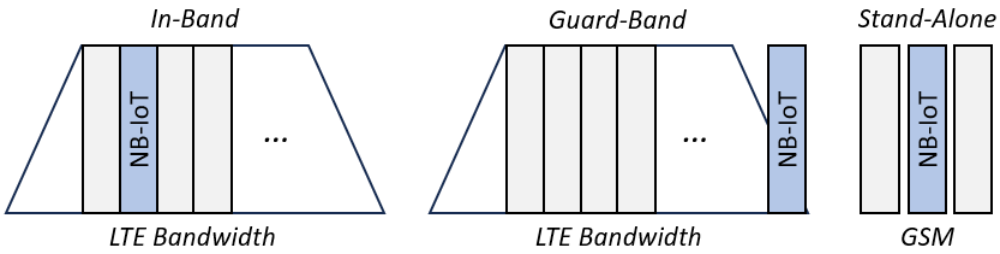


Figure 2.11: NB-IoT deployment modes

- **Core Network**: This includes the network components that manage the communication between the devices and the applications. The core network, namely the Evolved Packet Core (**EPC**), is responsible for transmitting the collected IoT data to the cloud platform for further processing and managing mobile devices [12].

Like LoRaWAN, the UEs communicate packets using a physical layer to the eNB. In the NB-IoT, the downlink packets are modulated using Orthogonal Frequency-Division Multiplexing (**OFDM**), which allows a better spectrum efficiency and reliability performance. In the uplink, the UEs receive the packets in a Single-Carrier Frequency Division Multiple Access (**SC-FDMA**) modulated signal. SC-FDMA is preferred for the uplink transmission in NB-IoT due to its power efficiency, and better performance in high-speed mobility scenarios.

NB-IoT occupies the 180 kHz frequency band, corresponding to a block of resources in the LTE bandwidth. NB-IoT supports three deployment modes, illustrated in Figure 2.11. The In-band mode occupies one of the Physical Resources Blocks (PRBs) of LTE. The Guard-band mode occupies only the protection band of LTE. In the stand-alone mode, NB-IoT can be deployed in any frequency spectrum, such as Global System for Mobile (GSM) frequency bands.

Figure 2.12 illustrates the structure of NB-IoT radio frame. Each radio frame has a duration of 10 ms and is divided into 10 subframes. Each subframe is made up of 2 slots. One subframe consists of 12 x 14 Resource Elements (REs) with a 15 kHz subcarrier for downlink (3.75 kHz or 15 kHz for uplink). 3GPP has defined several channels and signals with distinct functions for uplink and downlink, as described hereafter.

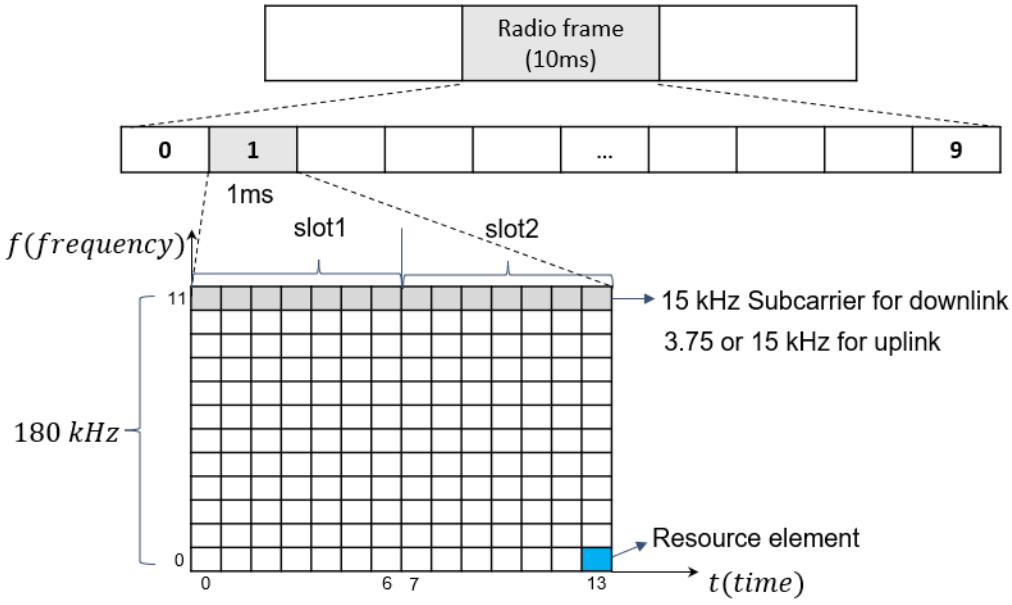


Figure 2.12: Radio Frame of NB-IoT

For the downlink, two synchronization signals, Narrowband Primary Synchronization Signal (NPSS) and Narrowband Secondary Synchronization Signal (NSSS), are transmitted in the subframes 5 and 9 to synchronize UE and eNB in time and frequency. The first subframe is Narrowband Physical Broadcast Channel (NPBCH), which is used to exchange critical system information such as deployment mode. The remaining subframes are occupied by the other two channels: Narrowband Physical Downlink Control Channel (NPDCCH) and Narrowband Physical Downlink Shared Channel (NPDSCH). The control channel contains information on uplink and downlink resource scheduling, allowing the UE to know when to receive or send messages. In the shared channel, downlink data and other system messages are exchanged. Note that control and shared channels may occupy several subframes, depending on the size of the message, the number of repetitions, etc.

Two different channels are defined for uplink transmission. The first connection attempt (Random access preamble) of the UE to the eNB is transmitted in Narrowband Physical Random Access Channel (NPRACH), where the collision happens. The uplink data is transmitted in Narrowband Physical Uplink Shared Channel (NPUSCH). Those resources are allocated by the upper layer avoiding a-priori any collision. Therefore, this channel is also known as a non-contention channel. Finally, NB-IoT supports two

transmission modes: Multiton and Singleton. Singleton uplink messages occupy only one subcarrier, while Multiton uplink messages occupy multiple (3, 6, 12) subcarriers. So multiple UEs can occupy the same channel, allowing more users to be connected simultaneously.

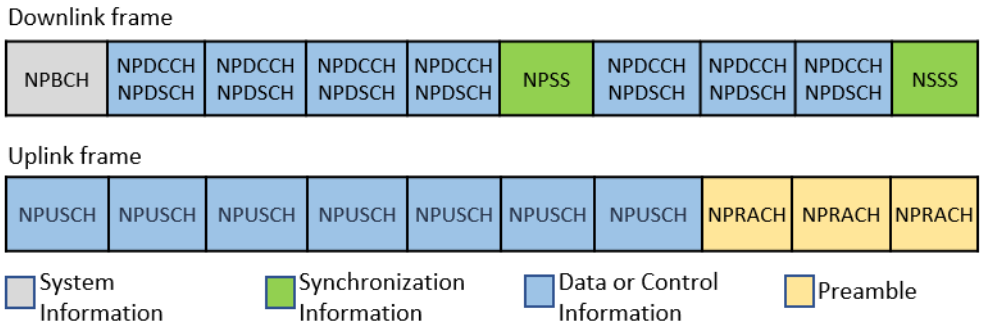


Figure 2.13: Downlink and uplink frames of NB-IoT

The UE of the NB-IoT must synchronize with the eNB by receiving the synchronization signals before connecting with the eNB. Note that in the LoRa PHY protocol, only Class B enables the synchronization between the ED and the GW using beacons.

When a UE is covered by more than one eNB simultaneously, it measures the received power and then selects the one with the best available coverage (best signal quality). In LoRa, the ED transmits to any LGWs in its coverage range. It is up to the LNS to select the best LGW for sending back downlink traffic. Moreover, while in LoRaWAN, the ED receives the configuration parameters from the LNS, in an NB-IoT network, the UE itself determines the Coverage Enhancement (CE) level according to its distance from the eNB and thus, chooses the number of repetitions of a message (2-1024 times). The higher the CE level (0-2), the higher the power consumption of the data transmission [20].

When the UE has a message to send or monitors paging, it will connect to the eNB as shown in figure 2.14. The random-access process will begin once the UE completes the synchronization with the eNB. A random-access preamble is sent to the eNB using the random-access channel (Msg1). The UE starts a timer and waits for a random-access response (Msg2). If no response is received, the UE will send a new preamble. After a successful reception of Msg2, Msg3 is sent from the UE to the eNB, which holds control information of radio resources, data volume, reconfiguration request, etc. Msg4 is the connection setup and the contention resolution, where the eNB accepts to establish

the connection with a UE. After receiving Msg4, the UE will enter the connected state from the idle state. Then the eNB and UE exchange messages for authentication and AS security configuration (Msg6-9). After that, UE sends its uplink data and receives downlink data. Finally, the eNB releases the connection if it detects inactivity from the UE (Msg10).

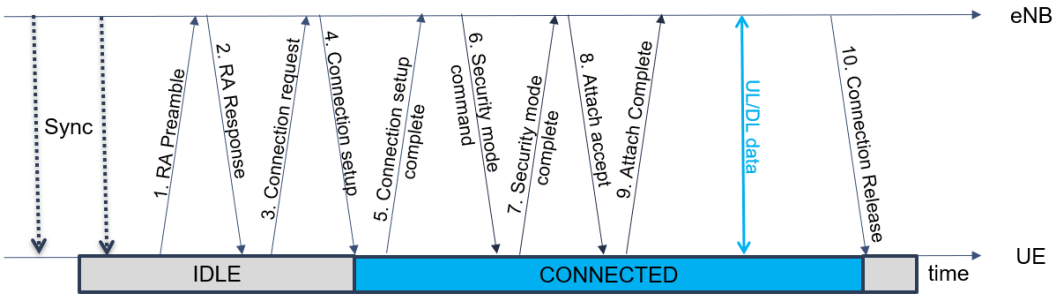


Figure 2.14: NB-IoT Workflow

NB-IoT defines two optimization methods for data transmission to reduce message exchange: the User Plane (UP) and the Control Plane (CP) optimization. The CP carries the signaling responsible for accessing the UE, allocating resources (e.g., messages exchanged after random access), etc.; the UP carries the user data.

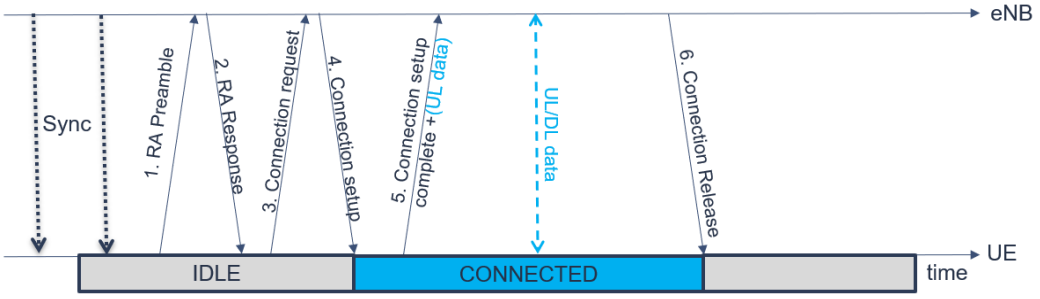


Figure 2.15: NB-IoT CP optimization

It must be noticed that for sending and receiving a few bytes of data, the signaling overhead consumed by the UE from the idle state to the connected state is much more significant than the data load. To make data transmission more efficient, two optimization schemes have been proposed: CP and UP optimization. With the CP optimization, small packets can be added to the control message (Msg5) and bypass the security configurations to improve the speed for transferring small data. This mode is insecure compared to other modes.

The UP optimization allows idle users to transfer data quickly through the *suspend and resume* process. After establishing the first connection, the user's information can be stored in the eNB. No Connection Release message is transmitted. When there is new data to transfer, the UE can soon recover the connection without re-establishing the security information.

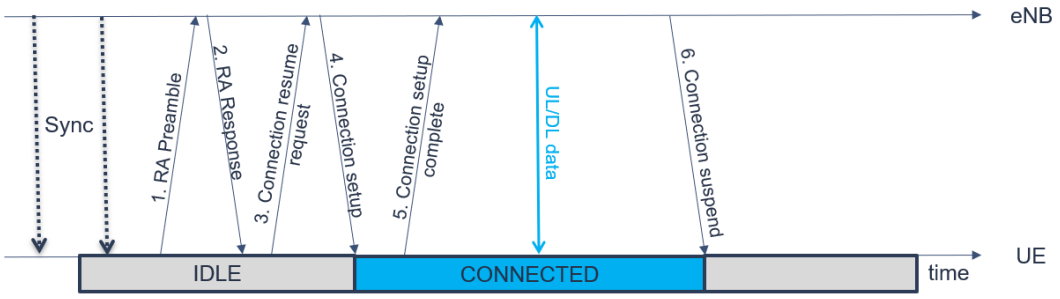


Figure 2.16: NB-IoT UP optimization

In Release 15, 3GPP defined the *Early Data Transmission* (EDT) mode to reduce UE energy consumption and message latency by reducing the number of transmissions [21]. Specified for both UP and CP optimization, the EDT can be used when the UE is in idle mode and has less than the maximum broadcast uplink data to send.

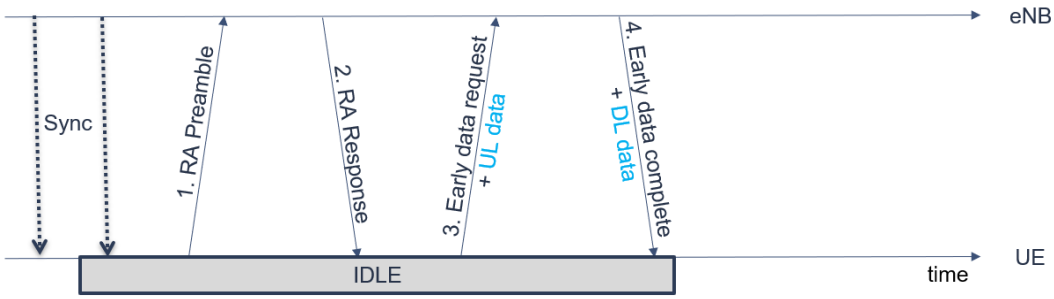


Figure 2.17: NB-IoT Early Data Transmission

In this mode, only four messages between the eNB and the UE are required to complete the data transmission because the data is sent during the random-access procedure. As shown in Figure 2.17, the data is included in Msg3. The method of encapsulating and transmitting uplink data is like the optimization of the CP. If the UE receives the Msg4 indicating that the procedure is terminated, it can go to the sleep state or stay in the idle state.

Table 2.3: Comparison between NB-IoT and LoRaWAN

Factor	**NB-IoT**	**LoRaWAN**
Energy Efficiency	Very Low Energy	Very Low Energy
Delay	Similar	Similar
Coverage	Good	Wide
Complexity	Complex	Simple
Frequency Band	Licensed	Unlicensed
Cost	Higher Infrastructure Cost	Lower Infrastructure Cost

Thanks to more complex synchronization and resource allocation techniques, NB-IoT can offer higher reliability compared to LoRa. This is paid with (i) longer transmission delays, which can be reduced using CP, UP optimization, and EDT; and (ii) higher energy consumption for the IoT device. LoRa, while being more energy-efficient, it suffers from high collision probability due to the random-access mechanism. Both reliability and throughput can be improved using resource allocation schemes, like TDMA approaches.

2.3 Comparison and IoT protocol selection

Both NB-IoT and LoRaWAN have specific features which make them well suited technology for different applications. As provided in the Table. 2.3, LoRa/LoRaWAN devices are known for their **Energy efficiency**, allowing for long battery life. NB-IoT devices, on the other hand, may consume more power, impacting the longevity of battery-powered devices. Also LoRa/LoRaWAN offers lower **Latency** compared to NB-IoT. This is crucial for applications where real-time data transmission is required. This lower latency happens mostly in the session setup procedure.

LoRa/LoRaWAN technology is generally more cost-effective compared to NB-IoT, thanks to it's simplicity. The infrastructure costs for deploying LoRa networks are lower, making it an attractive choice for cost-effective applications. Simplicity of LoRaWAN networks makes it easier to deploy, maintain and also more accessible for a wider range of applications compared to NB-IoT networks. Also operating in the **unlicensed** spectrum provides more flexibility and ease of deployment, which makes NB-IoT to be more

restrictive and costly. Bringing wider **coverage** makes the technology more suitable to choose, for the applications in the rural remote areas with limited power devices.

In comparing NB-IoT and LoRa/LoRaWAN, several key factors led to the selection of LoRa/LoRaWAN over NB-IoT for our deployment. One of the primary considerations is the ability to deploy a private operational network with LoRa/LoRaWAN, an advantage that NB-IoT, which typically relies on licensed spectrum and operator-managed infrastructure, does not offer. Furthermore, LoRa/LoRaWAN is based on an open standard, providing flexibility to modify and optimize the protocol to better suit specific use cases. This stands in contrast to NB-IoT, which operates on a more rigid framework controlled by telecommunications providers [22], [23]. The adaptability inherent in LoRa/LoRaWAN's open standard aligns well with dynamic application requirements and fosters innovation within the IoT ecosystem. In contrast, NB-IoT depends on licensed bands, which can increase operational costs and limit flexibility. Collectively, these factors—private network deployment, open standard customization, and the use of unlicensed bands—make LoRa/LoRaWAN a more suitable choice for our specific needs over NB-IoT.

3 | Satellite Networks

Nowadays, satellite networks are crucial for closing communication gaps and providing global connectivity. Gaining a solid understanding of satellite networks—covering their orbits, characteristics, applications, benefits, and limitations—is vital to appreciating their impact in the current digital age.

Satellites, essentially artificial objects placed into orbits around Earth, serve as relay stations for transmitting various types of data, including telecommunications, television signals, internet connectivity, and navigation services. These satellites orbit the Earth at different altitudes and in various patterns, catering to diverse communication needs. Broadly categorized into LEO, Medium Earth Orbit (MEO), and Geostationary Orbit (GEO), each satellite network type possesses distinct characteristics that define its functionality and applicability. These orbits are described in detail in the following section.

3.1 Satellite Orbit Design

3.1.1. LEO Orbit

LEO satellites orbit the Earth at altitudes ranging from approximately 160 kilometers to 2,000 kilometers above the surface. These satellites are characterized by their relatively low latency, making them suitable for applications requiring real-time data transmission, such as voice communication and remote sensing. LEO satellites are commonly used for global internet coverage, offering high-speed broadband services to urban and rural remote areas. LEO satellites take benefit of having low latency, high-speed data transmission, global coverage, and suitability for real-time applications. IoT devices require lower power to communicate directly with LEO satellites comparing to the other

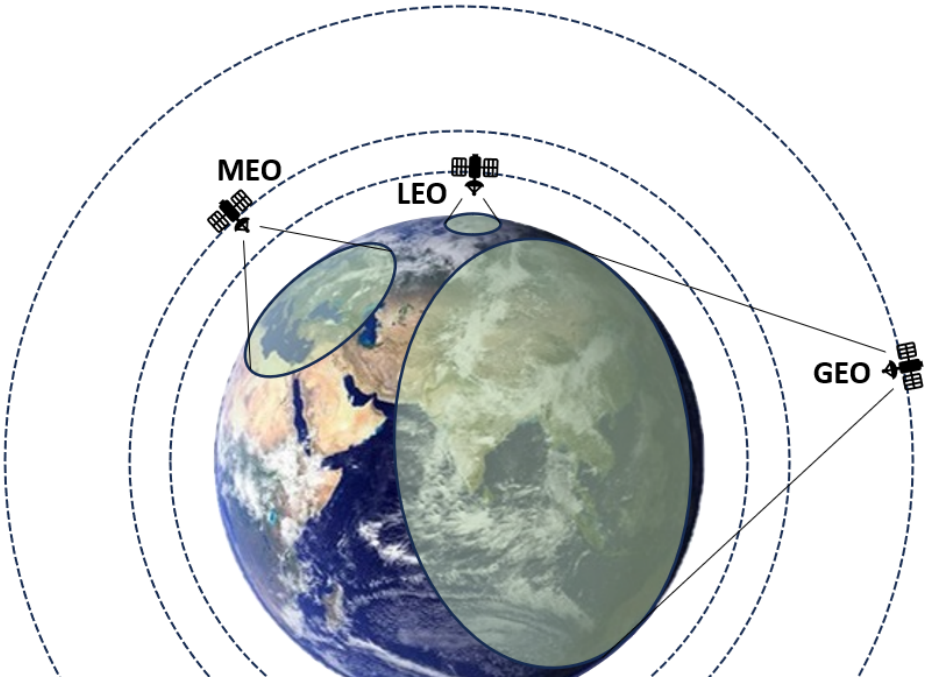


Figure 3.1: Satellite Orbits

available satellite orbits, thanks to their distance. In the other side, they have limited coverage per satellite, necessitating a larger number of satellites for complete global coverage; frequent orbital adjustments required due to atmospheric drag.

