SAPIENZA
UNIVERSITÀ DI ROMA

Study of scenarios and experimentation of load balancing algorithms for edge computing over LoRaWAN

Facoltà di Ingegneria dell' Informazione, Informatica e Statistica

Corso di Laurea Magistrale in Engineering in Computer Science

Candidate

Lorenzo Frangella

ID number 1899674

Thesis Advisor

Prof. Ioannis Chatzigiannakis

Co-Advisor

Prof. Domenico Garlisi

Academic Year 2023/2024

Thesis defended on 29 January 2025
in front of a Board of Examiners composed by:

Prof. Giorgio Grisetti (chairman)
Prof. Irene Amerini
Prof. Ioannis Chatzigiannakis
Prof. Riccardo Lazzeretti
Prof. Francesco Leotta
Prof. Andrea Marrella
Prof. Roberto Navigli
Prof. Angelo Spognardi

Study of scenarios and experimentation of load balancing algorithms for edge computing over
LoRaWAN
Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: frangella.1899674@studenti.uniroma1.it

Abstract

The work presented in this thesis directly addresses the critical problem of the lack of datasets containing transmission information in LoRaWAN networks. In the beginning, a general overview of the domain, including the technologies adopted, is depicted, and later on, the whole process of creation of the dataset is analyzed, including some evaluations of the dataset characteristics before and after the network data emulation process.
The dataset, based on the Roman Taxi Dataset, was designed to simulate real-world device mobility patterns and provide a basis for evaluating LoRaWAN network performance. It includes two scenarios: one representing the movements of individual devices and another representing a cluster of devices moving collectively. This dataset is then used as input for the evaluation of the `Edge2LoRa` and `Edge4LoRa` frameworks, which integrate edge computing into LoRaWAN to enhance scalability and performance.
The findings demonstrate that `Edge4LoRa` resource allocation is affected with different ways in the assignment strategy. This work contributes to the advancement of IoT networking by addressing key limitations in traditional LoRaWAN architectures and offering a scalable solution for future applications.

Contents

1 Introduction **1**
1.1 Motivation and Scope . . . 1
1.2 Objectives and Contributions . . . 2
1.3 Structure of the Thesis . . . 2

2 Adopted Technologies **3**
2.1 Low-power Wide-area Network (LPWAN) . . . 3
2.2 LoRa & LoRaWAN . . . 3
2.3 LoRa . . . 4
2.3.1 LoRa Description . . . 4
2.3.2 Lora Modulation Technique . . . 4
2.3.3 Lora Frame Structure . . . 5
2.3.4 Physical Payload . . . 6
2.4 LoRaWAN . . . 7
2.4.1 LoRaWAN architecture . . . 7
2.4.2 LoRaWAN End Devices . . . 8
2.4.3 End Device Activation . . . 8
2.5 NS-3 . . . 10
2.5.1 Key Features . . . 10
2.5.2 Applications . . . 11
2.5.3 Advantages over ns-2 . . . 11
2.5.4 Limitations . . . 11
2.6 Edge2Lora . . . 12
2.6.1 Edge2Lora Architecture . . . 12
2.6.2 Far Edge Devices . . . 13
2.6.3 Edge Devices . . . 13
2.6.4 Cloud . . . 13
2.7 Edge4Lora . . . 13
2.7.1 New Components in Edge4LoRa . . . 14

3 Large-scale Evaluation of LoRaWAN **16**
3.1 Description . . . 16
3.2 Starting Data . . . 16
3.2.1 Dataset Exploration . . . 17
3.2.2 Selected Dataset: Roman Taxi Data . . . 17
3.3 Dataset Description . . . 18

3.3.1 Dataset Overview 18
3.3.2 Data Format 18
3.4 Dataset Preliminary Analysis 19
3.4.1 Initial Observations 19
3.4.2 Comprehensive Distribution Analysis 19
3.5 Dataset Reshaping 22
3.5.1 Data Reduction 22
3.5.2 Scaling the Dataset 22
3.5.3 Snapshot Generation 22
3.6 Gateway Deployment 24
3.6.1 Gateway Deployment Strategy 25
3.6.2 Gateway Placement Methodology 25
3.6.3 Coverage Insights 26
3.7 LoRaWAN Network Emulation 28

4 Evaluation of Experimentation Scenarios 31
4.1 Scenario with a Cluster of devices 33
4.2 Deployment with the Dataset 34
4.2.1 Design of the Dashboard 35

5 Conclusions 37
5.1 Key Takeaways from the Research 37
5.2 Dataset Limitations 38
5.3 Future Works 38
5.4 Closing Remarks 39

Chapter 1

Introduction

The Internet of Things (IoT) has emerged as a transformative technology, enabling interconnected devices to interact with the external environment by collecting data via sensors and performing actions through actuators. Its applications span various domains, including smart cities, healthcare, transportation, and industrial automation. The ultimate goal of IoT systems is to create autonomous networks of devices capable of making decisions independently. This autonomy is increasingly enhanced by advancements in Machine Learning (ML) and Artificial Intelligence (AI), which allow devices to perform complex tasks with high accuracy. However, such tasks often demand substantial computational resources and access to large datasets, challenges that end devices, constrained by limited processing power and storage, are ill-equipped to handle.

To address these challenges, computational workloads can be shifted away from end devices. Initially, the Cloud Computing paradigm offered a solution by enabling heavy processing tasks to be performed remotely, with end devices acting as mere data collectors [1]. However, cloud computing introduces drawbacks such as increased latency and potential security risks due to data transmission over the internet. Recently, the paradigm of Edge Computing has emerged as a promising alternative. Edge Computing [2] processes data near the source, reducing latency, enhancing security, and ensuring scalability in IoT applications.

A critical enabler of IoT networks is Low Power Wide Area Network (LPWAN) technologies such as LoRaWAN. LoRaWAN is based on the LoRa protocol and enables long-range communication with minimal energy consumption. While LoRaWAN provides efficient data forwarding through gateways, its centralized architecture introduces challenges in scalability, latency, and managing mobility in dynamic scenarios. The architecture is primarily a packet-forwarding system, limiting its ability to perform advanced network functions.

1.1 Motivation and Scope

To address the limitations of centralized LoRaWAN networks, Milani et al. [10] proposed the `Edge2LoRa` framework. This architecture integrates edge computing into LoRaWAN networks, enabling a distributed processing paradigm and reducing reliance on centralized cloud servers. While `Edge2LoRa` improves scalability and

processing efficiency, it lacks optimal mechanisms for handling mobility scenarios and dynamic device-to-gateway assignments.

Building upon `Edge2LoRa`, this thesis introduces `Edge4LoRa`, an extended architecture designed to overcome these limitations. A significant contribution of this work is the creation of a realistic dataset derived from the Roman Taxi Dataset, simulating real-world mobility patterns. The dataset serves as the basis for evaluating the performance of `Edge4LoRa`, particularly in scenarios requiring effective resource allocation and load balancing.