3.1.2. MEO Orbit

MEO satellites occupy orbits typically ranging from 2,000 kilometers to 35,786 kilometers above the Earth's surface. These satellites strike a balance between coverage area and latency, making them ideal for applications such as GPS navigation systems and satellite-based communication services. MEO satellites offer wider coverage compared to LEO satellites while still maintaining relatively low latency, making them suitable for global-scale communication networks. MEO satellites brings enhanced coverage compared to LEO, moderate latency suitable for a wide range of applications. They create higher latency compared to LEO, by having fewer satellites in orbit compared to LEO networks, resulting in potential coverage gaps.

3.1.3. GEO Orbit

GEO satellites are positioned at an altitude of approximately 35,786 kilometers above the Earth's equator. These satellites orbit the Earth at the same speed as the Earth's rotation, appearing stationary relative to a fixed point on the Earth's surface. GEO satellites are extensively used for telecommunications, broadcasting, and weather monitoring due to their stationary position and wide coverage area. They have a stable coverage over a fixed area, ideal for continuous communication and broadcasting services. But, they bring higher latency due to the greater distance from Earth, susceptibility to signal interference and atmospheric conditions, limited capacity compared to LEO and MEO networks.

Understanding the characteristics and functionalities of LEO, MEO, and GEO satellite networks provides a comprehensive insight into the diverse applications and capabilities of satellite-based communication systems. Whether facilitating global internet access, enabling precise navigation, or delivering real-time communication services, satellite networks continue to play a crucial role in shaping our interconnected world. These orbits are compared in the Table. 3.1.

Table 3.1: Comparison between different satellite orbits

Satellite Orbit	Visibility	Power Required	Common use cases
LEO	$\leq 5min$	Minimum	Imaginary, IoT, Military, SAR
MEO	$\leq 20min$	Medium	Positioning
GEO	Always	High	Media Broadcasting

LEO satellites are particularly well-suited for direct IoT satellite communications due to their low latency, allowing for real-time data transmission. However, their visibility and availability may be reduced during the day due to their closer proximity to the Earth's surface. Despite this limitation, LEO satellites are preferred for IoT applications as they require less power from IoT devices, compensating for their reduced visibility.

The IoT devices require the knowledge about the availability and visibility of the LEO satellites for direct communications. To accurately calculate the availability and visibility of LEO satellites for a fixed location, understanding their orbital parameters is essential. LEO satellites are characterized by six orbital elements, depicted in Figure. 3.2, namely:

- **Inclination (i)**: The angle between the orbital plane of the satellite and the equatorial plane of the Earth.
- **Longitude the Ascending Node (Ω)**: The angle measured from a reference direction to the point (Υ) where the satellite crosses the equatorial plane from south to north.
- **Eccentricity (e)**: A measure of the deviation of the satellite's orbit from a perfect circle.
- **Argument of Perigee (ω)**: The angle measured from the ascending node to the point of closest approach to the Earth.
- **True Anomaly (ν)**: The angle measured from the perigee to the current position of the satellite along its orbit.
- **Mean Motion (n)**: The average number of orbital revolutions completed by the satellite per day.

While these parameters define the satellite's trajectory accurately, calculating its position at any given time becomes complex, especially for the low-computation devices. To simplify this process and make it in a standard format, Two-Line Element Sets (TLEs) are commonly used. TLEs provide a concise mathematical description of the satellite's orbit in time using a set of parameters derived from observations. TLE relates the orbital elements to real-world clock, and makes it relative to device clock. Therefore, the device can calculate the availability and visibility of the satellite whenever required.

A TLE consists of two lines of data:

- **Line 1**: Satellite name, identification data, and time epoch.
- **Line 2**: Satellite Orbital elements

For example, the TLE for a specific satellite, ISS ZARYA, is provided in the Table. 3.2.

Line 1 provides the satellite name, epoch and identification (Launched at 1998, Unclassified). Line 2 contains the satellite number (25544), epoch data (89st day of 2024), and the orbital elements necessary for calculations.

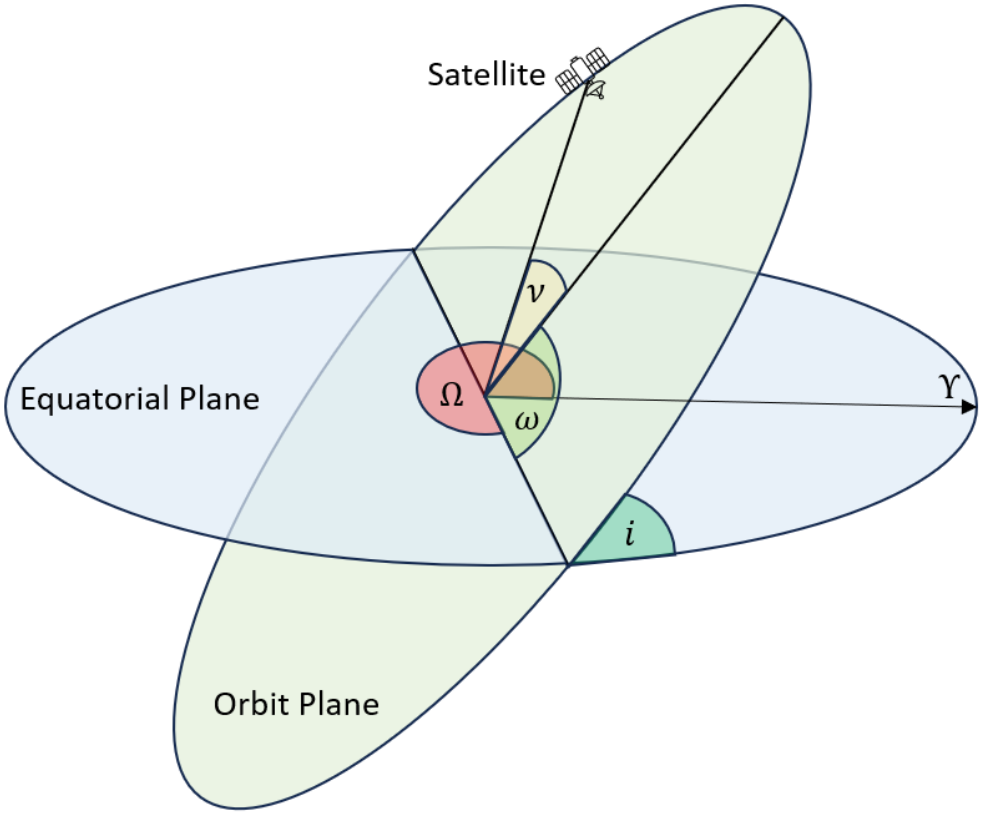


Figure 3.2: Satellite Orbital Elements

Table 3.2: Example TLE Data of ISS ZARYA

1	25544U	98067A	24089.245	.00025666	00000+0	46105-3	0	9993
2	25544	51.6418	351.7145	0004831	19.3054	123.6312	15.496	

By utilizing TLE data, precise satellite positions can be determined efficiently, enabling accurate predictions of visibility and availability for specific locations. Based on the application, updating frequency of the TLE can vary from several hours to several days. Calculating the satellite visibility for a fixed location requires also 4 more parameters. From the satellite point of view, the device has a location with a specific Latitude, and Longitude. The Longitude starts from Greenwich Mean Time (GMT) line and it can have a maximum value of ± 180, following the East $(+)$ or West $(-)$ direction. When the device tries to locate the satellite in the sky, Azimuth and Elevation of the satellite need to be considered. The Azimuth starts from 0° in North, heading to East 90°, South 180°, and West 270°, while the Elevation angle points to 0° in Horizon, and 90° in Zenith.

Achieving a higher elevation angle generally results in better communication between a satellite and an IoT device, as it reduces atmospheric interference and ground obstructions, ensuring a stronger and more reliable signal. However, as the elevation angle increases, the satellite's visibility to the IoT device decreases, limiting the communication window and coverage area. This creates a tradeoff between signal quality and network availability. In practice, an elevation angle of around 30 degrees strikes a good balance, providing adequate signal strength while maintaining sufficient visibility for reliable connections. Nevertheless, depending on the specific use case, other elevation angles can be explored in simulations and real-world deployments to optimize network performance for different environments.

3.2 Satellite IoT Communication

3.2.1. Challenges and Potentials

As per any radio communication protocol, LoRa coverage mainly depends on transmission parameters: SF, transmission power, and bandwidth (details were discussed in Sec. 2.1.2.). According to the standard specifications, and the modulation technique, a LoRaWAN network could reach up to 20 Km of coverage [24]. Besides theory, the actual coverage is impacted by the environmental conditions where the network - in particular EDs and GW - are deployed. Several empirical studies have assessed the LoRaWAN coverage both in indoor and outdoor environment. In [25], the authors showed that the actual coverage for outdoor deployment can be less than 20 Km even when adopting the higher SF. In indoor environment, lower values of SF are preferable to have a robust system, able to face with interference due to multiple walls [26]. Instead, higher values of SF can ensure better coverage for outdoor EDs, over long-distance. In [27], the authors run a network drive test to evaluate the loss exponent and Signal to Noise Ratio (SNR) of a LoRaWAN network in the Jakarta. The results have shown that there would be blank spots in the coverage due to the signal absorption caused by the transmission path in urban areas. In [28] a coverage testing method is proposed to ensure the LoRaWAN network coverage in the dense urban areas. By means of simulation results they show that highest values of SF and CR can provide better coverage.

LoRaWAN is intrinsically a terrestrial network but it can be integrated with a NTN network, by installing the gateway on a aerial vehicle, or on a LEO satellite. Such NTN gateway can provide more footprint on the earth, and thus more coverage compared to a terrestrial gateway, as shown in Fig. 3.3. Few research work have already investigating the performance achievable in such hybrid network configuration. Valencia et al. [29] have measured atmospheric data from the radiosonde using sensors installed on a balloon about 10 km height. In [15], Lacuna Space used a balloon at 832 Km of height to assess the extended coverage of a LoRaWAN network. Colavolpe et. all [30] mathematically modeled and analysed the quality of the LoRa signal received from a gateway installed on a LEO satellite. They considered Doppler and capture effects, when receiving data from the EDs on the ground. Also, Fernandez [31] studied the impact of several physical constraints, on the quality of the communication between the LoRa EDs and the satellite gateway. The effect of ionospheric scintillation was considered as a fading parameter for the LoRa signal. By using a Software-Defined Radio (SDR) test-bed, they computed packet delivery ratio and received power, showing the decrease in throughput when ionospheric scintillation increases.

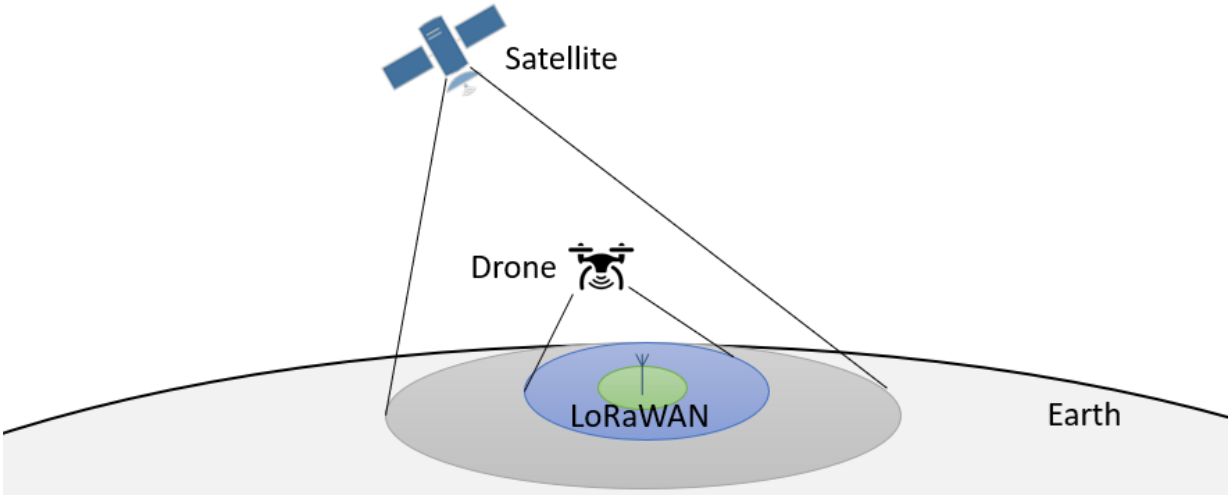


Figure 3.3: Coverage footprint of Terrestrial, aerial and satellite gateways for LoRaWAN networks.

Installing the LoRaWAN gateway on the LEO satellite brings several benefit, as described in Table 3.3. First, it largely increases the coverage range (thanks to the satellite footprint) and thus the number of supported EDs. Besides the initial high cost for building and launching the gateway integrated with the satellite, the NTN gateway is a cost-effective solution. It is robust, resilient, and does not request maintenance costs.

Table 3.3: Comparison Terrestrial vs NTN Gateway

Features	**Terrestrial**	**NTN**
Location	Fixed	Mobile
Coverage	5 - 40 Km	$\simeq$ 2000 Km (footprint)
HW cost	Low - Medium	High
Resilience	Limited	High (Against Disasters)
PHY effect	Reflection/Diffraction	Capture and Doppler Effect

Since LoRa was initially defined for fully terrestrial networks, the adoption of an aerial/satellite gateway asks for additional protocol optimization. For instance, improvements of the LoRa PHY protocol are needed to compensate budget link. Boquet et al. [17] have investigated on LR-FHSS, an extension of LoRa PHY: by using Frequency Hopping Spread Spectrum (FHSS). It reduces collisions, and thus increases scalability and Quality of Service (QoS). LR-FHSS improves the uplink transmission (i) by sending multiple headers which specify the exact list of frequencies that the ED is going to use in uplink depending on the data rate selection; and (ii) by transmitting continuously the data in random channels to avoid collision. No change was applied to the downlink. LR-FHSS is suitable for long range communication, such as from ground EDs to satellite gateways, thanks to the improved budget link. Optimisation of the LoRaWAN MAC protocol for communication over satellite are currently under development, and exploited both by academia (e.g. LORSAT project [32]) and industry (e.g. Lacuna Space).

3.2.2. Relevant Use Cases

Satellite IoT is essential due to its ability to provide connectivity in remote areas where constructing terrestrial networks like LPWAN is impractical [33]–[35]. The demand for high data rates from the increasing number of IoT devices necessitates innovative solutions, making LEO satellite communications a viable option for IoT applications such as smart metering, agricultural monitoring, environmental monitoring, and more [2]. In the following relevant use cases of the satellite IoT have been described.

Agriculture Monitoring: This application benefits from the field sensors to monitor crop health, soil moisture levels, and other factors that affect agricultural productivity. It helps farmers make informed decisions about irrigation, fertilization, and pest control.

Environmental Monitoring: Remote sensing is used to monitor changes in the environment such as deforestation, urban sprawl, and pollution levels. These data is crucial for conservation efforts and sustainable development, which they can be integrated from terrestrial sensor data collection to help refine the results [1].

Asset Tracking: Satellite based navigation technologies such as GPS are used to track the location and movement of assets such as vehicles, shipping containers, equipment, and also Wildlife Conservation. These data can be reached out via Satellite IoT networks in both urban and rural remote areas. This helps companies optimize their operations and improve security.

Disaster Management: In the case of disasters, the terrestrial networks might be unable to provide connectivity for monitoring the situation, or bring valuable data. Satellite IoT can establish an independent network from terrestrial networks, which it makes the system more robust comparing to the existing ones. Satellite IoT can help to manage natural disasters such as hurricanes, earthquakes, and wildfires [36]. This information is critical for emergency response planning and coordination.

Infrastructure Monitoring: Not all the vital infrastructures are located in the urban areas. Therefore, monitoring the condition of infrastructures such as roads, bridges, and buildings can be done with Satellite IoT technologies. This data helps identify maintenance needs and prevent failures.

Maritime Surveillance: Satellites and radar systems are used for maritime surveillance to monitor ship traffic, detect illegal fishing activities, and prevent piracy. This information is vital for maritime security and safety, which they can be provided also by Satellite IoT technologies [37].

Energy Management: Satellite IoT can be used to monitor energy resources such as solar and wind power. This data helps optimize energy production, improve efficiency, and reduce environmental impact.

Following the use cases mentioned above, the utilization of the Satellite link in the implementation is affected by the importance of the Age of Information (AoI). AoI factor

provides the validity of the data provided by the IoT devices by time [38]. In our target use cases, such as Agricultural and Environmental monitoring, the AoI has less impact on the performance of our system.

3.2.3. Market Analysis

Satellite IoT is a fast-growing industry that leverages satellite technology to enable connectivity for a wide range of IoT devices across the globe. Following the market report [39], here are some key points to consider in the market analysis for Satellite IoT:

Market Size and Growth: The market for Satellite IoT is expected to grow significantly in the coming years due to the increasing demand for global connectivity and the proliferation of IoT devices. This report shows that the market is poised for rapid expansion, for a compound annual growth rate (CAGR) of 14%, which is doubled in the market.

Market Trends: The current trends shaping the Satellite IoT market need to be analyzed, such as the adoption of LEO satellites, the integration of satellite communication with terrestrial networks, and the development of cost-effective IoT solutions.

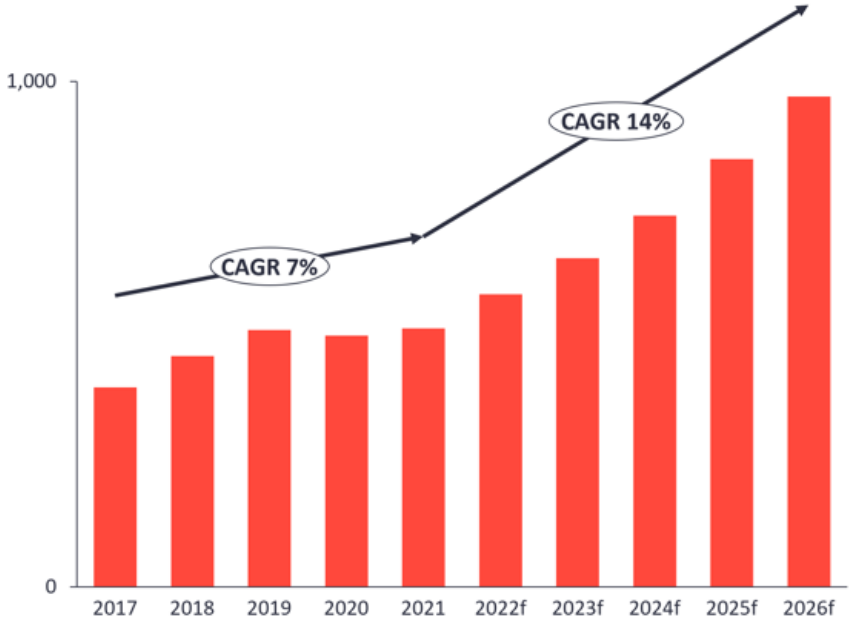


Figure 3.4: Satellite IoT Market in Million dollars [39]

[39] shows the interest of the market in the direct LEO satellite communication, while the traditional GEO communication and the combination of GEO and LEO satellites are in the considerations.

Market Drivers: The factors driving the growth of Satellite IoT, such as the need for ubiquitous connectivity in remote areas, the demand for real-time data transmission, and the increasing use of IoT in various industries like smart agriculture, environmental monitoring, and asset tracking.

Market Challenges: Like all other markets, there are several challenges for the Satellite IoT market, including high initial infrastructure costs, spectrum allocation issues, regulatory challenges, and competition from alternative connectivity technologies.

By conducting a comprehensive market analysis for Satellite IoT based on the above points, valuable insights can be gained into the current state of the market, key trends, opportunities for growth, and challenges that need to be addressed to succeed in this dynamic industry.

4 Time Synchronization in Satellite LoRaWAN networks

In the realm of LoRaWAN technology, the importance of time synchronization cannot be overstated. It serves as a critical component for enabling LoRaWAN devices to align themselves with the network infrastructure, thereby allowing them to effectively follow network commands, firmware updates, and other useful information. Moreover, time synchronization plays a pivotal role in the implementation of packet scheduling, ensuring efficient and reliable communication within the network. The necessity for precise time synchronization becomes even more pronounced when considering scenarios where a LoRaWAN gateway is deployed on a LEO satellite. In such setups, the synchronization of time becomes imperative for reliable communication and coordination between the Satellite LoRaWAN Gateway and the LoRaWAN devices connected to the network. This chapter delves into the fundamental aspects of time references and the various time synchronization methods prescribed by the standard LoRaWAN methods. By exploring these foundational concepts, there is the aim to pave the way for a deeper understanding of the challenges and opportunities associated with time synchronization in LoRaWAN networks.

The ED must have an *accurate time reference* on board, to be aware of the network time, and thus, be able of following the right transmission. In chapter, the *synchronization* issue is addressed that is being a pre-requisite for the performance of the entire end-to-end system. An ED can use three main time references: a *Global Navigation Satellite System* (GNSS) receiver, a *Real Time Clock* (RTC), and the *Internal Timer*. The GNSS module provides a high precision time reference, but it requests extra hardware components that implies higher cost, and more power consumption for the ED. The RTC also comes

with extra module and costs, including a battery for keeping the time reference alive. It can be a suitable choice for short duration of time, until the RTC module requires a time synchronization with the network to update its internal time reference. The Internal Timer is the cheapest and the most low-power option available in all the EDs. It is initiated when powering the ED on, but it is unaware of the real-world time.

The LoRaWAN protocol provides two time synchronization methods, one **Network-based** and the other **Application-based**. The network method uses the LoRaWAN MAC commands, and it is compatible with the EDs supporting at least the version 1.0.3 of the protocol [40]. The Application-based method takes advantage of the ED regular communication with the network, for asking a time reference. It adopts a *request and answer* message exchange in the user payload. Unlike the MAC-based method, it can be implemented also when the ED supports versions of the standard lower than v1.1 [41].

Low-power and low-cost LoRaWAN EDs may have a less-accurate oscillator crystal to create timestamps. The crystals can reach a drift accuracy not better than 10 PPM, which translates $864\ ms$ drift in one day [9]. The other elements of the network like servers and gateways might use different clock drift accuracy. It follows the need of having a heterogeneous compensation of the drift in the entire network.

In literature, few works tried to improve the LoRaWAN time synchronization, based on the MAC commands. In [42], a timestamp calibration is done by evaluating the accuracy of the previous time synchronization requests. Beltramelli et. al [43] modelled the clock errors using FM radio data system (FM-RDS), and timestamp-offset summations, where the EDs must wake up in a known synchronization period. Also, Ramirez et. al [44] introduced a model to gradually increase the synchronization accuracy in several requests by compensating the errors occurring in a regular LoRaWAN uplink (UL) message. They evaluated their model using a testbed, resulting in $10\mu s$ time synchronization accuracy. In all these works, the authors assumed that the EDs were equipped with a RTC and extra modules with specific versions of the LoRaWAN protocol to support their methods.

In this chapter, the proposed method is introduced, *Lo(R)aWAN diff(e)renti(a)l time syn(c)hroniza(t)ion (REACT)*, which stands as a novel approach towards addressing the time synchronization in LoRaWAN environments, applicable for both NTN and TN networks. Through a comprehensive exploration of time synchronization mechanisms and

the introduction of our innovative solution, this chapter aims to contribute significantly to the advancement of time synchronization techniques in LoRaWAN networks.

4.1 Standard Synchronization Methods

4.1.1. MAC Layer Method

The LoRaWAN provides the MAC layer time synchronization for the EDs supporting at least the version 1.0.3 of the standard. Since this time synchronization method benefits from the MAC payload, it does not reuse the space reserved for the user data. The procedure is simplified in the *request and answer* packets. The ED requests the time of the LNS by sending a blank MAC command, `DeviceTimeReq - 0X0D`, to the LNS (see Fig. 4.1). When the LNS receives the request, it stores the server time as the *answer* to put in the Downlink (DL). The LNS is not aware of the delays encountered over the backhauling link from the GW to the LNS (and reverse), and over the PHY layer. Therefore, there would be several milliseconds of uncertainty in the calculations. The LNS replies to the ED request with an answer, which contains the LNS timestamp (based on GPS time epoch) in the MAC payload. The answer is placed in the same MAC Command ID (CID), `DeviceTimeAns-0X0D`, with the field named `timeSinceGPSEpoch`. This value is divided into the *seconds* and *fractional Sub-Seconds* parts to present the time. When the ED receives this timestamp, it calculates the *ToA* of the DL packet sent from the LNS. Same as for the request, the ED is not aware of the backhauling delays between the LNS and the GW. The time reference received from the LNS, will be used by the ED as its new internal clock. According to the LoRaWAN specification [9], the MAC-based time synchronization method offers an accuracy less than $100\ ms$.

4.1.2. Application Layer Method

Another synchronisation approach was proposed in the protocol specification addendum [45] in the application layer, for any type of ED, supporting any version of the LoRaWAN standard. This method, unlike the MAC-based has the advantage of being backward compatible with any version of the standard, prior to v1.1. But it is less accurate. For this reason, in the addendum it is clearly stated that EDs operating in the

class B, or equipped with a GNSS receiver should not follow the app synchronization method.

Since the method is implemented in the application layer, the payload size reserved for the user data is reduced, compared to the standard specifications. The application method uses at least $5B$ of the payload size to send the ED internal time ($4B$) in seconds and parameters ($1B$) to the LNS. The payload has two important values: the ED internal time and the `TokenReq`. The `TokenReq` starting from 0 is a counter that increments each time the ED updates its internal time successfully. The APP server sends the answer by computing the difference between two clocks in seconds. This difference is tagged with the `TokenAns` to avoid sending a wrong answer to the ED. If both tokens have the same value, then the ED would accept this update. The LoRaWAN standard defines an acceptance accuracy less than $250\ ms$ for the application layer synchronization. Since the method provides timestamps in seconds, it results to be less accurate comparing the standard MAC method.

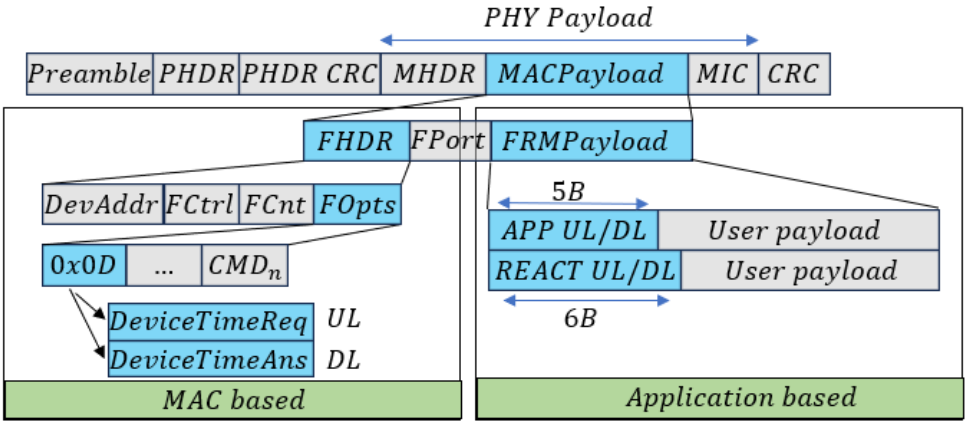


Figure 4.1: LoRaWAN Frame Format. All the time synchronization methods use the MAC payload to exchange time reference information. The MAC-based method uses the Frame Header (`FHDR`) and especially the `FOpts` field to send and receive MAC commands. In the Application layer methods, a part of the `FRMPayload` field is used: `5B` for the Standard app method, and `6B` for the REACT method.

4.2 REACT

In this section, the *REACT* method is presented to achieve time synchronization between EDs and network servers in LoRaWAN networks. It is independent from the LoRaWAN standard version, and it does not rely on time references that request additional

Table 4.1: Symbols and Definitions

Symbol	Definition
t_{REACT}	ED Internal REACT-based clock
t_M	ED Internal MAC-based clock
e_{REACT}	Accuracy of the REACT synchronization method
e_s	Compensated Time synchronization accuracy
T_L	Network RTC value at the time of packet arrival
T_A	Network RTC value at the time of packet arrival in the ED
ToA	Time on the Air of the LoRa packet
R	The Difference between ED and Network clocks
t_{EG}	Delay between the ED and the GW
t_{GL}	Delay between the GW and the LNS
t_a	ED internal timer value at the time of packet arrival
t_{pr}	Process time of the application layer to reply to the UL
t_r	The ED internal timer value at synchronization request
t_i	Initial value of the ED internal timer
t_l	The ED internal timer value at UL delivery in the LNS
t_d	The ED internal timer value at DL Creation in the LNS

hardware modules, like GNSS and RTC. The method is robust and accurate against any type of backhauling (terrestrial, and satellite). This method is designed for EDs operating in class A, which it can be used also in Class B and C. Because the LoRaWAN EDs behave like Class A while they are transmitting and UL. To avoid additional exchanges of specific synchronization messages, which could affect the energy consumption of the EDs, the method is implemented in the application layer, and it uses the regular data exchange in UL and DL.

To set up a time reference, the proposed REACT method takes benefit from the ED *internal timer*, available on all types of the EDs, implementing any LoRaWAN specification.

REACT is based on the same *request and answer* principle of the time synchronization method proposed by the standard at the application layer. As shown in the Fig. 4.1, REACT stores the time synchronization headers inside the `FRMPayload` as the `REACT UL/DL` within the $6B$. The REACT method can provide the timestamp with ms accu-

racy ($4B$ for seconds and $2B$ for milliseconds), while the standard application method performs in the s precision in $4B$. Note that also the standard application method may apply extra bytes to synchronize the ED. Both `REACT UL` and `REACT DL` stores $4B$ of the seconds and $2B$ of the sub-seconds of a number. The complete logic-diagram of the REACT method is shown in Fig. 4.2. Table 4.1 summarises the notation that will be used hereafter to describe the method.

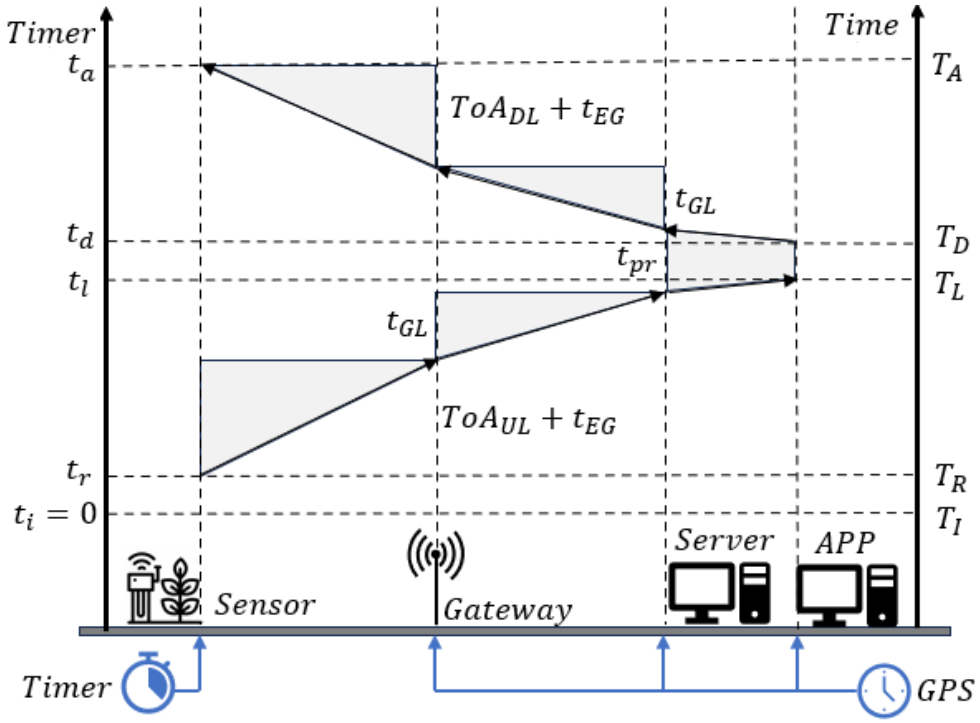


Figure 4.2: REACT method

When the ED is turned on for starting its operation, then the value of its timer initiates from $t_i = 0$. The ED would shape the packet at t_r for sending the *request* to the servers. t_r is the timer value of the ED at the time of synchronization request. The ED spends ToA_{UL} to create the LoRa signal and delivering it to the GW after a propagation delay, t_{EG}. When the GW receives the packet, it first encapsulates it in a TCP/UDP packet, and then it forwards it to the LNS along the backhauling link, after a delay t_{GL}. At the time T_L of the network, the packet is completely received and decoded in the APP server. At this stage, the APP server calculate the R as `REACT DL` to reply to the ED. R is the differential value required by the ED to shape the REACT time, as derived in Eq. 4.1.

$$R = T_L - t_r - ToA_{UL} \tag{4.1}$$

In the calculation of R, ToA_{UL} is considered to compensate the delay of the ED to shape the LoRa signal. Once calculated, the LNS puts the value R inside the user `FRMPayload` as `REACT DL` in the DL packet as shown in Fig. 4.1. The DL packet will reach the ED after t_{GL}, the delay between LNS and the GW, the t_{EG}, delay based on the distance between the GW and the ED, and the ToA_{DL}, the ToA on the DL. The internal timer of the ED is t_a in the time of packet arrival. The ED shall add the R to the t_a to rebuild its own time as t_{REACT}. The t_{REACT} would become the time reference for the ED for any time-based actions; such as internal events like measurements and actuation, and exact time for UL transmission, according to a schedule (e.g. SALSA[46]).

$$t_{REACT} = t_a + R \tag{4.2}$$

Eq. 4.3 provides the accuracy of the REACT method (expressed as error) by considering the difference between the T_A and t_{REACT}.

$$e_{REACT} = T_A - t_{REACT} \tag{4.3}$$

To derive the expected synchronization accuracy value, the Eq. 4.3 is expanded on t_{REACT} and T_A as follow:

$$e_{REACT} = T_A - t_a - R \tag{4.4}$$

$$t_a = ToA_{UL} + ToA_{DL} + 2 \times t_{EG} + 2 \times t_{GL} + t_{pr} + t_r \tag{4.5}$$

$$T_A = T_L + t_{pr} + t_{GL} + t_{EG} + ToA_{DL} \tag{4.6}$$

$$e_{REACT} = -(t_{EG} + t_{GL}) \tag{4.7}$$

Eq. 4.7 shows that the accuracy of REACT method depends mainly on the distance among the main elements of the network (and thus, the respective propagation delay); between the ED and the GW, t_{EG}, and between the GW and the LNS, t_{GL}. Since t_{EG} and t_{GL} are positive numbers, it results that the accuracy error is negative ($e_{REACT} \leq 0$). Therefore, the ED will consider the timestamps in advance comparing to the LNS.

5 Packet Scheduling in Satellite LoRaWAN networks

Scheduling is a crucial aspect of LoRaWAN network as it helps in optimizing the utilization of physical resources and managing the flow of data efficiently. In LoRaWAN networks, which are designed for low-power, wide-area applications, scheduling plays a key role in coordinating the communication between end devices and gateways. By implementing scheduling mechanisms, the network can ensure that devices has transmission chance at the right time to avoid collisions, and drops, and minimize energy consumption. Additionally, scheduling helps in improving network capacity, reducing latency, and enhancing overall network performance in LoRaWAN deployments.

5.1 Related Works

Duty cycle restrictions in sub-GHz ISM bands limit LoRa radios, challenging the use of Time-Slotted approaches with scarce physical resources [47]. Fehri et al. [48] propose a schedule-based schema for uplink communications in LoRaWAN, deterministically allocating time, channel, and SF for collision-free transmissions. SF are assigned to EDs based on predefined conditions, particularly for LoRaWAN class A devices. Triantafyllou et al. [49] emphasize beacon frame broadcasting by LoRaWAN gateways to synchronize communication with end devices. End devices use beacon information to select a SF for data transmission and choose a random time offset for packet transmission using the CSMA-CA algorithm. Clocks are synchronized by listening to the latest beacon from the LoRaWAN gateway. Junhee et al. [50] describe an algorithm allocating SF, frequency channels, and time slots for end devices to communicate with LoRaWAN

gateways. Upon receiving a scheduling request, the LNS allocates SF based on signal power and schedules frequency channels and time slots for simultaneous uplink transmissions on links with the same SF. Zorbas et al. [51] introduce a mobile LoRaWAN gateway mounted on Drone, then a scheduling algorithm based on Minimum Data Collection Time, where closer end devices use smaller SF for efficient data collection. Rafik et al. [52] discuss Class B scheduling with a Pseudo-Random Access Mechanism incorporating Orthogonality Awareness, utilizing Uplink Offset based on frequency, beacon time, and end device address. Abdelfadeel et al. [53] advocate for buffering data for scheduled time slot transmission rather than immediate transmission, with schedules based on the number of end devices, buffered data, and path loss to ensure synchronization. Mhatre et al. [54] propose a model optimally allocating transmission parameters like SF, channel, Transmission Power, and time slot to minimize ToA and energy consumption, using reinforcement learning to schedule end device transmission effectively and reduce interference. RS-LoRa, as discussed by Reynders et al. [55], allows end devices to autonomously select transmission parameters based on beacon information, reducing collisions and improving overall network performance. Haxhibeqiri et al. [56] present a model for fine-grained synchronization and scheduling in LoRaWAN networks, optimizing packet scheduling based on uplink traffic, clock drift accuracy, and re-synchronization frequency.

Recent studies have explored direct LoRa to satellite communication but lacked enhancements in network performance through scheduling techniques. Ullah et al. [57] modeled the PHY channel between end devices and the satellite LoRaWAN gateway, assuming ALOHA-like LoRa standard transmission. However, the literature reveals a gap in significant contributions to LoRaWAN scheduling with LEO satellites. This chapter addresses these challenges by providing a packet scheduling algorithm to improve reliability and scalability of Satellite LoRaWAN networks.

5.2 SALSA Scheduling Method

In the present chapter, *a Scheduling Algorithm for LoRa to LEO SAtellite (SALSA)* [46] is proposed. In this chapter, the algorithm is described in all its details, from the system assumptions, to the different access policies.

5.2.1. System Assumptions

In designing the algorithm, the LNS is assumed to know the location of the EDs and the availability time of the gateway (i.e., the visibility period of the satellite). The EDs may have been pre-registered in the network using Over The Air (OTA) registration while the gateway was still on the ground. All the EDs operate in Class A. Additionally, the LNS is assumed to send the scheduling table to each ED in the network via unicast during the RX windows. To account for deviations in the satellite's current orbit from its initial orbit, periodic scheduling updates should be sent directly to each device using unicast messages. The algorithm assumes that location of EDs is fixed. Thanks to the long distance of LEO satellite to the ED, the movement of an ED in a small area, like a farm, will not affect this mobility assumption. Note that the exchange of configuration parameters and scheduling tables is outside the scope of the this algorithm, which is why such assumptions are made.