1.2 Objectives and Contributions

This thesis focuses on advancing the integration of edge computing within LoRaWAN networks by addressing key challenges in mobility and scalability. The main objectives are:

- To create a realistic dataset that replicates mobility patterns in dynamic environments, enabling detailed analysis of network behavior.
- To implement a scalable simulation environment using the NS-3 simulator for evaluating LoRaWAN network performance under various conditions.
- To develop and test load-balancing algorithms to optimize device-to-gateway assignments, enhancing the efficiency and reliability of `Edge2LoRa` and `Edge4LoRa`.

Key contributions include:

- Constructing a comprehensive dataset that simulates dynamic device movements and interactions with gateways, providing valuable insights into network behavior.
- Analyzing the dataset to identify key characteristics, inefficiencies, and opportunities for improvement.
- Extending the `Edge2LoRa` framework into `Edge4LoRa`, validating its ability to handle large-scale networks with enhanced gateway processing, efficient packet redirection, and improved load balancing.

1.3 Structure of the Thesis

The thesis begins with an overview of IoT technologies, focusing on LoRaWAN, its limitations, and the role of Edge Computing in addressing these challenges. It then introduces the `Edge2LoRa` and `Edge4LoRa` frameworks, which serve as the foundation for this work. The process of dataset creation, along with an evaluation of its characteristics before and after the reshaping process, is detailed. The thesis also describes the visualization tools developed to analyze the effectiveness of the proposed balancing algorithms in the `Edge4LoRa` architecture. Finally, experimental results are presented, demonstrating the scalability and efficiency of the proposed framework, followed by conclusions and suggestions for future research.

Chapter 2

Adopted Technologies

The design and implementation of the `Edge2LoRa` and `Edge4LoRa` frameworks rely on a diverse set of technologies that enable efficient data processing, network communication, and scalability. This chapter provides an overview of the key technologies adopted in this work, highlighting their roles and contributions to the development and evaluation of the proposed systems.

The technologies selected for this research were chosen based on their relevance to the challenges of Low Power Wide Area Networks (LPWANs) and edge computing. These include simulation tools, communication protocols, and data processing frameworks, each playing a critical role in addressing specific aspects of the problem domain. By leveraging established tools and frameworks, the study ensures a robust and reliable foundation for experimentation and innovation.

2.1 Low-power Wide-area Network (LPWAN)

In the IoT domain standard networking transmission technologies and protocols are not the best solution, in such context there is the need to design new protocols to manage the constraints of Iot devices, like small computational power and the need to save as much energy as possible. On the other hand, devices often do not need to exchange many bytes per transmission. Combining data from a wide set of sensors and periodic transmissions from the same device generates a wide quantity of data that must be processed or stored. Low-power Wide-area networks LPWAN are innovative solutions that support the needs of IoT and at the same time take advantage of the low throughput needed. A player in LPWANs is LoRaWAN a network protocol that is widely used in the IoT context.

2.2 LoRa & LoRaWAN

LoRaWAN is one of the most important network protocols in the LPWAN's domain. LoRa (**Lo**ng **Ra**nge) is a modulation technique for radio signals, it is combined with LoRaWAN which is a network protocol, or used to build mesh networks. In this work, the mesh approach is not included, since the Edge2Lora infrastructure is based on LoRaWAN. [8]

2.3 LoRa

2.3.1 LoRa Description

As said above, LoRa is a physical layer protocol, is proprietary, patented by Cycleo in 2014, and then acquired by Semtech. LoRa's main capabilities include long-range communication and low power consumption. The frequency bands used depend on the geographical area, all of them are in the Sub Gigahertz spectrum, like 868 MHz in the European Community and 915 MHz in other regions like the US. The low-frequency spectrum allows LoRa to reach impressive distances in signal transmission and reception in perfect conditions (line of sight and without the presence of radio frequency interferences). An evaluation of transmission capabilities has been evaluated in "Long-range communications in urban and rural environment" [3], considering three different scenarios: Rural, Suburban, and Urban. This shows that the LoRa modulation technique can reach Rural areas more than 5 km with still a high link quality and the transmission range in urban context drops around 2 km. This work also shows how objects in the middle and different elevations of transceivers affect the transmission.

2.3.2 Lora Modulation Technique

The ability of LoRa to reach Large distances in communication is given by the frequency spectrum used, is known that lower frequencies can reach longer distances given their higher permeability of space, but the bit rate offered is small. The spectrum used alone is not the only actor in the performances offered by LoRa, another key component is the Spreading Factor (SF). The signal generated by LoRa is called the Chirp Signal, whose characteristic is a modulation of the frequency that starts from a F_{min} and is linearly increased up to F_{max} (Up Chirp if moves in the other direction is Down Chirp). In order to By chirping the signal over a wider bandwidth, LoRa spreads the energy over a larger frequency range. This increases resistance to interference and noise while improving the signal reception at the gateway.

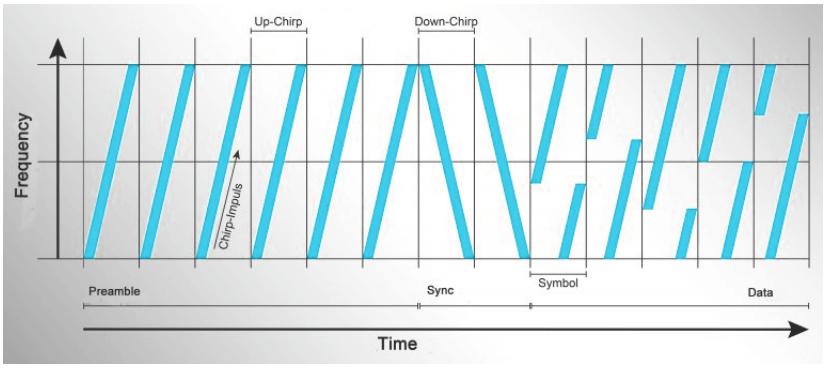

Figure 2.1. Lora Chirping and spreading Factor

The spreading factor values in LoRa are from 7 to 12, as the spreading factor increases, the generated chirp signals are wider in the frequency spectrum and the time slot for each tone increases.

Preamble	Physical Header	Physical Header CRC	Payload	Payload CRC
6 - 65535 bits	1 Byte	1 - 2 Bytes	1 - 255 Bytes	4 Bytes

Table 2.1. Lora Physical Uplink Frame

2.3.3 Lora Frame Structure

Lora Frames are divided in two types: uplink and downlink. Uplink frames are used when a device sends a packet to the gateway, ensuring more reliable and secure communication. Downlink is instead a packet that is sent from the gateway to the device. In the following table, there is an example of the uplink frame

- **Preamble** is a sequence of Up-Chirps signals, followed by two Down-Chirps, its scope is to synchronize the communication between the two devices.
- **Physical Header** contains the basic information needed for the communication like the spreading factor, the coding rate, and bandwidth and presence of CRC that is not provided in the case of Downlink Frame
- **Physical Header CRC** is the integrity check for the header
- **Payload** Contains the Physical payload divided itself into different fields
- **Payload CRC** is the integrity check for the payload (only in Uplink frames)

The one seen above is the explicit format of the packet, the implicit form removes the header and can be used whenever the communication parameters are already known or fixed. Beacons use LoRa radio packet implicit mode for sending time-synchronizing information from gateways to the end devices.