The LNS is responsible for scheduling the uplink transmissions for each ED and assigning priorities. To ensure long-range transmission from the EDs to the satellite gateway, all EDs are assumed to transmit using $SF = 12$. Using a single SF value means that only one transmission can be scheduled at a time without causing a collision. According to the standard, a single uplink transmission occupies the channel for a ToA period. Adding two guard times is proposed, T_g, one before and one after the ToA, to account for potential synchronization issues between the ED and the mobile gateway (see Fig. 5.1). Thus, whenever the LNS schedules a transmission for an ED, it reserves the channel for that ED for a period equal to $T_r = ToA + 2 \times T_g$.

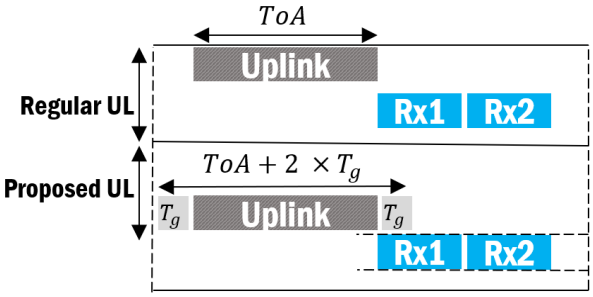


Figure 5.1: Time reserved T_r for each ED for an uplink transmission. $2 \times T_g$ have been introduced to mitigate uncertainties and inaccuracies of the synchronisation between the ED and the satellite gateway. The receive windows RX_1 and RX_2 are opened after the end of the uplink transmission, according to the standard.

5.3 Algorithm Policies

5.3.1. FCFS Policy

The LNS builds the scheduling tables for each of the EDs in the network based on their *discovering time* of the satellite. The latter depends on the satellite footprint, the satellite movement along its orbit, and the value of the elevation angle. As shown in Fig.5.2, while the satellite moves along its orbit, the satellite footprint also moves in time: therefore, different EDs will be progressively in the satellite visibility, one after the other, in different periods. To efficiently use the network resources and maximize the number of EDs that can communicate with the mobile gateway, the LNS gives priority to the EDs that are visited first by the satellite footprint. In the example in Fig.5.2, the ED_i will transmit before the ED_j being $r_i < r_j$, with r_i relative distance of ED_i from the satellite footprint.

Many nodes may be concentrated in the same area (i.e., dense area), one very close to another, and thus, while an ED is still transmitting, another one could enter the satellite footprint and attempt a transmission as well. To avoid such a scenario generating collisions, the concept of *slide* is introduced. The satellite footprint is divided into several slides. They can have variable time duration: in the simpler case, a slide is equal to the time needed for completing a single uplink transmission T_r. It is the case when there is no overlap between concurrent transmissions from several EDs because the EDs are one far away from another. On the contrary, when there is an overlap (i.e., a dense area with many EDs concentrated together), and an ED cannot finish a transmission before another ED enters the satellite coverage, the LNS allocates a larger slide. The slide's duration is set to $N_s \times T_r$, where N_s is the number of neighbor nodes within the identified dense area. The LNS defines the starting time T_{S_i} of the ED_i transmission. If the Start of the Satellite Visibility for that ED$_i$, T_{SSV_i} overlaps with an ongoing transmission from another ED_j, the ED$_i$ has to wait till T_{E_j} (ending time of ED_j transmission). Either wise, it can start transmitting as soon as it sees the satellite (see Algorithm 1). In the example in Figure 5.2, the ED$_4$ will not transmit as soon as it enters the satellite footprint, but after the ED$_3$ finishes its transmission. By doing so, SALSA with its Time Division Multiple Access (TDMA) approach and First Come First Served (FCFS) policy results

to be a collision-free scheduling algorithm. It intrinsically avoids collision, not allowing conflicting parallel overlapping transmissions.

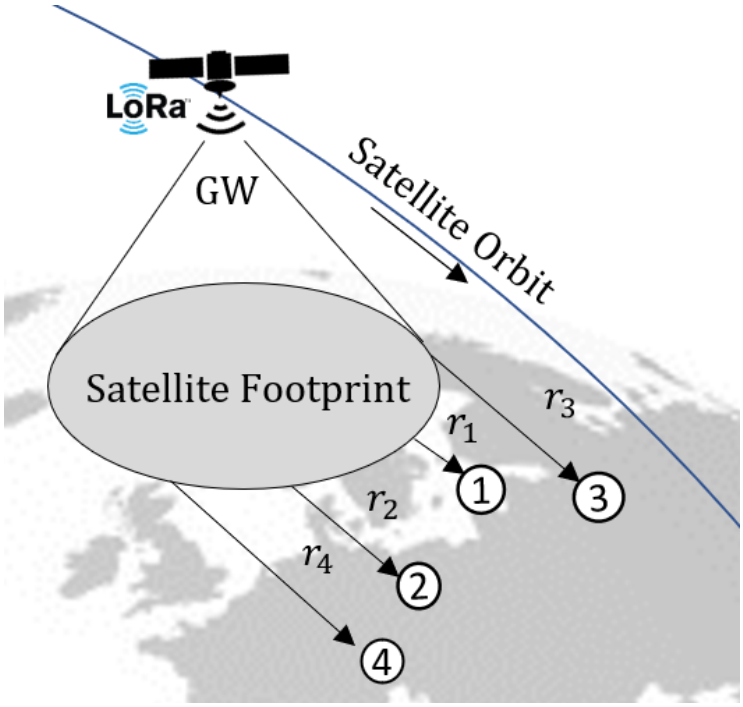


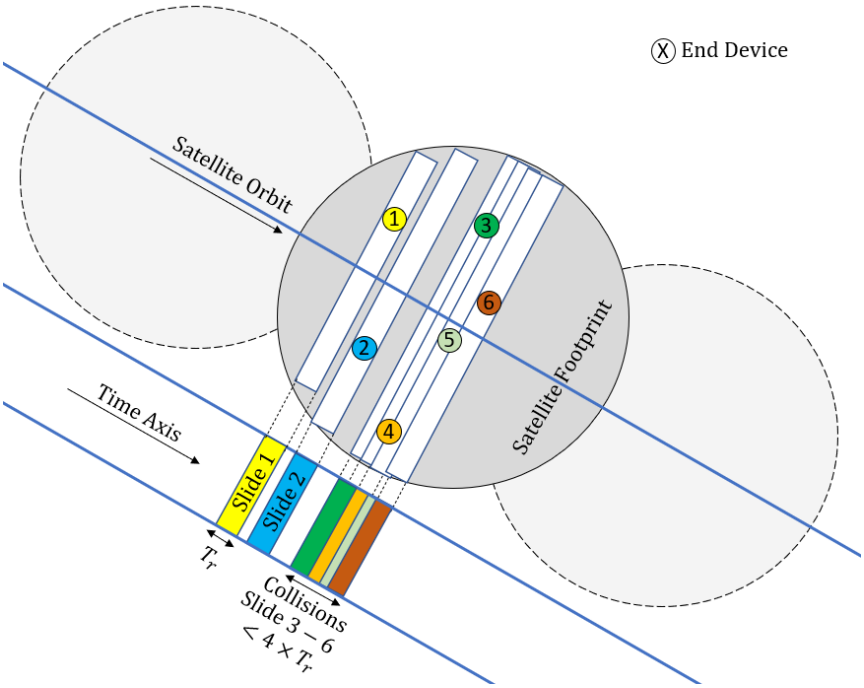
Figure 5.2: Scheduling of EDs transmissions according to the satellite movement along its orbit, and the visit time of the EDs. EDs that are visited first have first chance to transmit.

Algorithm 1 SALSA with FCFS policy

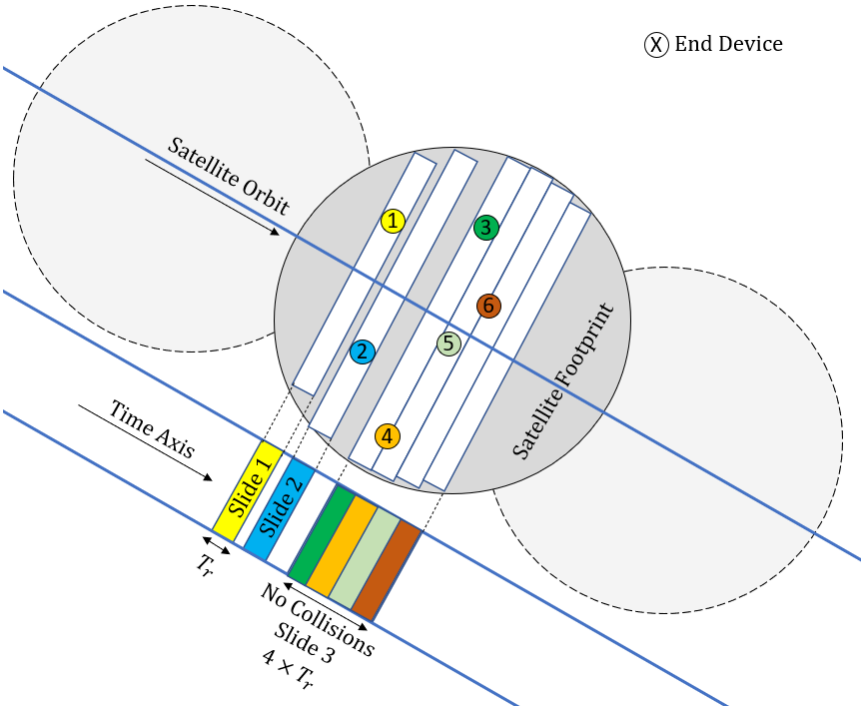
```
for slide_k do
    for ED_i do
        if T_SSV_i < T_E_j then
            T_S_i ← T_E_j                    ▷ TX after last ED
        else
            T_S_i ← T_SSV_i                  ▷ TX when satellite is available
        end if
    end for
end for
```

5.3.2. Fair Policy

The LEO satellite is available for a limited time, equal on average to $100s$. The visibility time could be shorter than the total time needed for scheduling at least one uplink transmission from all the EDs within every single slide. In a dense area with hundred of



(a) Overlapping transmissions.



(b) Collision Cancellation.

Figure 5.3: Satellite footprint and slides. Cancellation of collision with TDMA approach and FCFS policy.

EDs, several EDs that are visited last by the satellite may never get a chance to transmit. They will miss the satellite each time it is their turn to transmit, according to the FCFS policy. The FAIR policy is proposed to overcome such shortcomings and ensure that all the EDs have an equal chance to transmit. The logic behind it is described by Algorithm 2.

Algorithm 2 SALSA with Fair Policy

```
for slide_k do
    for ED_i do
        if n_{tx_i} ≤ n_{tx_j} then                    ▷ Fairness check
            if T_{SSV_i} < T_{E_j} then
                T_{S_i} ← T_{E_j}                      ▷ TX after last ED
            else
                T_{S_i} ← T_{SSV_i}                    ▷ TX when satellite is available
            end if
        else
            Do not TX                                  ▷ Give the chance to the next ED
        end if
    end for
end for
```

Let consider N EDs within a given slide of the satellite footprint. The LNS first checks if the ED$_i$ had already several transmission opportunities, n_{tx_i}, higher than the other end devices ED$_j$, within the same slide. In this case, with $n_{tx_i} > n_{tx_j}$, and $j \neq i$, the ED$_i$ will not be granted an uplink slot. On the contrary, if it can transmit because it had transmission opportunities fewer times than its neighbors, the LNS has to define the starting time T_{S_i} of its transmission, according to the FCFS policy (see Algorithm 1).

5.3.3. Optimal Schedule

After scheduling the EDs' transmissions according to the *FAIR policy*, there might be still several slots available, depending on the satellite visibility and the density of the EDs in the coverage area. Due to (i) the short availability time of the satellite and (ii) the LoRa duty cycle limitations imposed by the LoRaWAN regional parameters, it is very unlucky that a given ED have a second chance of transmitting within a single satellite visit. For instance, the maximum ToA for {51 Bytes, EU868, SF12, 125KHz}[58] is equal to 2793.5ms while the duty cycle is 1% in the EU region. So the ED must wait for 279.35s

before having a new uplink opportunity, while the availability time for a LEO satellite is shorter than 120s at 500Km height above 30° elevation.

While running the SALSA algorithm the LNS also implements a check on the duty cycle limitations and further optimizes the use of the limited network resources. And whenever it identifies a *gap* (empty slot) in the schedule, it allocates it to another ED_i, at the conditions that: (1) The new expected reserved time, T_{r_i}, does not create any conflict in the scheduling table with the already reserved slots for other EDs. (2) The ED_i has visibility with the satellite in that expected time.

5.3.4. Multi Channel SALSA Approach

The SALSA algorithm is designed for worst-case scenarios, utilizing a single SF and one channel. To extend its application, we analyzed its performance in the context of multiple channels and various permutation techniques [59]. In [59], we introduced two novel scheduling strategies, L2L-A and L2L-AP, optimized for LoRa-to-Satellite networks, with the goal of enhancing the uplink efficiency of the SALSA algorithm.

The LoRa to LEO Alternating Channel (L2L-A) algorithm emphasizes the effective use of multiple frequency channels, whereas the LoRa To LEO Alternating and Permutation (L2L-AP) algorithm incorporates time slot swapping for allocated devices, creating new transmission opportunities. These approaches significantly improve the uplink efficiency of the SALSA algorithm, particularly in dense network scenarios, resulting in notable performance gains.

The research further explores the synergy of these strategies, demonstrating the ability to boost system performance while maintaining uplink efficiency above 50% across different multi-channel implementations. Additionally, this study refines the FCFS policy of SALSA to provide more opportunities for devices at the end of the satellite's visibility window.

As illustrated in Figure 5.4, the L2L-AP Multi-Channel scheduling technique leverages additional PHY resources, surpassing the SALSA FCFS and Fair policies in scheduling uplinks. Depending on hardware capabilities, the Multi-Channel approach supports 3 to 8 channels, reducing the number of missed uplink transmissions to the satellite.

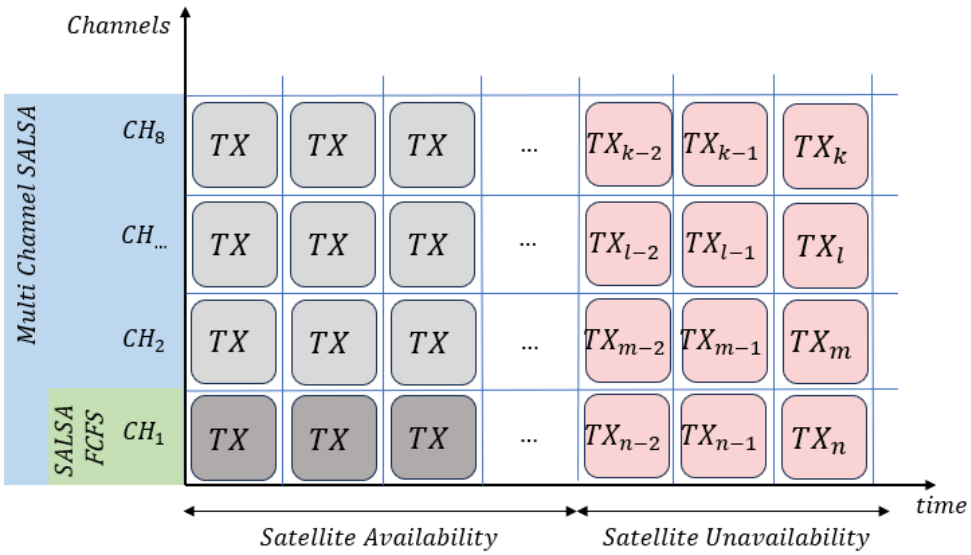


Figure 5.4: Multi Channel SALSA provides more PHY resources for the scheduling comparing to the Classical SALSA policies by taking benefit of several PHY Channels.

In the SALSA FCFS policy, several devices are unable to transmit uplinks when it is their turn in the schedule. Conversely, the Multi-Channel approach enables time slot swapping, ensuring transmission opportunities for all devices. This methodology shares similarities with the SALSA Fair policy but with the added advantage of using multiple channels to increase available time slots and uplink transmissions.

6 | Testbed Design

In order to evaluate the effectiveness of the proposed SALSA packet scheduling and REACT time synchronization methods, it is essential to observe their performance under realistic operational conditions. After detailing the theoretical design and mechanisms of these methods in previous sections, this chapter focuses on their application within both simulated and real-world environments. First, the chapter presents the simulation environment used to assess the SALSA packet scheduling technique, providing a controlled framework to analyze its performance. Then, the chapter introduces the LORSAT testbed, a comprehensive system designed to evaluate the methods in a real-world context. The LORSAT testbed encompasses various components and subsystems, each playing a critical role in testing and validating the methods under realistic constraints. These elements will be described in detail, offering insights into how they replicate real-world conditions.

6.1 Simulation Environment

Before progressing to the implementation and testing within the LORSAT testbed, it was essential to initially validate the SALSA method through a simulation environment. This preliminary simulation stage served as a critical step in refining the algorithm and ensuring its functionality before its deployment under real-world conditions. The simulation environment utilized both Python and MATLAB software to achieve comprehensive validation. In particular, the SALSA algorithm, responsible for efficient packet scheduling, was primarily implemented in MATLAB, where its performance could be meticulously analyzed and fine-tuned. Additionally, satellite visibility module was also

integrated into Python, ensuring that the dynamic nature of satellite communication networks was adequately modeled.

This simulation not only provided insights into the algorithm's performance but also set the foundation for its integration into the testbed. The concept initially tested in the simulation environment was later adopted within the LORSAT testbed, transitioning from a controlled virtual setup to a real-world context. As the project evolved, the simulation approach was further refined to align with the open-source principles, moving towards a complete Python-based framework. This shift ensured greater compatibility with the testbed and offered the flexibility needed for extended functionality in real-time conditions. The open-source Python implementation also promoted wider accessibility and collaboration within the research community, aligning with contemporary trends in open science. The source code for both the SALSA algorithm and the satellite visibility checks is available on the LIST GitHub repository [60], providing a public resource for further development and validation.

6.1.1. Module *SatelliteFly.Py*

The `SatelliteFly.py` module is structured to perform two primary functions: generating geographic positions within a selected region and determining satellite visibility from those positions during a specific time period. The code leverages external libraries for handling geospatial data.

The script imports several libraries: `skyfield.api` for astronomical calculations and working with satellite data. `geopy.geocoders.Nominatim` for reverse geocoding, converting latitude and longitude coordinates into readable location information. `random.uniform` for generating random geographic coordinates. `pandas` and `time` for data handling and timing operations, respectively.

Generating Geographic Positions: This section aims to generate specified number of random geographic points within a specified bounding box. The code randomly selects latitude and longitude values between the ranges of selected region. These coordinates are then reverse geocoded using `Nominatim` to determine the country for each generated point. For each valid coordinate, the latitude and longitude are written to a file (`positions.txt`).

Satellite Visibility Calculation: Once geographic positions are generated, the script proceeds to calculate satellite visibility for each point. This has to be handled via Satellite TLE data. Satellite TLE data is loaded from an online source (For example. Celestrak), which contains the orbital elements of many active satellites. The satellites of interest are selected from this TLE data.

For each point (latitude and longitude) saved in `positions.txt`, the `skyfield.wgs84.latlon()` method converts the coordinates into a geographic location (`bluffton`). Using the satellite object (`satellite.find_events()`), the code checks for events (such as satellite rise, culmination, and set) at this location during the given time period (t0 to t1). The function filters events based on an elevation threshold of 30 degrees, ensuring only significant visibility are considered.

The script generates three lines of output per visibility event (rise, culmination, set), each with the timestamp. The results are saved to text files for each position, prepared to be used by the SALSA algorithm.

Key Features and Considerations: This script ensures the generation of diverse points within a narrow geographic region using the random.uniform() function. Also by reverse-geocoding the coordinates and filtering based on country, the script is specifically tailored to study satellite visibility over interest region of the user. The TLE data allows the script to accurately track the satellite and predict visibility at various locations on the Earth's surface using precise orbital mechanics. This module effectively combines geographic data generation with satellite tracking, providing a comprehensive workflow to determine satellite visibility across a specified region and time period.