2.3.4 Physical Payload

The Physical payload is the housing of the LoRaWAN packet. LoRaWAN's physical structure is organized as follows:

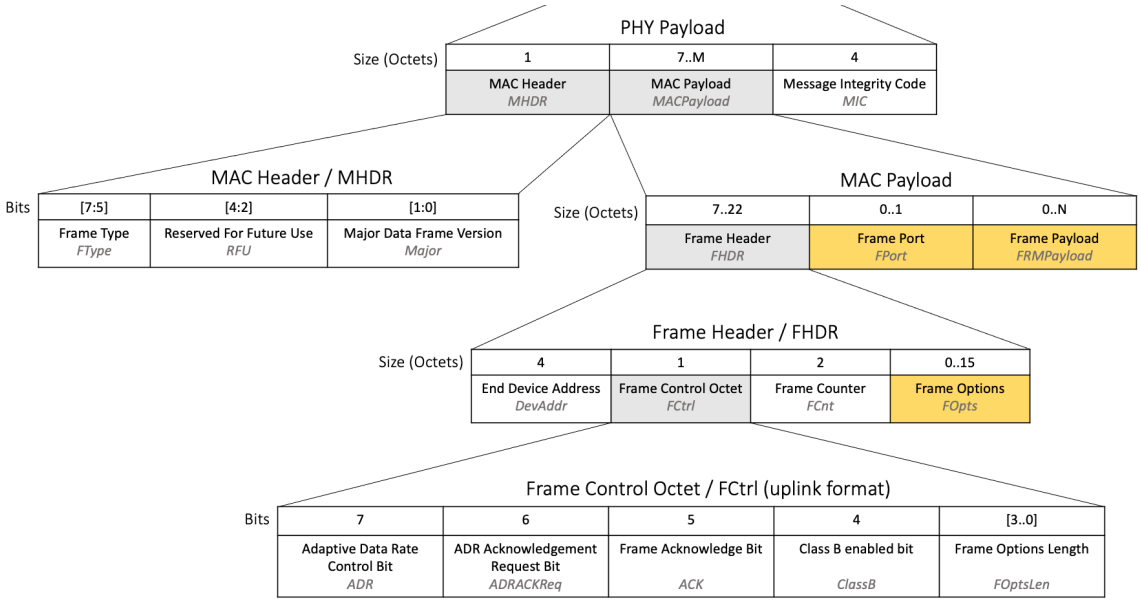

Figure 2.2. Physical LoraWan Payload

The second block in the Frame Header is made to contain all the control information, it is organized in the following structure:

- **Mac Header** is one byte long, and contains the following fields:
  - **Frame Type** containing a 3-bit code used to identify the message type, like a JOIN REQUEST/ACCEPT, or to specify the presence of confirmed/unconfirmed uplink or downlink data.
  - **Reserved for future use (RFU)** field of 3 bits not used.
  - **Major Version** Two bits are used to specify the main version of the protocol.
- **Mac Payload** contains the whole LoRaWAN Transmission and Control information.
  - **Frame Header**
    - **End Device Address** is composed by the End Device address which is a 32-bit address, is not a unique address for the device, but identifies the device in the current network and this field is related to LoRaWAN's network composition. The DevAddr can be split into two parts: 7 bits represent the network address, and the other 25 bits are used for the device. The address is assigned to the device during a phase called Activation which will be analyzed later in this Background section.

    - **Frame Control Octet** a sequence of bits, one for each control option like: Adaptive Data Rate (ADR) Control Bit, ADR Acknowledgment Request Bit, Frame Acknowledgment bit, Class B enabled bit and Frame Options Lenght that occupy the remaining 3 bits.
    - **Frame Counter**
    - **Frame Options**
  - **Frame Port** (optional) identifies the application port
  - **Frame Payload** Variable length, it contains the message data

2.4 LoRaWAN

is a network protocol widely used in LPWANs, it is not only composed of specifications in the definition of packets format and transmission parameters, but it can be seen as a whole Network architecture composed of different devices and actors.

2.4.1 LoRaWAN architecture

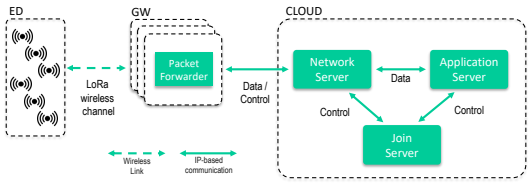

Figure 2.3. LoRaWAN Network Architecture

In figure 2.3 is displayed the infrastructure specification for a LoRaWAN network. End devices communicate with gateways (also called concentrators) using a wireless link with LoRa and then the messages collected from Gateways can be sent with an IP-based communication link (Ethernet, Wifi, or a mobile Cellular network connection) to the cloud, where the second part of the LoRaWan Network is deployed with an Application Server, a Join Server, and a Network Server.

- **LoRaWAN End Devices (ED)** Small devices most of the time, equipped with sensors and battery operated. Sensors are embedded with the device and are periodically used to collect data from the environment. Those data cannot be stored locally so the device sends the data over a LoRa transceiver in broadcast to all the Gateways that it can reach.
- **LoRaWAN Gateways (GW)** Gateways always listens to the LoRa Carrier, to receive messages from EDs. To act as a bridge from the LoRaWAN network to the Internet they are equipped with at least two interfaces: a LoRa Transceiver and an IP-based interface that connects the gateway to the Internet like Wifi, Ethernet, or a 4G/5G connection. Here there is no need to save energy, gateways in some scenarios are connected to the grid where they can collect energy

- **LoRaWAN Network Server (NS)** is the entity that manages the network control information like the Spreading Factor and the Data Rate. The network server is not a physical component, like an Application Server or Join Server. All three are pieces of software with different roles. Since the communication in LoRaWAN is broadcast, is normal that multiple gateways receive the same packet, the network server is the one that checks that a message has not duplicates in the AS.
- **LoRaWAN Join Server (JS)**. Devices in LoRaWAN have to pass an Over The Air Activation (OTAA) Phase to join the network. After the activation, a session key is produced for secure and private communication between the device and the Application Server.
- **LoRaWAN Application Server (AS)**. Is the software component that can access the payload of the messages, the whole communication is encrypted, and other components can access only control information.

2.4.2 LoRaWAN End Devices

End devices are divided into three different classes based on their behavior and functions that support them.

- **Class A**: All the devices support the functions that are included in this class, it supports bi-directional communication by scheduling after each uplink communication two short downlink windows, turning on the transceiver to receive a packet from a gateway. This is the lowest power consumption operation mode for an end device
- **Class B**: Those devices allow more receiving slots, adding receiving slots scheduled (in addition to the random receiving slots of Class A)
- **Class C**: Devices that are always listening to the carrier stop the reception only when a transmission is needed. Due to this behavior is the most energy-consuming class.

2.4.3 End Device Activation

The following part describes an overview of the activation phase in 1.1 version of LoRaWAN. Activation of devices can be performed with two different methods, using a ABP activation method, hard-coding the keys in the device. A Device that joins the LoRaWAN network has to be configured and activated through an activation phase called Over-The-Air. To understand better the Activation procedure that uses several identifiers and parameters, the following identifiers and codes are used:

- **JoinEUI** is a 64-bit identifier and is a unique identifier for the network, it must be already loaded inside the devide before activation.
- **DevEUI** is the unique identifier for the device, same format of JoinEUI and also this must be inside the device in activation phase.

- **NwkKey** and **AppKey** are AES-128 root keys, specific for the end device, assigned during fabrication.

During the OTAA activation of an end device, the NwKey is used to derive the following session keys: NwkSIntKey , SNwkSIntKey and NwkSEncKey. AppKey is used to derive the AppSKey session key.

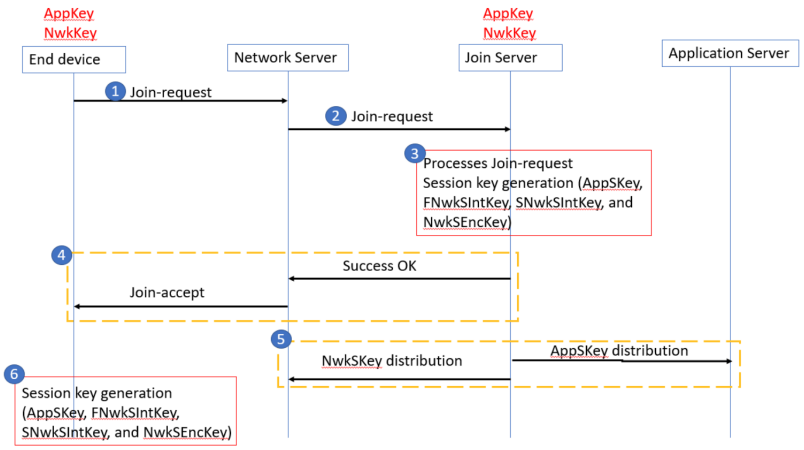

Figure 2.4. LoRaWAN 1.1 Activation procedure

In the above figure there is the communication schema between end device, network server, join server and application server. The Activation procedure is divided into the following steps:

1. At the beginning the End Device must share with the Join Server through a Join Request the JoinEUI the message is always sent to the Network Server

2. The Network Server forwards the Join-request message to the corresponding Join Server. The Join Request is composed by: JoinEUI, DevEUI and a nonce

3. The Join Server processes the received Join Request and generates all the session keys needed.

4. The Network server creates a Join Accept message, including the following fields: Join Nonce used from the end device to generate all the session keys, NetID an identifer for the network, Dev Address identifying the Device in the nework, and others communication parameters.

5. The Join Server sends the AppSKey to the Application server and the three network session keys (FNwkSIntKey, SNwkSIntKey, and NwkSEncKey) to the Network Server

6. The end-device decrypts the Join-accept message using AES encrypt operation. The end device uses AppKey, NwkKey, and JoinNonce to generate session keys.

Once all those steps are done, the device is activated, storing the following information: DevAddr, FNwkSIntKey, SNwkSIntKey, NwkSEncKey and AppSKey.

The DevAddr is the devices identifier in the network, all the others are session keys. Each Sessio Key has its own role:

- **FNwkSIntKey** is used by the end device to calculate the MIC (partially) of all uplink data messages for ensuring message integrity.
- **SNwkSIntKey** is used by the end device to calculate the MIC (partially) of all uplink data messagse and calculate the MIC of all downlink data messages for ensuring message integrity
- **NwkSEncKey** is used to encrypt and decrypt the payloads with MAC commands of the uplink and downlink data messages for ensuring message confidentiality.
- **AppSKey** is used by both the Application Server and the end device to encrypt and decrypt the application data in the data messages for ensuring message confidentiality

2.5 NS-3

`ns-3` is a widely used discrete-event network simulator designed for research and educational purposes. It provides a robust framework for simulating various types of networks, enabling researchers to model and analyze network behaviors in a controlled and reproducible environment. Developed as an open-source project, `ns-3` is written in C++ with Python bindings, allowing for flexibility in simulation scripting and analysis.

2.5.1 Key Features

- **Realistic Models:** `ns-3` offers a wide range of realistic network models, including wireless, wired, and hybrid network configurations. These models simulate the behavior of real-world protocols, devices, and channels, providing insights into network performance.
- **Scalability:** The simulator supports large-scale simulations, making it suitable for testing scenarios with thousands of nodes, including those in wireless sensor networks, IoT, and data centers.
- **Flexibility:** With its modular architecture, `ns-3` allows users to extend or modify existing models and integrate custom protocols or features tailored to specific research needs.
- **Integration with Real Systems:** `ns-3` can interact with real hardware through emulation, enabling researchers to test their designs in a hybrid simulation-real environment.
- **Rich Analysis Tools:** The simulator includes built-in tools for tracing, logging, and visualizing network behaviors, facilitating detailed performance evaluation.

2.5.2 Applications

`ns-3` has been employed extensively in various domains, including:

- **Wireless Networks:** Simulating Wi-Fi, LTE, 5G, and LoRaWAN networks.
- **IoT and Edge Computing:** Modeling large-scale IoT deployments and edge computing scenarios.
- **Protocol Development:** Testing and validating new networking protocols under controlled conditions.
- **Education:** Teaching networking concepts through hands-on simulations.

2.5.3 Advantages over ns-2

`ns-3` is the successor to `ns-2` and offers several improvements:

- A cleaner and more modular architecture for easier development and maintenance.
- Support for modern networking technologies and protocols.
- Improved performance and scalability, making it suitable for contemporary research challenges.

2.5.4 Limitations

Despite its advantages, `ns-3` has certain limitations:

- The learning curve can be steep for new users unfamiliar with C++ or network simulation.
- Some advanced features may require significant effort to implement or configure.
- The simulation of certain real-world behaviors, such as highly dynamic or heterogeneous environments, may require custom model development.

2.6 Edge2Lora

The starting point of this work is Edge2Lora, an architecture built on LoRaWAN that enables edge processing and solves some of LoRaWAN's inefficiencies. Edge2Lora enhances LoRaWAN's capabilities, increasing performance in scalability and reducing latency in data processing. Edge2Lora architecture is placed on top of existing LoRaWAN architecture, ensuring backward compatibility with legacy devices that do not support Edge2Lora.