6.1.2. Matlab Modules

Following the initial evaluation of geographic locations and satellite visibility with the Python-based module, the next step involved incorporating packet scheduling methodologies using MATLAB-based modules from the SALSA framework. After generating and assessing random satellite positions through the Python script, these positions were employed in the packet scheduling process to simulate and analyze data transmission across different scheduling strategies.

Initially, a periodic transmission model (`Scheduled_Traffic.m`) has been implemented. This approach aimed to observe the behavior of the direct satellite IoT communication system in the absence of any packet scheduling mechanism. By applying periodic transmissions, the fundamental dynamics of data flow and latency could be assessed without the influence of advanced scheduling, providing a baseline for later comparisons.

Building upon this, the SALSA scheduling modules (`SALSA_FAIR.m` and `SALSA_FCFS.m`) were utilized to verify the impact of SALSA packet scheduling. Specifically, two FCFS and Fair scheduling policies were tested. These modules allow the user to compare the performance of SALSA scheduling techniques with no-scheduling conditions.

These modules enable the verification of key transmission metrics, including the number of packets successfully transmitted, dropped, or collided during the scheduling process. By tracking these parameters, the system is capable of calculating the Packet Delivery Ratio (PDR), which is essential for assessing the efficiency and reliability of the communication system.

6.2 LORSAT Testbed

The LORSAT testbed is developed to extend the simulations of the SALSA scheduling algorithm into a real-world context by implementing a complete end-to-end LoRaWAN network under different backhauling conditions. The objective of this setup is to replicate the operational environment in which the SALSA algorithm would function, thereby enabling a comprehensive evaluation of its performance in real network scenarios.

In designing this testbed, key elements of a traditional LoRaWAN network were adapted to account for satellite link characteristics, such as extended network delays, variable link quality, and intermittent connectivity. This approach ensured that the end-to-end network design incorporated all relevant factors, including gateway placement, device configurations, link budgets, and satellite availability that would affect data transmission between terrestrial devices and satellites.

By implementing the SALSA scheduling algorithm and the REACT time synchronization method within this testbed, the testbed aims to measure the efficacy of the proposed methods in managing resource allocation, minimizing and avoiding packet collisions, and maximizing PDR under real-world constraints. This design allows for rigorous testing of FCFS and FAIR policies, previously evaluated in simulated conditions, while also assessing their adaptability to dynamic satellite environments. The performance of REACT time synchronization method can also be evaluated, using the LORSAT testbed.

6.3 Architecture

The LORSAT testbed adheres to the e2e network architecture (Chapters 4 and 5), integrating a series of real-world devices and infrastructure to model and analyze communication flows, especially SALSA and REACT methods. In this testbed, an end device is configured to transmit data packets via LoRa modulation, which are then received by the LoRaWAN Gateway. The gateway, in turn, forwards these packets to the LoRaWAN server, which processes the data and responds through a downlink transmission back to the end devices, completing the communication loop. The backhaul link between the LoRaWAN gateway and the server can be customized based on the network configuration, employing a range of technologies such as Ethernet, cellular data, drones, and satellite communications, as illustrated in Figure 6.1.

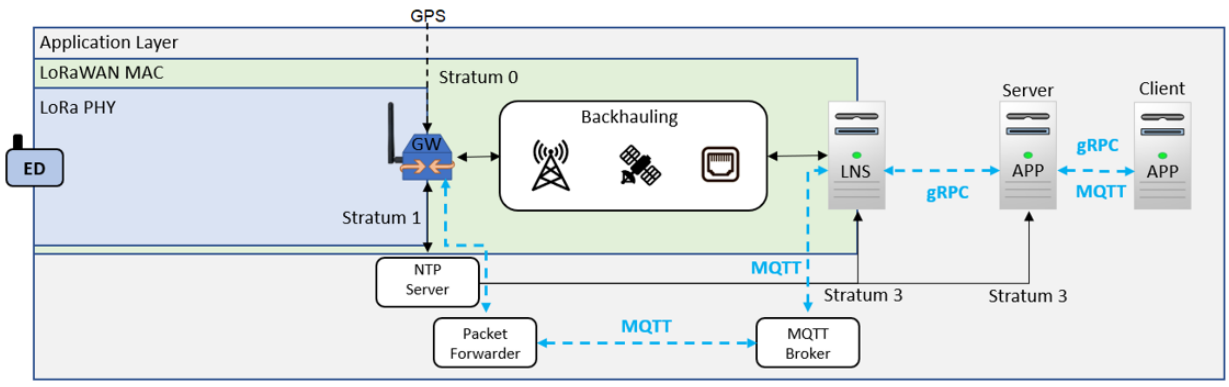


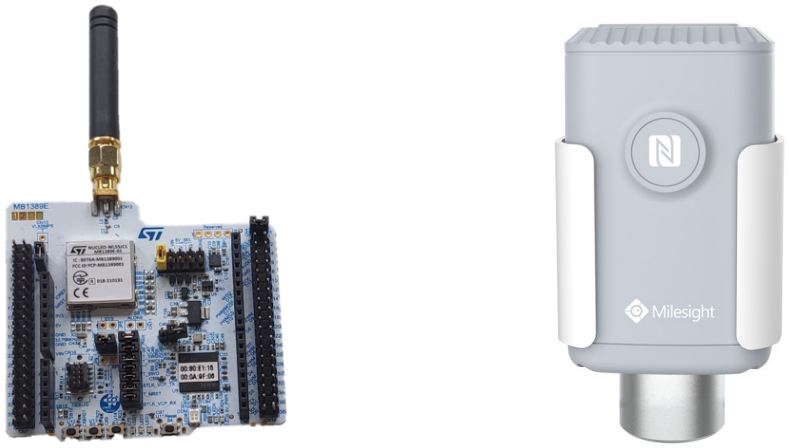
Figure 6.1: LORSAT Testbed Architecture

Each component of the LORSAT testbed will be discussed in detail, covering both hardware and software elements, including the physical devices, transmission protocols, and network management strategies. The adaptations and optimizations necessary to

meet the specific requirements of this study will be thoroughly elaborated, particularly in terms of network management. This will include the configuration of the backhaul link, which is critical for optimizing performance under different environmental and operational conditions, as well as ensuring seamless communication between the LoRaWAN Gateway and the server under varying network loads. The optimizations applied to the testbed were driven by the need to simulate real-world conditions accurately, while addressing the particular constraints and goals of the current research work, presented in this thesis.

6.3.1. Devices

End devices (**EDs**) collect data and transmit in uplink via the LoRa PHY layer. The LoRa PHY can be CSS and LR-FHSS. In this thesis, the CSS have been used to transmit the data, because there was no commercial LoRaWAN gateway available on the market to decode the LR-FHSS signal. These EDs can be prototyping devices, whose firmware can be modified to follow the optimizations and methods, as depicted in the Figure. 6.2. Other commercial EDs are only useful to generate LoRaWAN traffic, following the configuration imposed by their manufacturer. In this testbed, both ED types are used and evaluated.



(a) STMicroelectronics WL55JC1

(b) MileSight EM500-CO2

Figure 6.2: Prototyping and Commercial LoRaWAN EDs

The `STM32CUBEIDE` application compiles the codes in C++ programming language to program the STMicroelectronics `WL55JC1` ED. In the context of this thesis, this ED is

programmed to apply SALSA and REACT methods. When the ED sends an UL to the LoRaWAN gateway, the LNS performs computations and responds via a DL packet to the ED. This ED triggers internal alarms to wake up the radio part of itself to transmit an UL. These options are not fully available to modify in the commercial LoRaWAN EDs.

6.3.2. Gateway

In the LORSAT testbed, various types of LoRaWAN gateways have been employed to explore different connectivity solutions with different scenarios. LoRaWAN gateways are generally categorized based on the number of channels they support and their deployment environment. The two common categories include 8-channel and 16-channel gateways, and they can be further classified as either indoor or outdoor gateways. Additionally, these gateways differ in their modulation support, with some supporting only CSS modulation, while others support both CSS and LR-FHSS.

Within this testbed, gateways that support CSS modulation have been used, because of their availability in the market. These include both 8-channel indoor and outdoor models. The indoor gateway utilized in the testbed is a RAKWIRELESS model based on Raspberry Pi architecture, providing flexibility and cost-efficiency for indoor deployment. On the other hand, the outdoor gateway is a Multitech MTCDIP-L4E1, which offers robust performance in harsh outdoor environments. However, the setup is not limited to these specific models. The infrastructure can accommodate a variety of brands and models, allowing for flexibility in future modifications and deployments

These LoRaWAN gateways forward the data to both entities in the network, EDs and LNS. A forwarder application has been configured on top of the LoRa Radio transceiver.

6.3.3. Network Server

Two distinct methodologies have been employed in the implementation of the LoRaWAN network server via the LORSAT testbed. The initial approach utilized a tower computer to process data packets and manage network operations within a contained, isolated environment. In contrast, the current method leverages a cloud-based server with a static IP address, enabling the management of the network remotely from any

(a) RAK-Wireless 7248C **(b)** MultiTech MTCDIP-L4E1

Figure 6.3: Indoor and Outdoor LoRaWAN GWs

location. To facilitate network event handling, the Chirpstack network server has been configured to oversee the creation and management of End Devices (EDs), LoRaWAN gateways, as well as the administration of multiple users and applications

The decision to employ the Chirpstack Network Server stems from its ability to facilitate the creation of a private network, offering full control and flexibility for modifications at all levels. This adaptability allows for the network to be optimized specifically for the unique requirements and scenarios of the testbed, ensuring enhanced customization and performance tuning across different operational parameters. The Chirpstack LoRaWAN network server interfaces with the Chirpstack packet forwarder, which is installed on the LoRaWAN gateway, through the use of the MQTT protocol and gRPC services. This communication framework enables efficient data exchange and command execution, facilitating seamless integration between the network server and gateway for the management of packet forwarding and other network operations. The utilization of both MQTT and gRPC ensures reliable, scalable, and low-latency communication, essential for maintaining robust network performance.

6.3.4. Smart Application Server

While the sole use of the Chirpstack Network Server does not provide the flexibility required to implement fully customized scenarios for handling each packet and generating appropriate downlink responses, this thesis addresses these limitations by developing a smart application. The application is designed to overcome these challenges,

enabling tailored packet management and precise downlink responses that align with the specific needs of the network. This solution ensures more dynamic and context-aware handling of network events, addressing the gaps in the standard Chirpstack implementation.

This application has been developed using the Python programming language and integrates the SALSA and REACT modules. These modules enable the evaluation of real packet flows, allowing for comprehensive testing and performance analysis within the network environment. By incorporating SALSA and REACT, the application is equipped to assess the network's behavior under real-world conditions, ensuring that the system can handle and respond to packets in a way that meets the specific requirements of the customized scenarios.

Although Chirpstack provides access to packets and network events through the MQTT protocol, the LORSAT smart application server leverages these MQTT events to address specific operational needs. By utilizing the real-time data and event notifications transmitted via MQTT, the smart application processes and responds to network demands with greater precision. This integration allows for a more flexible and tailored approach to managing network behaviors, ensuring that the system can dynamically adapt to various scenarios based on the incoming packets and events.

The smart application server also capitalizes on the Chirpstack API to manage and adjust network configurations as needed. By utilizing the Chirpstack API, the application is able to exert control over various network parameters, enabling real-time modifications to optimize network performance and accommodate changing operational requirements. This API-driven approach provides the flexibility necessary for dynamic reconfiguration, ensuring that the network remains adaptable and responsive to evolving demands.

The smart application server additionally offers Satellite visibility time windows for the LoRaWAN gateway, a feature critical for modeling the availability of LEO satellites in relation to the gateway. By providing these visibility windows, the application enables the precise scheduling and prediction of communication opportunities between the LEO satellites and the LoRaWAN network. This functionality is essential for optimizing satellite connectivity and ensuring efficient data transmission during the periods when the satellites are within the gateway's communication range.

6.3.5. Backhauling Links

The LORSAT testbed enables the integration of diverse backhauling options to connect the LoRaWAN gateway with the LoRaWAN network server. Among these, Ethernet is the most straightforward choice, offering reliable communication via standard cable-based connections. This method ensures a stable and consistently available link, with an average delay of approximately 2 milliseconds, excluding additional delays from internet backhauling. However, in rural and remote areas, Ethernet connectivity is often unavailable, necessitating alternative solutions such as cellular and satellite links. Cellular connectivity, while sometimes accessible, is entirely absent in many rural regions, making it an unreliable option for widespread deployment. Even when available, cellular networks typically introduce delays of tens of milliseconds due to inherent network latency. In contrast, satellite communication—though characterized by even higher latency—provides a more universally accessible backhauling option in remote areas where terrestrial networks are absent. By incorporating Ethernet, cellular, and satellite links, the LORSAT testbed ensures robust and adaptable network connectivity, addressing the diverse infrastructure challenges encountered in varied geographic settings.

The LORSAT testbed leverages three distinct satellite-based architectures, as illustrated in Figure 6.4. The first architecture connects the LoRaWAN gateway to LoRaWAN servers through a GEO satellite terminal. This terminal utilizes the SES SATCUBE antenna, which is equipped with a modem, that connects directly to the ASTRA 4 satellite. The configuration depicted in Figure 6.5 shows the installation of the SES SATCUBE antenna on the rooftop of the LIST Belvaux site during one of the experimental tests. This satellite terminal facilitates the forwarding of received LoRaWAN packets to the real GEO satellite, introducing a significant RTT delay of approximately 630 ms into the network.

To enhance control over satellite conditions and configurations in the research environment, the testbed also integrates the OpenSAND open-source GEO satellite emulator [61]. OpenSAND emulates the satellite terminal, gateway, and GEO satellite components, offering a versatile platform for studying GEO satellite network design. Furthermore, OpenSAND provides flexibility for expanding the system into real-world networks with capabilities for both IP-based and MAC-based LAN extensions. Through

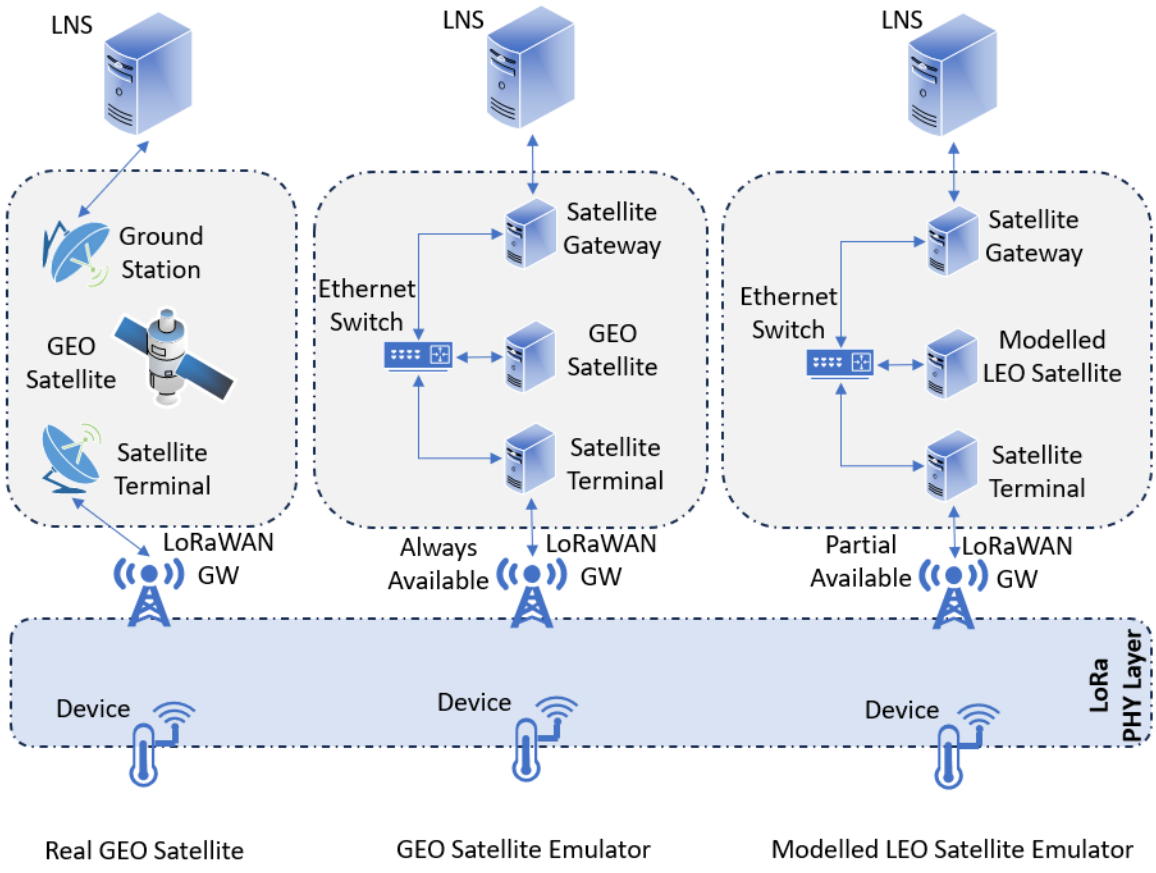


Figure 6.4: LORSAT Testbed configurations

this setup, the emulator allows for adjustments in delay and link quality, enabling the execution of various test scenarios within the LORSAT testbed to simulate different satellite communication conditions.

6.3.5.1. LEO Satellite Modelling

The methods proposed in this thesis necessitate the deployment of a LoRaWAN gateway installed on a LEO satellite. Currently, there is no LEO satellite emulator available. Communication with the satellite terminal is therefore essential as an intermediary step. Despite the fact that several companies are developing LEO satellites for LoRaWAN services, significant limitations remain, such as the absence of downlink capabilities and the lack of control over the network infrastructure. Consequently, in this research, the

Figure 6.5: SES SatCube Terminal

presence of a LEO satellite has been modeled using the OpenSAND GEO satellite emulator.

Implementing this model requires careful consideration of several factors. Since the LoRaWAN gateway used in the testbed is a fixed device, the model assumes that all devices are located in a single geographic location. Furthermore, it is assumed that the LoRaWAN gateway is available only when real satellites are visible overhead at this location. This approach ensures that if the proposed methods function effectively under these conditions, they are likely to operate successfully with actual LEO satellite systems as well. This modeling strategy facilitates the study of satellite communication within constrained environments, thereby providing valuable insights into the viability of integrating LoRaWAN gateways into LEO satellite networks.

To ensure the effective functioning of this model, a LEO modeler application was developed as part of this thesis. This application utilizes visibility windows generated by the Smart Application Server to simulate real-world satellite interactions. When the LEO satellite is within the visibility range of the devices, the application activates the Chirpstack gateway forwarder service, enabling communication. Conversely, when the satellite moves out of range, the application disables the Chirpstack gateway service, causing any subsequent packets to be dropped.

This approach ensures that packet transmission and reception align with the satellite's actual visibility, thereby emulating realistic conditions for communication. By controlling gateway availability in this manner, the model assesses the accuracy of the REACT time synchronization and the SALSA packet scheduling methods. The integration of these mechanisms into the LEO modeler ensures that the testbed accurately replicates the dynamic nature of LEO satellite communication, thus providing a robust framework for testing and validating the proposed methods under realistic operational constraints.

7 | Performance Evaluation

In this chapter, the performance of the REACT and SALSA methods has been thoroughly evaluated. To validate the effectiveness of these proposed methods, a two-step approach has been adopted: Simulation and Testbed experimentation. Initially, the methods were assessed through simulations to verify their overall concepts. Subsequently, their proof of concept was confirmed using LORSAT testbed.

7.1 Simulating SALSA

In this section, first the test setup used for reproducing the system behavior is described for implementing the SALSA algorithm. Then, the performance achievable with SALSA is evaluated, and compare them with the benchmark solution.

A network size variable in the range $[50, 500]$ has been considered. The EDs are located according to a random and uniform distribution in a target area. In this evaluation, the target area is assumed to coincide with the Luxembourg country (Fig.7.1). the exact location of the EDs has been derived using random uniform distribution in the GeoPy [63] library of Python programming language. Besides the considered scenario, the algorithm is not limited to this selected geographical area and applies to any other one.