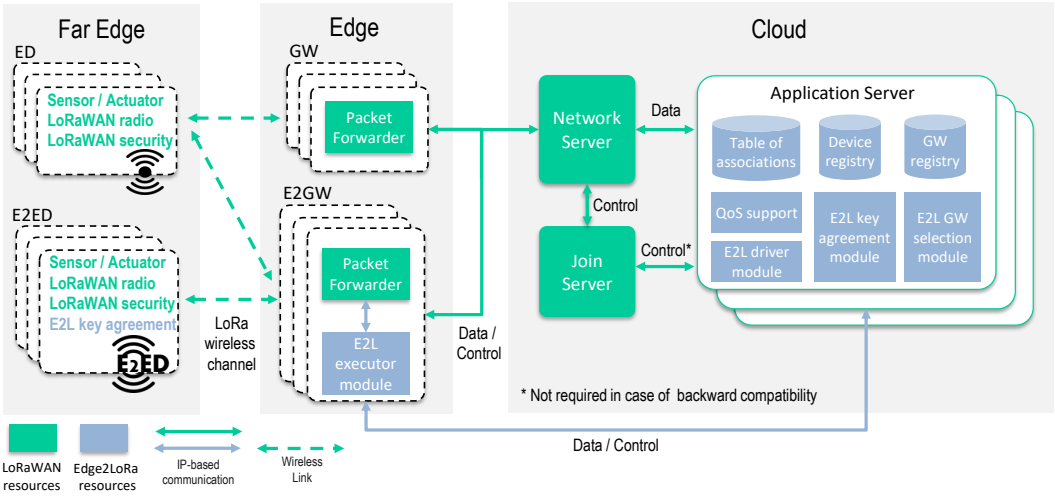

Figure 2.5. Edge2Lora architecture schema including LoRaWAN Legacy devices

Figure 2.5 provided in the description of Edge2Lora published by Milani et al. [10] shows that Legacy devices and gateways can co-operate in the same network, with `Edge2Lora` devices and gateways. E2GWs are the Gateways in `Edge2Lora`, and thoser are also able to behave like a normal LoRaWAN gateway, forwarding the packet to the application server.

2.6.1 Edge2Lora Architecture

The `Edge2LoRa` framework is composed of several fundamental components that enable its functionality and performance. These components include the device registry, the group key establishment mechanism, and the sensor data stream processing system. Together, these elements co operate to ensure efficient and secure communication within the network.

The framework introduces specific device classifications to represent its compatibility. The *End Devices* (EDs) compatible with the `Edge2Lora` framework are referred to as **E2ED**, while the Gateways (GWs) designed to operate within the `Edge2Lora` framework are referred to as **E2GW**.

Each **E2ED** operates by transmitting sensor data streams, which are received and processed by an assigned **E2GW**. The **E2GW** acts as the intermediary responsible for executing data processing tasks, ensuring that the transmitted data streams are handled efficiently and securely. This architecture ensures that the computational load is distributed, enhancing the scalability and responsiveness of the system.

The main role of `Edge2Lora` framework is to manage the presence of duplicate packets. This already happens in LoRaWAN when a packet is delivered duplicated to several gateways and the best one (in terms of quality of signal is chosen). In `Edge2Lora` the important thing is to avoid a duplicate processing/storage of data from multiple gateways. The gateway that now do not behaves only as packet forwarder, has an active role in computing and is not useful repeating the processing of data on multiple gateways.

The architecture of the `Edge2Lora` framework is illustrated in Figure 2.5. Its components are categorized into three classes, based on their position within the network and their computing power.

2.6.2 Far Edge Devices

Far Edge Devices are designed to operate with minimal computational resources. These devices are capable of running small programs without requiring significant processing power. Examples of suitable hardware for far edge devices include *Arduino* boards or *Heltec Cubecell* modules. The primary role of far edge devices is to collect data using their sensors and transmit it via LoRa. These devices are typically battery-operated to facilitate deployment in remote or resource-constrained environments.

2.6.3 Edge Devices

Edge Devices are more powerful than far edge devices and are generally connected to a reliable energy source, such as the electrical grid. Unlike far edge devices, which are often battery-operated, edge devices can handle more computationally intensive tasks. A suitable hardware example for edge devices is a *Raspberry Pi* equipped with a LoRa antenna, which allows it to receive LoRaWAN packets from far edge devices. Edge devices act as gateways, bridging the far edge devices to higher-level processing systems.

2.6.4 Cloud

The final class of devices in the `Edge2LoRa` framework is the *Cloud*. This layer is responsible for running computationally heavy software components and managing centralized tasks for the entire network. Key functionalities performed at this level include the key management system and the device-gateway association module. The cloud acts as the backbone of the framework, enabling efficient operation and management of the devices within the network.

2.7 Edge4Lora

Starting from `Edge2LoRa`, `Edge4LoRa` is a derived architecture that builds upon the original implementation. It incorporates several additional modules to enhance data processing capabilities at the gateway level. The `Edge4LoRa` framework inherits all the components of `Edge2LoRa` while introducing a `Map/Reduce Engine` for data

processing and a new interface designed to manage packet redirection and support load balancing between gateways.

2.7.1 New Components in Edge4LoRa

At the gateway layer, `Edge4LoRa` introduces three new components:

- **Message Broker:** Facilitates the exchange of frames using *MQTT* as the communication protocol, ensuring reliable message delivery.
- **Apache Spark Engine:** Provides a robust environment for processing large-scale data, enabling complex computations directly at the gateway.
- **Edge4LoRa Parser:** A dedicated message parser designed to handle incoming data streams efficiently, ensuring seamless integration and preprocessing of sensor data.

These additions significantly improve the framework's ability to process, analyze, and manage data at the edge, reducing the dependency on cloud resources and enhancing overall system performance.

In Figure 2.6, there is a representation of the whole architecture of `Edge4Lora`, which is organized, like `Edge2Lora`, in a three-layer architecture: the Cloud Layer, the Edge Layer, and the Far Edge Layer. The figure illustrates how the components mentioned earlier are organized and placed within the entire architecture.

Gateways now include a new set of components:

- The **Message Parser** is responsible for opening the content of the packet to extract the information contained inside.
- Each Gateway also includes:
  - A **SPARK Module**, capable of performing a set of map-reduce operations on the data.
  - A **Local Broker**, which is an MQTT server that facilitates the exchange of both control information and data between gateways.
  - A **Edge4Lora Parser**, which is a component in charge of unpacking data contained in the LoRaWAN frame.

The figure also describes three possible scenarios that can occur when a packet is delivered using LoRaWAN:

1. **Processing on Receiver:** In this case, the packet is received by the Gateway responsible for processing the data contained in the delivered message. The packet is first received by the `Packet Forwarder`, then unpacked by the `Edge4Lora Parser`, and subsequently published on the designated MQTT topic. If the topic corresponds to the receiving Gateway, the data is processed directly, and the result is published on another topic that is accessible to the `Edge4Lora Sink`.