The Lacuna satellites have been considered, in particular *Lacuna-Sat 3* and *Lacuna-Sat 2B* which have the approximate height of 500Km to 600Km from the earth. In the performance analysis, their real visibility has been adopted over Luxembourg during October 2021. Moreover, an elevation angle $\Theta_e = 30^\circ$ has been assumed for both satellites.

The SKYFIELD library [64] has been used with the CelesTrak [65] online TLE sources for generating the real visibility time tables of the satellites per each ED. A TLE provides the

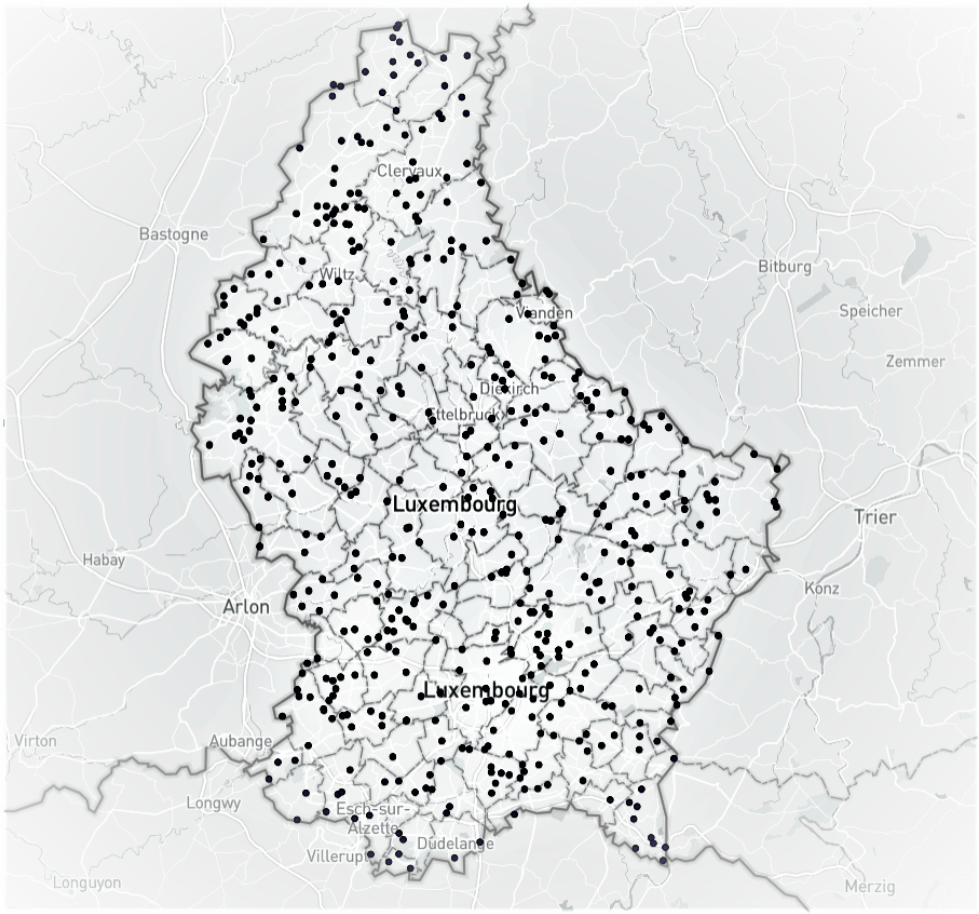


Figure 7.1: Location of EDs over the country of Luxembourg, following a random uniform distribution. Visualization done with [62].

satellite location along its orbit for a given pointing time (the time epoch), by encoding the orbital elements. The specific location of the EDs (uniformly distributed over the Country of Luxembourg), the satellites TLE, and the fixed elevation angle ($\Theta_e = 30°$) were given in input to the SKYFIELD. Finally, the obtained human-readable time outputs were translated into epoch timestamps and used in the Matlab algorithm, implementing SALSA. The code is public available at [66].

The Guard Time T_g has been assumed to be equal to $10ms$; and the ToA is the maximum allowed ToA in the EU868 region, with $SF = 12$ and the maximum payload of 51 bytes. It follows, the time reserved for each ED, for a single transmission is $T_r = 2 \times T_g + ToA = 2813.5ms$.

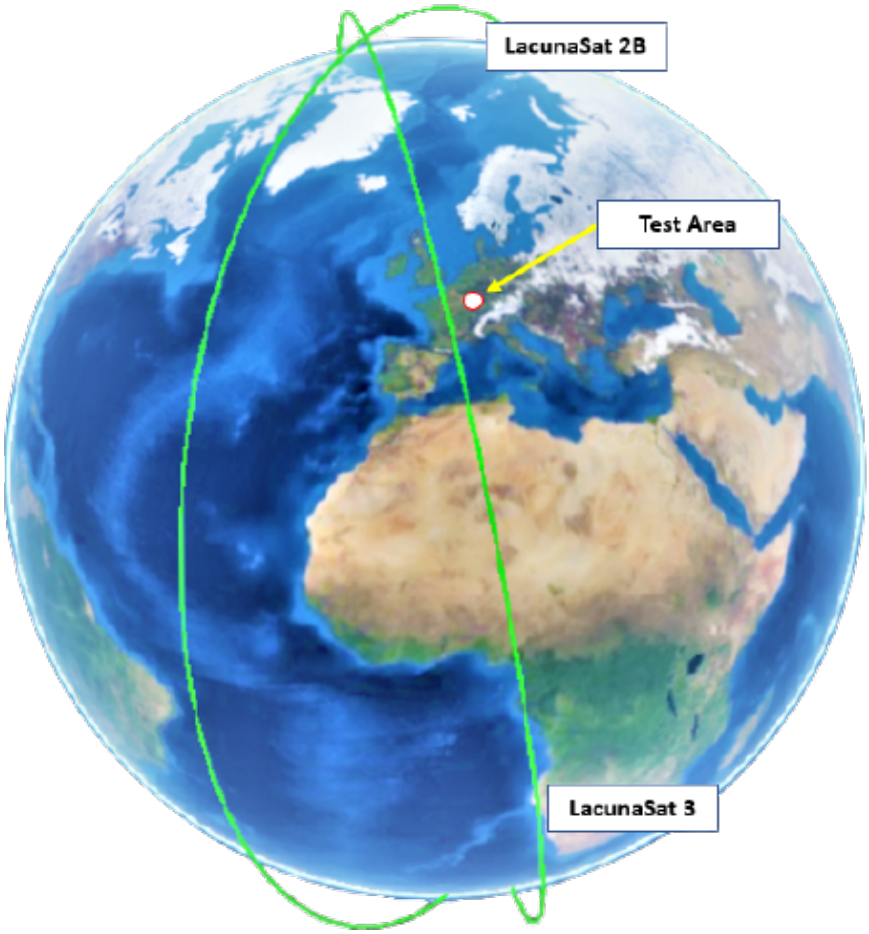


Figure 7.2: Orbits of LacunaSat 3 and LacunaSat 2B satellites compared to the target area on the Earth, Luxembourg.

7.1.1. Achievable Performance

The behaviour of the satellite LoRaWAN network has been compared in three different scenarios:

- **ALOHA**: the EDs transmit following the LoRa standard, with random ALOHA access. They transmit periodically, every 30 minutes, without having any knowledge about the satellite visibility. Note that a random uniformly distributed offset has been introduced in the start time of the first uplink per each ED.
- **SALSA w/ FCFS**: the EDs that are visited first by the satellites transmit first, but only when the channel is not busy (no collision due to overlapping transmissions).

- **SALSA w/ FAIR**: all the EDs get an equal chance to transmit thanks to the FAIR policy, in addition to FCFS.

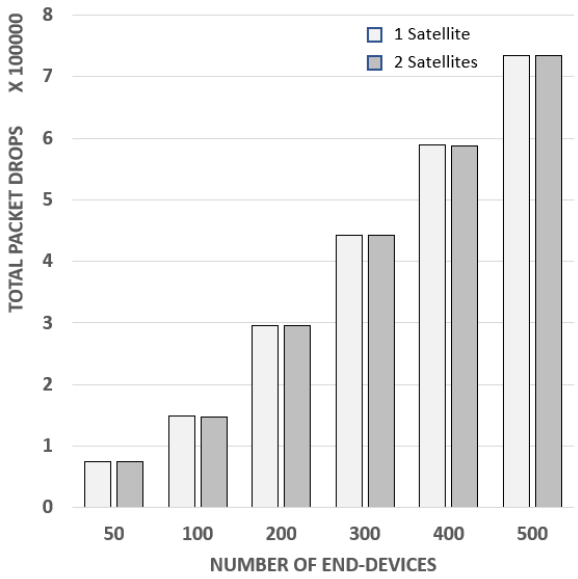


Figure 7.3: Packet drops during one month when the EDs transmit with ALOHA-based LoRa, generating periodic traffic, every 30 minutes.

When (i) the EDs are not aware of the visibility time of the satellite(s), and (ii) no scheduling algorithm is implemented, the network faces a high number of *packet drops* and *packet collisions*. Packet drops are due to the lack of availability of the satellite gateway, while packet collisions are due to concurrent random transmissions. As shown in Fig. 7.3, the adoption of a constellation of two satellites does not improve the network performance in the absence of a scheduling technique. As depicted in Fig. 7.4, the performance in term of the number of collisions get even worst with two satellites: it is almost double with a large network size.

Contrary to random ALOHA-like access, when adopting SALSA the LNS schedules the transmissions while aware of the location of the EDs and the satellite availability time for each of them.

The performance achievable with FCFS and FAIR policy has been compared in the worst-case scenario, with a single LEO satellite, and with a constellation of two satellites (Fig. 7.5). FCFS policy schedules the transmissions according only to location-based priority. It follows that some devices (those visited last by the satellite) never get a chance

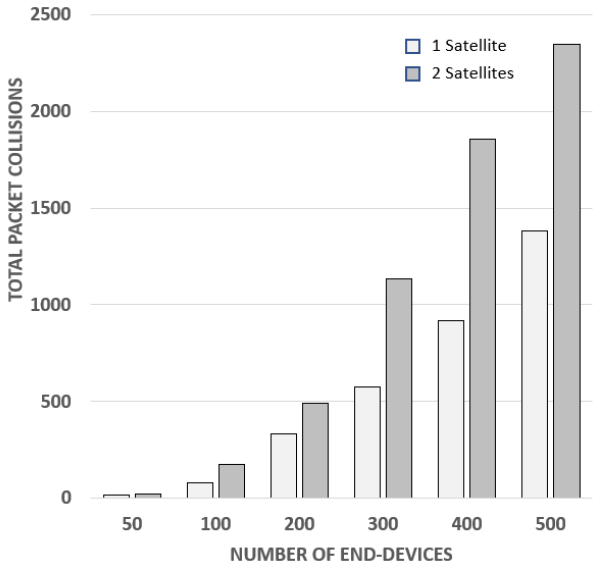


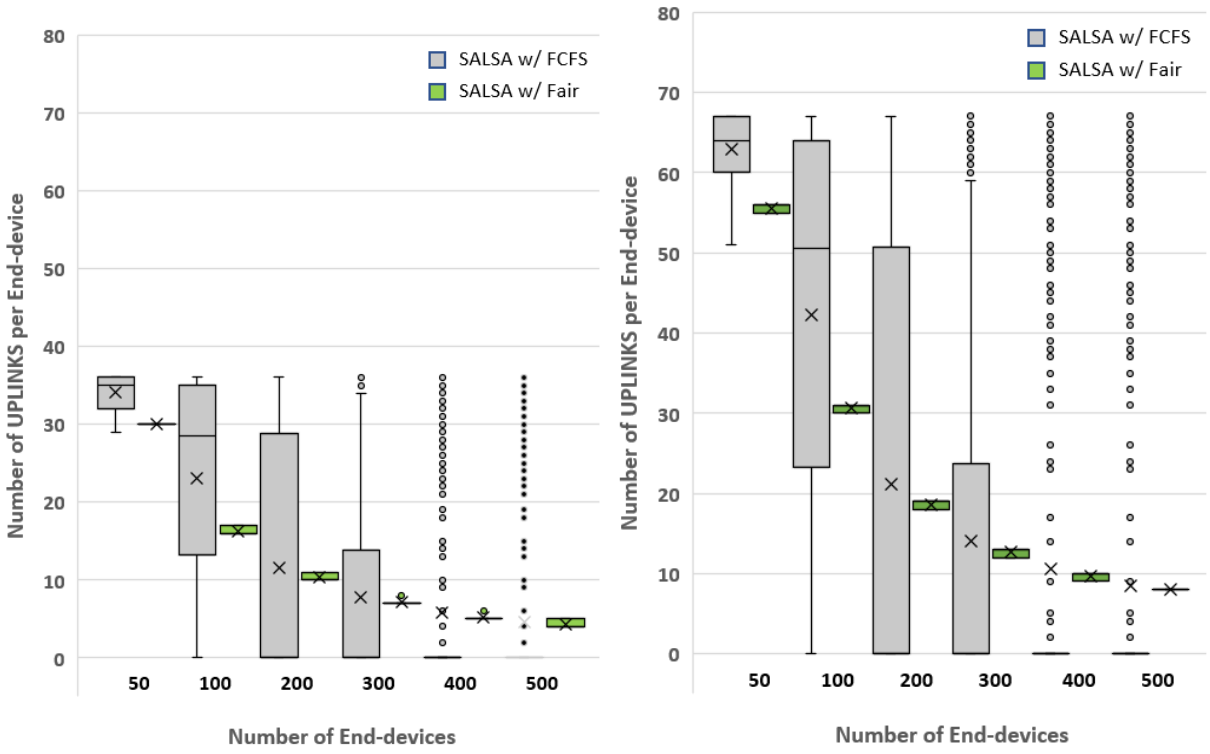
Figure 7.4: Packet collisions during the satellite visibility in one month with 1 and 2 satellites.

of an uplink opportunity. Moreover, each device may get a different number of reserved slots (as confirmed by the large whiskers and the many outliers). This behavior is more pronounced with two satellites. On the contrary, the FAIR policy guarantees all devices get the same chance, even in large networks, besides the size of the constellation. With two satellites, the network performance improves, and the average number of uplink transmissions per ED is almost duplicated compared to the scenario with a single LEO satellite. The obtained results confirm the reliability and scalability of the SALSA algorithm with the FAIR policy.

7.2 Evaluation of REACT method

To validate the proposed REACT method, and check its feasibility, LORSAT testbed by OpenSand GEO satellite emulator [46] configuration has been used, supporting two backhauling links (terrestrial, and satellite). The testbed with its hardware (HW) components, and the Software (SW) configuration is illustrated in Figure. 7.6.

The EDs acting as the sensors and actuators transmit the data to the GW using the LoRa CSS PHY, where the GW is able to forward the packets through terrestrial and satellite backhauling links. The LoRaWAN GW is synchronized with a GPS clock. This GPS receiver is assumed as the main time reference of the entire network, in our method and



(a) 1 Satellite

(b) 2 Satellites

Figure 7.5: Average Number of Uplink Transmission for each ED, with FCFS and FAIR policy (a) with one satellite, and (b) with two satellites.

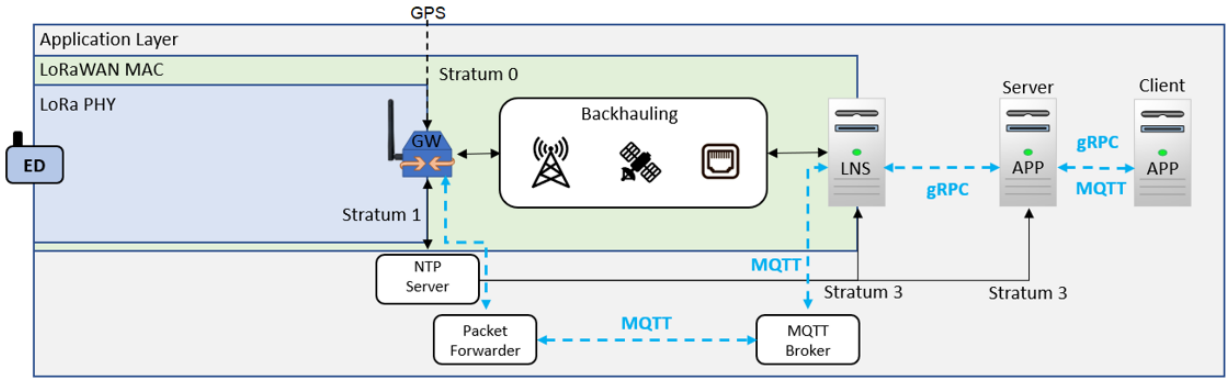


Figure 7.6: LORSAT Testbed Architecture; Configuration for performance evaluation of REACT.

formulation. A Network Time Protocol (NTP) server is configured on the GW, receiving the Stratum level 0 time accuracy from the GPS, and sharing the time with the LNS and the APP server. The LNS and the APP servers are synchronized as the NTP clients with the Stratum level 3 accuracy, which is enough accurate for the LoRaWAN network. Therefore, the entire network is time synchronized except the EDs.

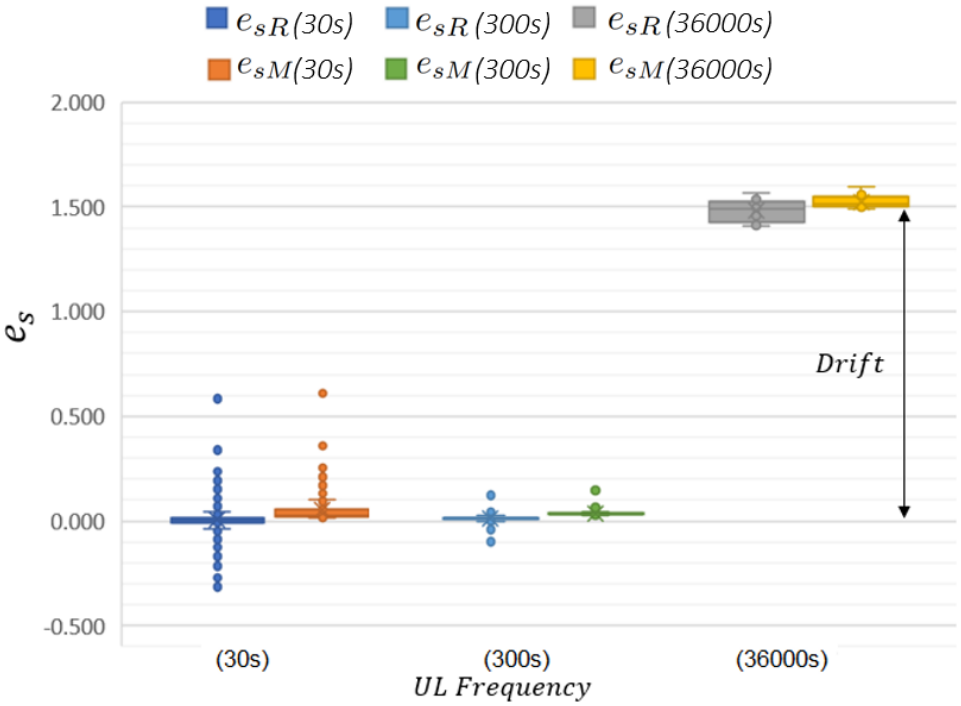


Figure 7.7: Accuracy of synchronization methods with terrestrial backhauling configuration, for different frequencies of UL updates.

The EDs will follow the REACT or MAC-based time synchronization methods to synchronize with the network. The LNS and the APP server are regular computer servers controlling the network events. The Chirpstack is installed and configured as the GW packet forwarder, the LNS and the APP layer instant. The Chirpstack benefits from the MQTT (Message Queuing Telemetry Transport) and the Remote Procedure Call (gRPC) API manager to handle the messages in the network. The gRPC is an open source Remote Procedure Call (RPC) framework that helps to control the network data flow in an efficient heterogeneous way. When there is a new event in the network, the MQTT will publish this event as a message using the MQTT broker. The subscribers, as the MQTT clients, would receive the message by establishing a connection to the MQTT server. The gRPC together with the MQTT create a full end to end message flow in the LoRaWAN network. The APP client would also benefit the MQTT using a client version and a gRPC extension. In our testbed, the APP client is the entity that answers the REACT messages. The backhauling links available for this test on our LORSAT testbed are the terrestrial (regular Ethernet), and the Satellite (OpenSAND Emulator).