2. **Processing on Another Gateway:** If the Gateway that receives a packet is not responsible for processing the data within that frame, it behaves similarly to the previous case. However, in this scenario, the MQTT topic for the message is not associated with the receiving Gateway but with another Gateway in the network.

3. **Legacy Device:** If a delivered message originates from a legacy device that does not support `Edge4Lora` or `Edge2Lora`, the data from the device will not be published on the processing topic. Instead, it will only be forwarded directly to the application server.

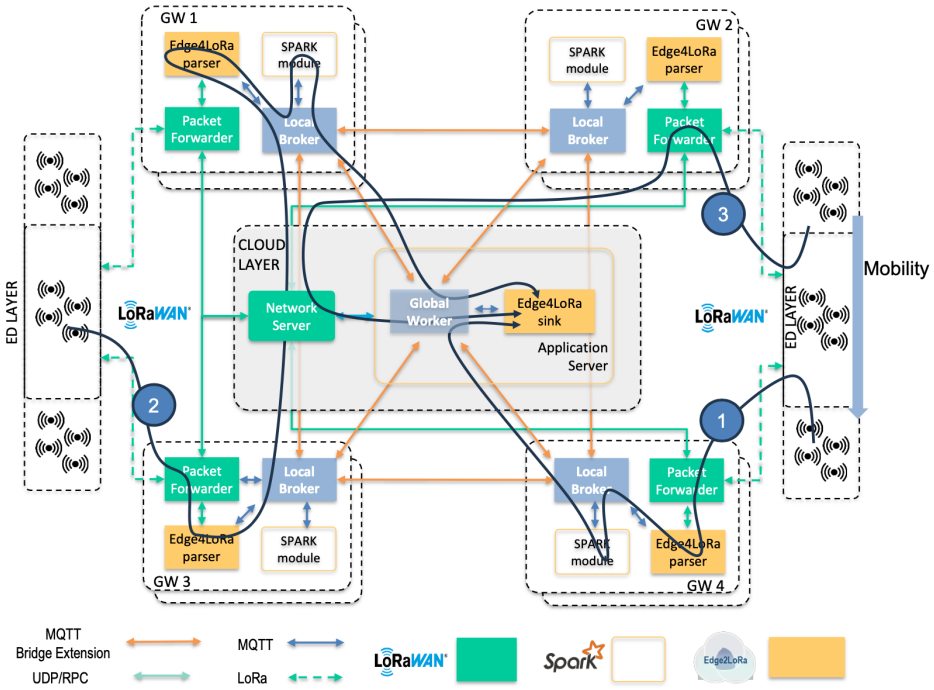

Figure 2.6. Edge4Lora architecture

Chapter 3

Large-scale Evaluation of LoRaWAN

3.1 Description

The primary objective of this research is to thoroughly analyze and understand the behavior of the `Edge2Lora` system when deployed in a large-scale, dynamic scenario. In such an environment, a significant number of devices move across a defined geographic area, altering the gateways capable of receiving their LoRaWAN messages as they change locations.

A notable limitation of the current `Edge2Lora` implementation is the absence of an intelligent component for managing device-to-gateway assignments. This means that when a gateway becomes unavailable—whether due to range limitations or other factors—the connection is severed, and the responsibility for processing the device's packets is passed to the next available gateway. While this approach is functional, it does not optimize the system for efficiency or reliability in scenarios with high device mobility or dense gateway deployment.

The literature surrounding network management and optimization in LoRaWAN systems is relatively sparse, particularly in the context of data-driven analyses of gateway assignments in mobile scenarios. This gap in existing research further highlights the novelty and necessity of this work.

To address this, the study incorporates the creation of a custom dataset that emulates a realistic network scenario. This involves leveraging location-based information and enhancing it with additional datasets to produce a robust foundation for testing and analysis. By generating a dataset tailored to the unique demands of this research, the study aims to provide valuable insights into `Edge2Lora`'s performance under varying conditions, offering a pathway toward optimizing device-to-gateway interactions in LoRaWAN networks.

3.2 Starting Data

Given the absence of data in the literature regarding network information, it was necessary to create or identify a dataset to serve as a starting point. This dataset needed to include a set of devices that move, not randomly, but following a specific

and meaningful pattern to provide a clear and realistic representation of the world. After analyzing different options, a promising solution was identified: leveraging data that documents the movement of taxis in a city, as these offer interesting and well-defined movement patterns.

3.2.1 Dataset Exploration

To identify the most suitable dataset, several options were evaluated. Each dataset was scrutinized based on its accuracy, completeness, and relevance to the research objectives.

New York Taxi Dataset

The first dataset considered was one containing information on taxi movements within the city of New York [23]. Upon quick analysis, however, this dataset was deemed unsuitable. The key limitation was the lack of precise positional data; taxi movements were described only in terms of starting and destination areas. These areas, defined as neighborhoods in New York City, covered approximately 3 square kilometers each. Such coarse granularity made it impossible to accurately model the continuous movement of devices, which is a critical requirement for this study.

Porto Taxi Dataset

The second dataset evaluated was a year-long collection of data from taxis operating in the city of Porto [19]. This dataset provided more detailed information, including the positions of taxis during individual trips. However, it did not continuously collect positional data, instead organizing the information into sets of taxi trips. While this dataset included additional metadata, such as the type of trip, its structure introduced a significant drawback: the potential loss of positional information between consecutive trips. This limitation made it challenging to reconstruct a continuous movement trajectory for analysis.

Roman Taxi Dataset

The third dataset, published by Bracciale et al. [6], was ultimately selected for use in this research. This dataset contains detailed positional information for taxis operating in Rome during February 2014. Each taxi recorded its GPS coordinates at intervals of 7 to 15 seconds over the course of a month. This high frequency of data collection provided a more complete representation of taxi movements.

3.2.2 Selected Dataset: Roman Taxi Data

The Roman Taxi dataset proved to be the most suitable choice for creating the scenario required in this work. With positional data collected at regular intervals and an average GPS accuracy of 20 meters, this dataset enabled the construction of a realistic and detailed representation of device mobility. The dataset includes 320 taxis, and the positional data reflects working hours (approximately 8 hours per day) rather than 24/7 monitoring.

This dataset forms the foundation of the proposed work, providing the necessary mobility patterns to emulate a real-world network scenario. By leveraging this detailed data, the study effectively bridges the gap left by the lack of similar datasets in the literature.

3.3 Dataset Description

The dataset [6], published by Bracciale et al., provides detailed positional information for taxis operating in Rome. It is an essential resource for this study as it captures real-world mobility patterns with a high degree of granularity. The dataset comprises records of taxi positions collected at intervals averaging between 7 to 15 seconds, ensuring a detailed temporal resolution of movements.

3.3.1 Dataset Overview

The dataset includes a total of 320 taxis. However, the location information was not collected continuously over a 24-hour period. Instead, data collection appears to have been limited to the hours when drivers were actively operating their vehicles, which, upon analysis, is approximately 8 hours per day per taxi.

The collection period spans from February 1, 2014, to March 1, 2014, providing one month of continuous data. GPS technology was employed to capture positional information, offering an accuracy of approximately 20 meters. This level of precision is sufficient for studying urban mobility patterns and their implications for network behavior.

It is important to note that the dataset description does not explicitly state the collection rate. However, based on the data analysis, it can be inferred that positional records were stored only during active driving periods, making the dataset representative of typical urban transportation patterns rather than continuous tracking.

3.3.2 Data Format

The dataset is organized in a single CSV file, structured in rows, with each row representing an individual data point. The format ensures that the data is both easy to process and analyze.

Each row contains the following fields:

- **Timestamp:** The exact time at which the positional data was recorded.
- **Taxi ID:** A unique identifier assigned to each taxi, allowing for differentiation and traceability of individual vehicle movements.
- **Location:** The recorded geographic position of the taxi, represented as `POINT(latitude, longitude)`.

The positional data is sorted chronologically by timestamp, ensuring that the movement traces of each taxi can be reconstructed seamlessly. This organization facilitates both individual and collective analysis of taxi mobility, making the dataset particularly well-suited for use in this study.

By providing a high-frequency and well-structured representation of urban mobility, this dataset forms the backbone of the simulation scenario. Its detailed and accurate data enables the emulation of realistic network conditions, critical for evaluating `Edge2Lora`'s behavior in dynamic environments.

3.4 Dataset Preliminary Analysis

To gain deeper insights into the dataset and to extract meaningful information, extensive preprocessing was performed. This step was essential to better understand the positional data, analyze taxi movements, and explore their active times and patterns.

3.4.1 Initial Observations

The first key observation pertained to the distribution of taxis across the metropolitan area of Rome. According to the dataset description, taxis that collected positional data primarily operated in the central parts of Rome. However, further analysis revealed that their movements were not confined to this area.

To illustrate this, the movement of a single taxi over the course of a day was analyzed. Figure 3.1 provides a visualization of this movement pattern.

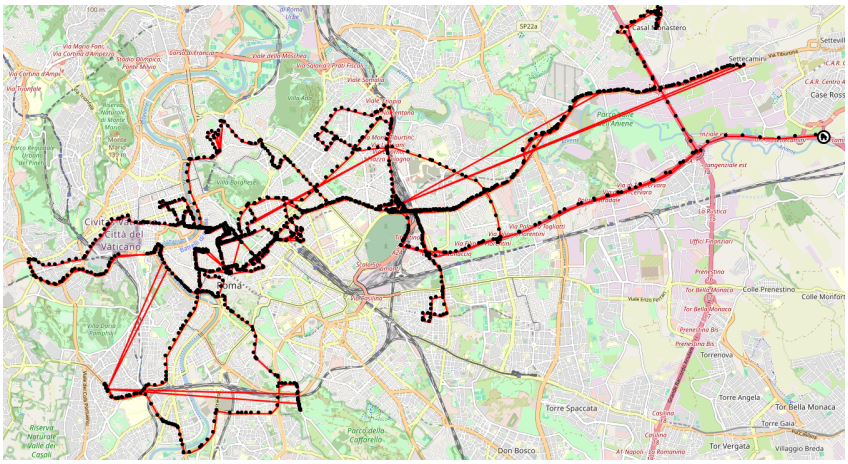

Figure 3.1. Movement of a Taxi during a Day

As shown in the figure, while the taxi's movement is concentrated in the central part of Rome, it occasionally extends to areas outside the city center. This behavior is not unique to this particular taxi; similar movement patterns were observed for other taxis. In some instances, taxis traveled tens of kilometers away from the central metropolitan area.

3.4.2 Comprehensive Distribution Analysis

To obtain a holistic view of taxi distribution, a heatmap was created using a random sample of 1 million positional data points drawn from the dataset. This sampling was performed to ensure computational efficiency while maintaining a

true representation of the overall dataset, which comprises approximately 21 million entries.

The resulting heatmap is shown in Figure 3.2.

Figure 3.2. Heatmap of Taxi Positions

The heatmap reveals several key insights:

- The highest density of taxi positions is concentrated in the center of Rome.
- Taxis frequently travel along major roadways and highways, including the G.R.A Highway and significant consular roads.
- There is notable activity at major transportation hubs such as Fiumicino Airport, Ciampino Airport, and the port of Civitavecchia.

These findings demonstrate that taxi movements are influenced by both urban traffic patterns and the geography of Rome, extending beyond the city center and following prominent routes. This broad distribution of movements makes the dataset an ideal basis for studying dynamic LoRaWAN scenarios.

In Figure 3.1, it is evident that while the GPS data is generally accurate, it is not without imperfections. A few positional anomalies are noticeable, where the taxi appears to "teleport" to a far-off location and then quickly return to its original path. These errors in position are likely due to GPS inaccuracies or transient data collection issues.

Interestingly, these erroneous data points were not excluded during the preparation of the dataset. This decision was made to preserve the integrity of the raw dataset and because such anomalies can be embedded in the payload of messages sent

by the taxi. Identifying and analyzing these anomalies could serve as a valuable task for future work, offering insights into handling and mitigating positional inaccuracies in similar datasets.

One possible method to address these errors was investigated. A straightforward approach involves setting a threshold based on the distance between consecutive positions and the average speed calculated between them. By applying such a method, unrealistic positional changes could be filtered out effectively, ensuring a cleaner dataset for certain applications.

It is also important to note that devices do not continuously register positional data. The dataset description does not specify how the GPS positions were collected. However, the description mentions that the `getAccuracy` function from Android's `LocationManager` objects was used for collecting GPS information. This suggests that the coordinates were likely collected via the smartphones of taxi drivers.

This detail adds a layer of context to the dataset, as it implies that the quality and frequency of the GPS data could depend on the drivers' smartphone configurations and the conditions under which they operated. Despite these limitations, the dataset remains a rich resource for studying real-world mobility patterns and their implications for LoRaWAN network behaviors.

3.5 Dataset Reshaping

The dataset in its original form was not optimal for the purposes of this work. Although it contained a substantial amount of data regarding the positions of taxis over the course of a month, its utility for evaluating the capabilities of `Edge2Lora` in massive scenarios was limited. This limitation arose due to the relatively small number of taxis included—320 vehicles, which were not consistently active throughout the dataset. Such a scale was insufficient for testing `Edge2Lora`'s performance in handling high-density scenarios and ensuring effective device handover management. Tasks described in the following section are performed unsing Python [12], Pandas[11], Numpy[14] and for data visualization is used Folium [13]

3.5.1 Data Reduction

The first step in reshaping the dataset involved reducing the granularity of the positional data. The original dataset recorded the location of each taxi at intervals ranging from 7 to 15 seconds, which was far too frequent for the intended purpose and not comfortable to emulate network information. In practical applications, a device typically transmits its collected location data every few minutes, rather than every few seconds. Therefore, this level of granularity was unnecessary and would only add computational overhead without providing additional meaningful insights. At the same thime the removed location information can be used in the payload of the messages sent as application information.