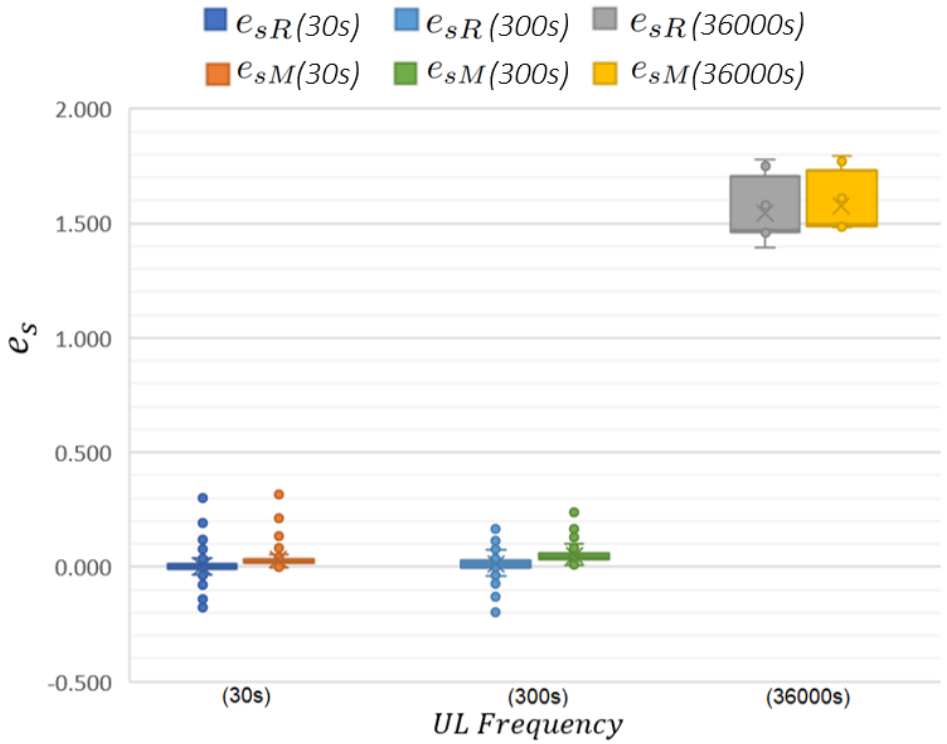


Figure 7.8: Accuracy of synchronization methods with satellite backhauling configuration, for different frequencies of UL updates.

This method is evaluated under the terrestrial and GEO satellite configuration, comparing the MAC and REACT method. Two EDs have been used to assess the proposed method, with during 1 week. The UL intervals are from $30s$ to $36000s$ to identify the method efficiency in the case of network unavailability. The $30s$ assess the method with the fast time synchronization intervals, while the $36000s$ models one LEO satellite availability with approx. $500km$ orbit height for a fixed location of one ED. Moreover, two backhauling types, satellite and terrestrial links have been considered to verify the tolerance of synchronization method to the backhauling. In the test set-up, $SF = 12$ has been selected during the experiments with the $EU868$ configuration.

To evaluate the performance of the synchronization methods, it is necessary to log the events in all the network component. The ED logs three values and place them in the UL payload to send them to the application server. Each value has $6B$ of data, presenting a large number with ms accuracy, in total $18B$. The first measurement is the value of the ED interval timer at the time of synchronization request (t_r). The others are MAC, and

$REACT$ based timestamps presenting the time measurements of each synchronization method.

Thanks to the Chirpstack forwarder application, the events of the GW can be logged using the Chirpstack $bridge$ service. The most important information are *"UL packet arrival"* and the *"DL TX"* timestamps. The difference of these values presents the round trip backhauling delay, and the process time in the LNS. The process and waiting times are around $200\ ms$ to $300\ ms$ at the Chirpstack LNS and APP servers. The LNS provides raw information of the packets in the network. An intelligent client application is developed to implement the $REACT$ method in the network, including the logging system. The client application logs the three values from the received UL, time of arrival - T_L and the ED Unique Identifier (EUI). By analyzing the logs, it is possible to compute the accuracy of the time synchronization methods, and the measurement of the drift effect.

To evaluate the e_{sR}, the REACT time synchronization accuracy from the test, calculating the difference of the ED timer is required in the time of request (t_r) and the $T_{A_{new}}$. From the Eq. 4.3, $T_{A_{new}}$ is the time when the ED is synchronized with the network and delivered a new synchronization request to the LNS. The $T_{A_{new}}$ would be derived following the Eq. 7.1. The T_R is also the time of the LNS when the ED requests the time, t_r.

$$T_{A_{new}} = T_R + ToA_{UL} + e_{REACT} \tag{7.1}$$

$$e_{sR} = T_{A_{new}} - t_{REACT} = T_{A_{new}} - (T_R + e_{REACT}) \tag{7.2}$$

By substituting the $T_{A_{new}}$ from the Eq. 7.1 in the Eq. 7.2, the e_{sR} would be as the Eq. 7.3.

$$e_{sR} = ToA_{UL} \tag{7.3}$$

When the LNS receives the packet, the ToA_{UL} would be a known value. So, the ToA_{UL} can be removed from the e_{sR} to achieve exact accuracy. In our experiment, the ToA_{UL} is $1810\ ms$ {`SF12, EU868, 18B`}. The ideal target is to have compensated e_{sR} as 0. The other calculated values rather than 0 would be considered as inaccuracy. With the same approach, e_{sM} would be the compensated accuracy showing the difference between the MAC-Based ED time and the LNS time, $T_{A_{new}}$.

The Figure. 7.7 and 7.8 show the Box and Whisker plots of the compensated e_{sR} and e_{sM} in different test scenarios. The tests are classified in three different transmission intervals {$30s$, $300s$, and $36000s$} for the MAC and $REACT$ methods. The performance of the time synchronization methods were tested in different backhauling scenarios. Therefore the Fig. 7.7 and the Fig. 7.8 provide the results with the terrestrial and the GEO satellite backhauling, respectively.

The figures clearly show that there is a tradeoff between the synchronization update interval and the accuracy performance. Frequent updates bring better accuracy but several variations exceed the normal accuracy range. On the other side, low frequent updates stabilize the range of the values, but they receive a high drift effect. In the 30 s test, the drift effect is around less than 1 ms, while it would be around 1.6 s in the 36000 s interval. In the *"partial available networks"*, this drift can have a huge impact on the scheduling techniques provided to reach the availability time. Due to these drifts, the EDs may assume that the gateway and the network is available, while it is not the case (and it will be only available at a later stage). The drift effect can be calculated for each ED and programmed in the ED application to compensate it. Using satellite backhauling with the large delays does not have a specific effect on the performance of the time synchronization, for both REACT and MAC methods. In all the tests, the $REACT$ method provides almost the same accuracy compared to the MAC method, with the ms accuracy. This option makes our REACT method operates better than the standard application method (offering s accuracy).

7.3 Validation of e2e system

In this section, an evaluation of the SALSA packet scheduling and REACT time synchronization algorithms will be conducted using the LORSAT testbed. The LORSAT testbed has been meticulously configured in the "modeled LEO satellite" mode to establish a realistic environment for the comprehensive evaluation of these algorithms. The LoRaWAN devices are located in the same place, close to the LoRaWAN gateway. The LACUNA Space satellites are selected as base for using in the SALSA scheduling algorithm. When one of the LACUNA Space satellites, for example `LACUNASAT - 2B`, passes over the LoRaWAN devices above a specific elevation angle, the LoRaWAN gate-

way starts to receive the packets from the devices. This scenarios allows to check the accuracy of the REACT time synchronization method, fitting into the satellite visibility time window. Also, analyzing of packet delivery ratio, PDR, of the e2e system proves the performance of the SALSA scheduling algorithm and REACT time synchronization method at the same time. If a packet is not following the correct system timing, then there would be packet drop or collision, even in the case of a perfect packet scheduling algorithm.

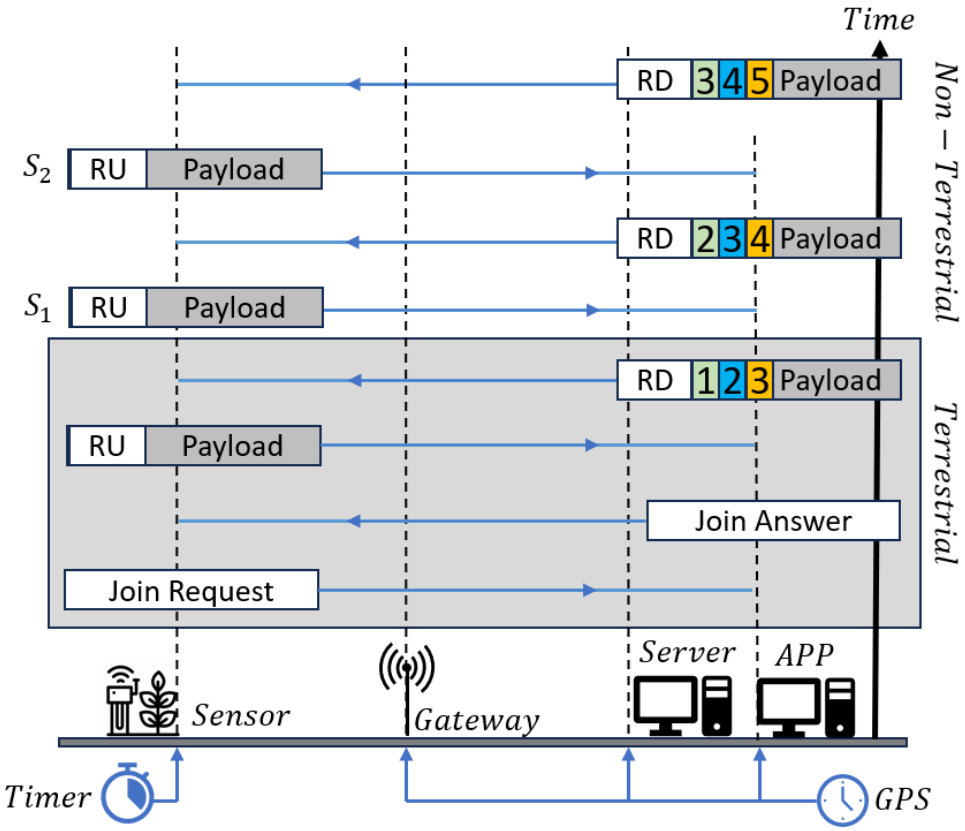


Figure 7.9: Test Setup

Based on the Figure. 7.9, the e2e system performs the evaluation of these two algorithms. The LoRaWAN device first requests to join the network, via a `Join Request` message. The LNS replies back with a `Join Answer` message to let device sets up the configuration. This section will continue for entire devices to join the network, and transmit their first packet via a terrestrial network. The first packet includes a `RU` message, in addition to the regular user payload. The `RU` message is the REACT uplink, sending the timer value of the device. When the LNS and APP servers receive the `RU`

message, they reply back via a `RD`, REACT downlink message, to the device, and the next three earliest time slots allocated for this specific device. These time slots are read from the time schedules calculated from the SALSA scheduling algorithm. The servers send three time slots to make sure that the device has enough chance to communicate with the satellite without losing time synchronization and receiving new time schedules.

After being sure that all the LoRaWAN devices joined the network and received their first time slots, `{S1,S2,S3}`, the network switches to a Non-Terrestrial setup using a modelled LEO satellite of OpenSand satellite emulator. The device will transmit at the first time slot allocated, `S1`, with the same request of `RU` in the uplink message. Then, the servers will respond via the three next possible allocated time slots, `{S2,S3,S4}`, and the `RD` to the device.

This scenario will continue till the network configuration changes or the LoRaWAN device looses all three possible downlinks, caused by any probable issue in the network or PHY layer. Therefore, transmitting pure three time slots can be optimized by a new algorithm to reduce the risk of loosing the device.

In the test setup, 10 STMicroelectronics WL55JC and a RAKWIRELESS 7248C LoRaWAN gateway have been used. These 10 devices followed SALSA packet scheduling algorithm based on the time synchronization of REACT method. The LoRa Transceiver was serving the EDs based on the passes of the LACUNA SPACE satellites over the fixed location. Therefore, the performance of the e2e system proved of having no packet drop and collision using these two methods.

8 Conclusion

8.1 Results and Discussion

This chapter encapsulates the results and implications of this thesis, which focused on packet scheduling and time synchronization techniques for satellite IoT networks. The findings demonstrate the effectiveness of the proposed methods across various conditions and configurations, highlighting reliability and scalability improvements of the end to end network performance.

In the absence of terrestrial IoT networks with reliable internet access to send data to clouds and servers for monitoring, satellite technology emerges as a critical enabler. LEO satellites, due to their proximity to Earth, offer significant advantages including reduced power requirements for data transmission and enhanced connectivity for sensors deployed in the field. While GEO satellites can provide a backhauling link via LoRaWAN gateways, LEO satellites remain preferable due to their lower latency and power needs. However, the introduction of satellite connectivity, particularly with LEO satellites, presents the necessity for devices to be aware of satellite visibility windows, and challenges such as increased packet collision probability.

To ensure accurate time synchronization across devices—a prerequisite for the allocated time slots required by event-based transmissions and scheduling methods—the REACT time synchronization method was proposed. REACT achieves time synchronization accuracy on par with standard methods and is compatible with all devices irrespective of the LoRaWAN specification version or additional hardware requirements. When satellites are in use, a larger number of devices are in the coverage of the network, bringing more packet collision and drop issues. To mitigate these challenges, the SALSA schedul-

ing algorithm was proposed, incorporating two distinct policies: FCFS and FAIR. This algorithm effectively allocates time slots to devices, addressing the visibility and reliability issues inherent in LEO satellite usage.

The validation of the SALSA scheduling algorithm commenced with the development of a simulator, which confirmed its feasibility. Subsequently, the implementation of our developed testbed allowed for real-world application and testing, alongside the REACT time synchronization method. The use of a satellite emulator provided a realistic environment to emulate satellite conditions between the LoRaWAN gateway and the LoRaWAN network server, further substantiating the robustness of the proposed solutions.

The empirical results from these tests corroborate the effectiveness of the REACT time synchronization method and, the SALSA scheduling algorithm demonstrating their potential to meet Satellite IoT's demands. These outcomes highlight the practical applicability of combining LoRaWAN with satellite connectivity. Thus, this research not only contributes to the advancement of IoT technologies in challenging environments but also provides a scalable solution framework for similar contexts globally.

The conclusions drawn from this study emphasize the significant impact that advanced IoT and satellite technologies can have on fostering greater productivity and sustainability. The methodologies developed herein lay the groundwork for future research and development, with the potential to extend applications requiring reliable, low-cost, and efficient connectivity solutions in remote areas.

8.2 Future Research Work and outlook

While the proposed methods have demonstrated significant effectiveness, they also open up promising avenues for future research. Further exploration could focus on optimizing these methods under more diverse scenarios, enhancing their scalability, or integrating them with emerging technologies to address the evolving demands of satellite IoT networks. The practical implementation of these solutions hinges significantly on the advancements and collaborations within the satellite industry. To fully realize the potential of IoT networks, satellite companies need to address several critical issues. Foremost among these is the challenges of PHY layer between devices and satellites,

which currently pose significant barriers to seamless communication. Enhanced PHY layer compatibility is crucial for ensuring reliable data transmission and reception between ground-based sensors and orbiting satellites.

Moreover, the deployment of satellite constellations with Inter-Satellite Links (ISL) is critical for advancing IoT network capabilities. ISL-enabled constellations allow satellites to communicate directly with one another, enhancing data relay efficiency and significantly reducing latency. A key direction for future research lies in the development of a 3D network architecture that incorporates layers between satellites and ground devices, including Drones and High Altitude Platforms (HAPs). These intermediary layers can facilitate more seamless communication pathways, improving connectivity and data management. Advancing both software and hardware will be essential to fully realize these improvements, enabling the effective coordination and operation of this multi-layered architecture in IoT networks.

9 References

[1] N. Victor, P. K. R. Maddikunta, D. R. K. Mary, *et al.*, "Remote Sensing for Agriculture in the Era of Industry 5.0—A Survey," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 17, pp. 5920–5945, 2024. DOI: 10.1109/JSTARS.2024.3370508.

[2] 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 and Tutorials*, vol. 23, no. 3, pp. 1693–1720, 2021. DOI: 10.1109/COMST.2021.3078433.

[3] G. M. Capez, S. Henn, J. A. Fraire, and R. Garello, "Sparse Satellite Constellation Design for Global and Regional Direct-to-Satellite IoT Services," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 5, pp. 3786–3801, 2022. DOI: 10.1109/TAES.2022.3185970.

[4] M. M. Azari, S. Solanki, S. Chatzinotas, *et al.*, "Evolution of Non-Terrestrial Networks From 5G to 6G: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 4, pp. 2633–2672, 2022. DOI: 10.1109/COMST.2022.3199901.

[5] H. E. Susilo and J. Suryana, "Research on LPWAN Direct to Satellite IoT: A Survey Technology and Performance on LEO Satellite," in *2023 29th International Conference on Telecommunications (ICT)*, 2023, pp. 1–7. DOI: 10.1109/ICT60153.2023.10374072.

[6] T. T. T. Le, N. U. Hassan, X. Chen, M.-S. Alouini, Z. Han, and C. Yuen, "A Survey on Random Access Protocols in Direct-Access LEO Satellite-Based IoT Communication," *IEEE Communications Surveys and Tutorials*, pp. 1–1, 2024. DOI: 10.1109/COMST.2024.3385347.

[7] X. Zhu and C. Jiang, "Integrated Satellite-Terrestrial Networks Toward 6G: Architectures, Applications, and Challenges," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 437–461, 2022. DOI: 10.1109/JIOT.2021.3126825.

[8] O. Kodheli, E. Lagunas, N. Maturo, *et al.*, "Satellite Communications in the New Space Era: A Survey and Future Challenges," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 1, pp. 70–109, 2021. DOI: 10.1109/COMST.2020.3028247.

[9] *LoRa Alliance, LoRaWAN Specifications, Available online: Https://lora-alliance.org/*, Accessed on 1 Jan 2022.

[10] SEMTECH, "LoRa and LoRaWAN: A technical Overview," SEMTECH Corporation, Tech. Rep., Feb. 2020. [Online]. Available: https://lora-developers.semtech.com/uploads/documents/files/LoRa_and_LoRaWAN-A_Tech_Overview-Downloadable.pdf.

[11] W. Ayoub, A. E. Samhat, F. Nouvel, M. Mroue, and J.-C. Prévotet, "Internet of Mobile Things: Overview of LoRaWAN, DASH7, and NB-IoT in LPWANs Standards and Supported Mobility," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1561–1581, 2019. DOI: 10.1109/COMST.2018.2877382.

[12] *Standards for the IoT, Available online:https://www.3gpp.org/news-events/1805-iot_r14*, Accessed on 2 dec 2016.

[13] Z. Zhou, M. Afhamisis, M. R. Palattella, N. Accettura, and P. Berthou, "Pervasive LPWAN Connectivity Through LEO Satellites: Trading Off Reliability, Throughput, Latency, and Energy Efficiency," in *Low-Power Wide-Area Networks: Opportunities, Challenges, Risks and Threats*, I. Butun and I. F. Akyildiz, Eds. Cham: Springer International Publishing, 2023, pp. 85–110, ISBN: 978-3-031-32935-7. DOI: 10.1007/978-3-031-32935-7_3. [Online]. Available: https://doi.org/10.1007/978-3-031-32935-7_3.

[14] Chirpstack, "ChirpStack, open-source LoRaWAN® Network Server," Chirpstack, Tech. Rep., Aug. 2024. [Online]. Available: https://www.opensand.org/.

[15] The Things Network, *LoRa World Record Broken: 832km/517mi using 25mW*, Online, Apr. 2020. [Online]. Available: https://www.thethingsnetwork.org/article/lorawan-world-record-broken-twice-in-single-experiment-1.

[16] LORIOT, "Hybrid Network Management System for Massive IoT," LORIOT, Tech. Rep., Aug. 2024. [Online]. Available: https://loriot.io/.

[17] 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. DOI: 10.1109/MCOM.001.2000627.

[18] *Low Power Wide Area Network (LPWAN) Market Size is Projected to Reach 6440 Million by 2025 | Valuates Reports*, Online, Aug. 2020. [Online]. Available: https://www.prnewswire.com/in/news-releases/low-power-wide-area-network-lpwan-market-size-is-projected-to-reach-6440-million-by-2025-valuates-reports-817168917.html.

[19] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted ALOHA on LoRaWAN-design, analysis, and deployment," *Sensors*, vol. 19, no. 4, p. 838, Feb. 2019. DOI: 10.3390/s19040838.

[20] R. Harwahyu, R.-G. Cheng, D.-H. Liu, and R. F. Sari, "Fair Configuration Scheme for Random Access in NB-IoT with Multiple Coverage Enhancement Levels," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1408–1419, 2021. DOI: 10.1109/TMC.2019.2962422.

[21] A. Hoglund, D. P. Van, T. Tirronen, O. Liberg, Y. Sui, and E. A. Yavuz, "3GPP Release 15 Early Data Transmission," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 90–96, 2018. DOI: 10.1109/MCOMSTD.2018.1800002.

[22] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of lorawan," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, 2017. DOI: 10.1109/MCOM.2017.1600613.

[23] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of lpwan technologies for large-scale iot deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, 2019, ISSN: 2405-9595. DOI: https://doi.org/10.1016/j.icte.2017.12.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959517302953.

[24] J. Petajajarvi, K. Mikhaylov, A. Roivainen, T. Hanninen, and M. Pettissalo, "On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology," in *2015 14th International Conference on ITS Telecommunications (ITST)*, 2015, pp. 55–59. DOI: 10.1109/ITST.2015.7377400.

[25] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A Study of LoRa: Long Range ; Low Power Networks for the Internet of Things," *Sensors*, vol. 16, no. 9, 2016, ISSN: 1424-8220. DOI: 10.3390/s16091466. [Online]. Available: https://www.mdpi.com/1424-8220/16/9/1466.

[26] J. Haxhibeqiri, A. Karaagac, F. Van den Abeele, W. Joseph, I. Moerman, and J. Hoebeke, "LoRa indoor coverage and performance in an industrial environment: Case study," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8. DOI: 10.1109/ETFA.2017.8247601.

[27] R. Apriantoro, A. Suharjono, K. Kurnianingsih, and I. K. A. Enriko, "Investigation of Coverage and Signal Quality of LoRaWAN Network in Urban Area," in *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, 2020, pp. 326–331. DOI: 10.1109/CENIM51130.2020.9297982.

[28] C. Paternina, R. Arnedo, J. Dominguez-Jimenez, and J. Campillo, "LoRAWAN Network Coverage Testing Design using Open-source Low-Cost Hardware," in *2020 IEEE ANDESCON*, 2020, pp. 1–6. DOI: 10.1109/ANDESCON50619.2020.9272128.

[29] M. A. C. Valencia, F. R. G. Cruz, and R. B. Balakit, "LoRa Transmission System for Weather Balloons," in *2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, 2019, pp. 1–5. DOI: 10.1109/HNICEM48295.2019.9072712.

[30] G. Colavolpe, T. Foggi, M. Ricciulli, Y. Zanettini, and J.-P. Mediano-Alameda, "Reception of LoRa signals from LEO satellites," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 6, pp. 3587–3602, Dec. 2019. DOI: 10.1109/taes.2019.2909336.

[31] L. Fernandez, J. A. Ruiz-De-Azua, A. Calveras, and A. Camps, "Assessing LoRa for satellite-to-earth communications considering the impact of ionospheric scintillation," *IEEE Access*, vol. 8, pp. 165 570–165 582, 2020. DOI: 10.1109/access.2020.3022433.

[32] M. Afhamisis and M. R. Pallattella, "LORSAT Project," Luxembourg Institute of Science and Technology (LIST), Tech. Rep., Aug. 2024. [Online]. Available: https://lorsat.lu.

[33] "A DT Machine Learning-Based Satellite Orbit Prediction for IoT Applications," *IEEE Internet of Things Magazine*, vol. 6, pp. 96–100, 2023. DOI: 10.1109/IOTM.001.2200271.

[34] "Collaborative LEO Satellites for Secure and Green Internet of Remote Things," *IEEE Internet of Things Journal*, vol. 10, pp. 9283–9294, 2023. DOI: 10.1109/JIOT.2022.3223913.

[35] "Low Earth Orbit Satellite Communications for Internet-of-Things Applications," vol. 01, no. 3, pp. 11–13, 2023. DOI: 10.47874/2023:pp;11-13.

[36] G. Canello, K. Mikhaylov, and T. Hänninen, "Communication Perspective of Wildfire Detection and Suppression: A Survey of Technologies, Requirements, and Future Directions," in *2023 15th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2023, pp. 157–164. DOI: 10.1109/ICUMT61075.2023.10333282.

[37] S. Aslam, M. P. Michaelides, and H. Herodotou, "Internet of Ships: A Survey on Architectures, Emerging Applications, and Challenges," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9714–9727, 2020. DOI: 10.1109/JIOT.2020.2993411.

[38] İ. Kahraman, A. Köse, M. Koca, and E. Anarim, "Age of Information in Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 11, no. 6, pp. 9896–9914, 2024. DOI: 10.1109/JIOT.2023.3324879.

[39] IoT Analytics, *Satellite IoT connectivity: Three key developments to drive the market size beyond $1 billion*, Online, Aug. 2022. [Online]. Available: https://iot-analytics.com/satellite-iot-connectivity/.

[40] "LoRaWAN® Specification v1.0.3," LoRaWAN Alliance, Tech. Rep., Jul. 2018. [Online]. Available: https://resources.lora-alliance.org/document/lorawan-specification-v1-0-3.

[41] "LoRaWAN® Specification v1.1," LoRaWAN Alliance, Tech. Rep., Oct. 2017. [Online]. Available: https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1.

[42] L. Tessaro, C. Raffaldi, M. Rossi, and D. Brunelli, "Lightweight Synchronization Algorithm with Self-Calibration for Industrial LoRA Sensor Networks," in *Workshop on Metrology for Industry 4.0 and IoT*, 2018, pp. 259–263. DOI: 10.1109/METROI4.2018.8428309.

[43] L. Beltramelli, A. Mahmood, P. Ferrari, P. Österberg, M. Gidlund, and E. Sisinni, "Synchronous LoRa Communication by Exploiting Large-Area Out-of-Band Synchronization," *IEEE IoT Journal*, vol. 8, no. 10, pp. 7912–7924, 2021. DOI: 10.1109/JIOT.2020.3041818.

[44] C. Ramirez, A. Dyussenova, A. Sergeyev, and B. Iannucci, "LongShoT: Long-Range Synchronization of Time," in *18th ACM/IEEE IPSN*, 2019, pp. 289–300. DOI: 10.1145/3302506.3310408.

[45] "LoRaWAN Application Layer Clock Synchronization Specification v1.0.0," LoRa Alliance, Standard, 2018.

[46] M. Afhamisis and M. Palattella, "SALSA: A Scheduling Algorithm for LoRa to LEO Satellites," *IEEE Access*, vol. 10, pp. 11 608–11 615, 2022. DOI: 10.1109/ACCESS.2022.3146021.

[47] M. A. Al-Muhaya, T. Al-Hadhrami, O. Kaiwartya, and D. J. Brown, "A Comprehensive Comparison in Time-Slotted Frame Protocols in LoRaWAN IoT Technology," in *2023 IEEE Smart World Congress (SWC)*, 2023, pp. 704–710. DOI: 10.1109/SWC57546.2023.10449261.

[48] C. El Fehri, N. Baccour, and I. Kammoun, "A New Schedule-Based Scheme for Uplink Communications in LoRaWAN," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 2815–2829, 2023. DOI: 10.1109/OJCOMS.2023.3321875.

[49] A. Triantafyllou, P. Sarigiannidis, T. Lagkas, and A. Sarigiannidis, "A Novel LoRaWAN Scheduling Scheme for Improving Reliability and Collision Avoidance," in *2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, 2020, pp. 1–4. DOI: 10.1109/MOCAST49295.2020.9200253.

[50] J. Lee, W.-C. Jeong, and B.-C. Choi, "A Scheduling Algorithm for Improving Scalability of LoRaWAN," in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 1383–1388. DOI: 10.1109/ICTC.2018.8539392.

[51] D. Zorbas and B. O'Flynn, "Collision-Free Sensor Data Collection using LoRaWAN and Drones," in *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, 2018, pp. 1–5. DOI: 10.1109/GIIS.2018.8635601.

[52] R. Louati, M. Thaalbi, and N. Tabbane, "Enhancing LoRaWAN Class B: A Pseudo-Random Access Mechanism with Orthogonality Awareness," in *2023 International Wireless Communications and Mobile Computing (IWCMC)*, 2023, pp. 1454–1459. DOI: 10.1109/IWCMC58020.2023.10183191.

[53] K. Q. Abdelfadeel, D. Zorbas, V. Cionca, and D. Pesch, "$FREE$ —Fine-Grained Scheduling for Reliable and Energy-Efficient Data Collection in LoRaWAN," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 669–683, 2020. DOI: 10.1109/JIOT.2019.2949918.

[54] J. Mhatre, A. Lee, and H. Lee, "Frequency Hopping Scheduling Algorithm in Green LoRaWAN: Reinforcement Learning Approach," in *2023 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2023, pp. 216–221. DOI: 10.1109/CSCN60443.2023.10453154.

[55] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, and S. Pollin, "Improving Reliability and Scalability of LoRaWANs Through Lightweight Scheduling," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1830–1842, 2018. DOI: 10.1109/JIOT.2018.2815150.

[56] J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Low Overhead Scheduling of LoRa Transmissions for Improved Scalability," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3097–3109, 2019. DOI: 10.1109/JIOT.2018.2878942.

[57] M. A. Ullah, K. Mikhaylov, and H. Alves, "Enabling mMTC in Remote Areas: LoRaWAN and LEO Satellite Integration for Offshore Wind Farms Monitoring," pp. 1–1, 2021. DOI: 10.1109/tii.2021.3112386.

[58] *The Things Network, ToA calculator, Access Online*, https://www.thethingsnetwork.org/airtime-calculator.

[59] F. A. Tondo, M. Afhamisis, S. Montejo-Sánchez, O. L. A. López, M. R. Palattella, and R. D. Souza, "Multiple Channel LoRa-to-LEO Scheduling for Direct-to-Satellite IoT," *IEEE Access*, vol. 12, pp. 30 627–30 637, 2024. DOI: 10.1109/ACCESS.2024.3368872.

[60] M. Afhamisis, "SALSA packet scheduling algorithm," Luxembourg Institute of Science and Technology (LIST), Tech. Rep., Jan. 2022. [Online]. Available: https://github.com/LIST-LUXEMBOURG/salsa.

[61] CNES, "OpenSAND," Centre national d'études spatiales, Tech. Rep., Aug. 2024. [Online]. Available: https://www.opensand.org/.

[62] *MapBox Studio, Access Online*, https://www.mapbox.com/.

[63] *GEOPY, a Python client for several popular geocoding web services, Access Online*, https://github.com/geopy/geopy (visited on 2021-11-1).

[64] *Skyfield Repository, Access Online*, https://github.com/skyfielders/python-skyfield (visited on 2021-11-1).

[65] *CelesTrak, Access Online*, https://celestrak.com/ (visited on 2021-11-1).

[66] *SALSA Algorithm, Access Online*, https://github.com/list-luxembourg/salsa.

[67] M. Afhamisis, S. Barillaro, and M. R. Palattella, "A Testbed for LoRaWAN Satellite Backhaul: Design Principles and Validation," in *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2022, pp. 1171–1176. DOI: 10.1109/ICCWorkshops53468.2022.9814560.

[68] M. Afhamisis and M. R. Palattella, "Feasibility Study of Synchronization in NTN-LoRaWAN with the REACT Method," in *2024 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2024.

[69] M. Afhamisis, S. Barillaro, and M. R. Palattella, "A Study of LoRa PHY Protocol and LoRaWAN Networks," in *Springer Handbook of Internet of Things*, S. Ziegler, R. Radocz, A. Q. Rodriguez, and S. Nieves Matheu Garcia, Eds. Cham: Springer International Publishing, 2024. [Online]. Available: https://link.springer.com/book/9783031396496.

10 | Appendix

10.1 Publications and Presentations

10.1.1. Journal publications

- M. Afhamisis and M. Palattella, "SALSA: A Scheduling Algorithm for LoRa to LEO Satellites," IEEE Access, vol. 10, pp. 11 608–11 615, 2022. DOI: 10.1109/ACCESS.2022.3146021 [46].
- F. A. Tondo, M. Afhamisis, S. Montejo-Sánchez, O. L. A. López, M. R. Palattella, and R. D. Souza, "Multiple Channel LoRa-to-LEO Scheduling for Direct-to-Satellite IoT," IEEE Access, vol. 12, pp. 30 627–30 637, 2024. DOI: 10.1109/ACCESS.2024.3368872 [59]

10.1.2. Conference publications

- M. Afhamisis, S. Barillaro, and M. R. Palattella, "A Testbed for LoRaWAN Satellite Backhaul: Design Principles and Validation," in 2022 IEEE International Conference on Communications Workshops (ICC Workshops), 2022, pp. 1171–1176. DOI: 10.1109/ICCWorkshops53468.2022.9814560 [67].
- M. Afhamisis and M. R. Palattella, "Feasibility Study of Synchronization in NTN-LoRaWAN with the REACT Method," in 2024 IEEE International Conference on Communications Workshops (ICC Workshops), 2024 [68]

10.1.3. Book chapters

- Z. Zhou, M. Afhamisis, M. R. Palattella, N. Accettura, and P. Berthou, “Pervasive LPWAN Connectivity Through LEO Satellites: Trading Off Reliability, Throughput, Latency, and Energy Efficiency,” in Low-Power Wide-Area Networks: Opportunities, Challenges, Risks and Threats, I. Butun and I. F. Akyildiz, Eds. Cham: Springer International Publishing, 2023, pp. 85–110, ISBN: 978-3-031-32935-7. DOI: 10.1007/978-3-031-32935-7_3. [Online]. Available: https://doi.org/10.1007/978-3-031-32935-7_3 [13].
- M. Afhamisis, S. Barillaro, and M. R. Palattella, “A Study of LoRa PHY Protocol and LoRaWAN Networks,” in Springer Handbook of Internet of Things, S. Ziegler, R. Radocz, A. Q. Rodriguez, and S. Nieves Matheu Garcia, Eds. Cham: Springer International Publishing, 2024. [Online]. Available: https://link.springer.com/book/9783031396496 [69].

10.1.4. Presentations

- M. Afhamisis, M. R. Palattella, "Satellite LoRaWAN Networks", in "Journees LPWAN 2023", Session "Communication avec des satellites pour l’IoT", Session Chair "O. Alphand, LIG", , Grenoble France, 2023, Link "https://journees-lpwan-2023.liglab.fr/".
- M. Afhamisis, M. R. Palattella, "LEO Constellations for Internet of Things: Benefits and Challenges on the Path toward 6G", in "IEEE INGR Workshop - Innovative Ideas for Satellite 6G", Sienna, Italy, 2023.
- M. Afhamisis, M. R. Palattella, "Satellite LoRaWAN Multicast: how feasible is it?", in "IEEE INGR Workshop - Advanced Solutions for 6G Satellite Systems", Track "Edge computing and intelligence for satellite 6G", Topic "Massive IoT via Satellite", 2022.
- M. R. Palattella, M. Afhamisis, "Satellite LoRaWAN: design principles and challenges - lessons learned from the LoRaSAT project", in "Journees LPWAN 2021", Session "Expérimentations et communications satellites", Link "https://journees-lpwan-2021.limos.fr/progprev.html", Online, 2021.

- M. Afhamisis, S. Barillaro, M. R. Palattella, "Simulating a Satellite LoRaWAN Network with ns-3", Session Chair "Stefano Avallone", WNS3 workshops, 2021, Link "https://www.nsnam.org/research/wns3/wns3-2021/program/".
- M. Afhamisis, M. R. Palattella, "Smooth integration of Satellites with LoRaWAN networks for agricultural applications", Lake Como School of Advance Studies, Como, Italy, July, 2021.

10.2 Patents

- Time Synchronization in Low Power Wide Area Networks, 11 Oct 2023, Under Application Number LU505256

10.3 Others

- Spotlight on Young Researchers, June 2022, Accessible online "https://www.fnr.lu/research-with-impact-fnr-highlight/spotlight-internet-of-things-satellites-smart-agriculture/"
- LIST TechDay 2023, presenting LORSAT project, Event Website "https://www.list-techday.eu/"
- My Connectivity Day 2023, presenting LORSAT project, Event Website "https://myconnectivity.lu/"

11 | Acronyms

3GPP 3rd Generation Partnership Project

AoI Age of Information

CAGR Compound Annual Growth Rate

CE Coverage Enhancement

CID Command ID

CP Control Plane

CR Coding Rate

CSS Chirp Spread Spectrum

DC Duty Cycle

DL LoRaWAN Downlink

EIRP Effective Isotropic Radiated Power

ED End Device

EDT Early Data Transmission

EPC Evolved Packet Core

EUI ED Unique Identifier

FCFS First Come First Served

FHDR Frame Header

FHSS Frequency Hopping Spread Spectrum

FM-RDS FM Radio Data System

FNR National Research Fund Luxembourg

gRPC Remote Procedure Call

GEO Geostationary Orbit

GMT Greenwich Mean Time

GNSS Global Navigation Satellite System

GSM Global System for Mobile

HAP High Altitude Platform

HW Hardware

IoT Internet of Things

ISL Inter-Satellite Link

ITU International Telecommunication Union

LEO Low Earth Orbit

LGW LoRaWAN Gateway

LNS LoRaWAN Network Server

LoRa Long Range

LoRaWAN Long Range Wide Area Network

LORSAT Design of LoRaWAN protocol optimisation over SATellite Connection for precision agriculture applications

LPWAN Low Power Wide Area Network

LR-FHSS Long-Range Frequency Hopping Spread Spectrum

L2L-A LoRa to LEO Alternating Channel

L2L-AP LoRa To LEO Alternating and Permutation

MAC Medium Access Control

MEO Medium Earth Orbit

MQTT Message Queuing Telemetry Transport

NB-IoT Narrowband Internet of Things

NPBCH Narrowband Physical Broadcast Channel

NPDCCH Narrowband Physical Downlink Control Channel

NPDSCH Narrowband Physical Downlink Shared Channel

NPRACh Narrowband Physical Random Access Channel

NPSS Narrowband Primary Synchronization Signal

NPUSCH Narrowband Physical Uplink Shared Channel

NSSS Narrowband Secondary Synchronization Signal

NTN Non Terrestrial Networks

NTP Network Time Protocol

OFDM Orthogonal Frequency-Division Multiplexing

OTA Over The Air

PHDR Packet Header

PHDR CRC Packet Header error Detector

PHY PHY Layer

PoC Proof of Concept

PRB Physical Resources Block

QoS Quality of Service

RE Resource Element

REACT Lo(R)aWAN diff(e)renti(a)l time syn(c)hroniza(t)ion

RFU Reserved for Future Usage

SALSA a Scheduling Algorithm for LoRa to LEO SAtellite

SNR Signal to Noise Ratio

SC-FDMA Single-Carrier Frequency Division Multiple Access

SDR Software-Defined Radio

SW Software

TDMA Time Division Multiple Access

TLE Two-Line Element Set

TN Terrestrial Networks

ToA Time on Air

UE User Equipment

UL LoRaWAN Uplink

UP User Plane