To align with realistic operational scenarios, the positional data was sampled at 5-minute intervals. This adjustment not only reduced the dataset size but also better reflected the expected behavior of devices in a LoRaWAN network.

3.5.2 Scaling the Dataset

To enable testing in more demanding scenarios, the dataset needed to represent a larger number of devices. Starting from the original dataset of 320 taxis moving over 30 days, the data was compressed into a single day of activity. The same taxi in different days is considered each day a new taxi, with a new identifier of the form: OriginalTaxiId-Day. This transformation involved aggregating the data and sampling geographical coordinates every 5 minutes for each taxi. The result was a dataset more suited for evaluating `Edge2Lora`'s scalability and handover capabilities.

3.5.3 Snapshot Generation

Each 5-minute time interval in the dataset is referred to as a *snapshot*. The reshaped dataset consists of 288 non-overlapping snapshot files, representing the 24-hour period of a single day divided into 5-minute intervals.

For each snapshot, only the taxis that were active—those that had registered at least one location during that interval—are included in the corresponding snapshot file. To simplify the representation, each active taxi is associated with a single pair of coordinates, randomly chosen from the set of locations registered by the taxi during that specific interval.

This snapshot-based approach ensures that the dataset is both manageable and representative of a realistic, large-scale deployment scenario. It provides a practical basis for evaluating `Edge2Lora`'s ability to handle dynamic, high-density networks effectively.

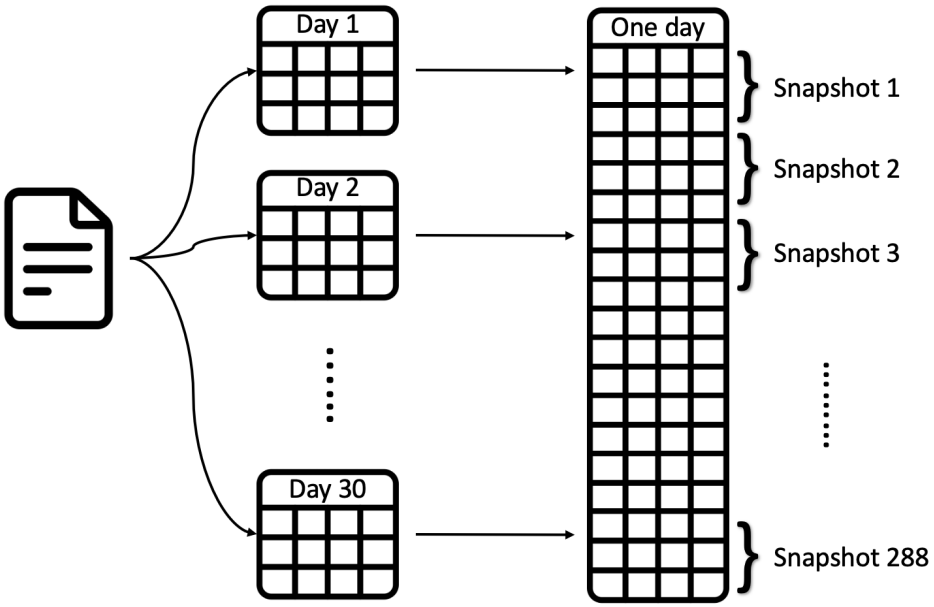

Figure 3.3. Dataset Reshaping Schema

Since drivers are not always driving their vehicles, a taxi is not always active, and taxis do not operate every day of the week. To ensure consistent service, taxis are organized into working shifts of 8 hours each. These shifts are designed to guarantee that a taxi service is available at all times of the day, but drivers are not equally distributed among the three shifts. Moreover, the number of active drivers can fluctuate, making it necessary to extract information regarding the number of taxis active during each hour of the day.

To address this, the dataset, which has already been divided into different snapshots, allows for easy extraction of the number of active taxis during each hour. By analyzing the timestamps associated with each taxi's location, it is possible to count the number of taxis that are active and available during each specific time interval. This information provides a clearer understanding of the operational dynamics of the fleet and the variations in taxi availability throughout the day.

Another important aspect of the dataset is the average moving speed of the taxis throughout the day. The speed at which taxis move can provide valuable insights into network behavior. For example, the moving speed can help determine how often a device (taxi) changes its geographical area or moves between different coverage zones. This information can be critical in deciding the number of gateways that should be deployed in the network, as faster-moving devices may require a more

frequent reassignment of gateways to maintain consistent connectivity.

These two key pieces of information—number of active taxis per hour and average moving speed—are essential in understanding the overall network dynamics and can help optimize the `Edge2Lora` system. Analyzing these factors allows for better predictions of network load and informs decisions on gateway placement, capacity planning, and handover strategies. The following figure illustrates these insights.

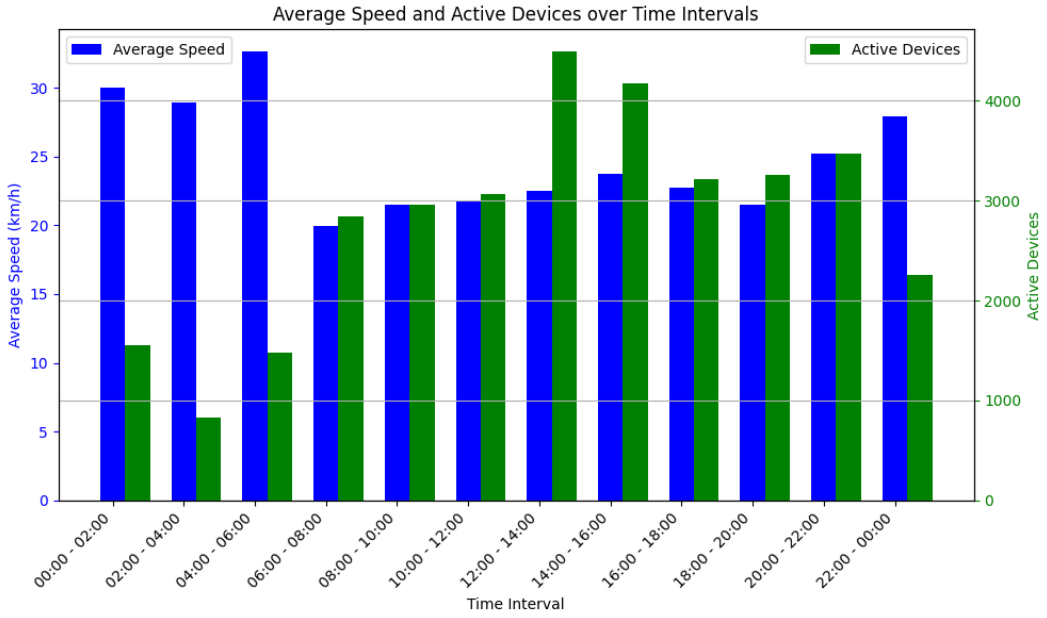

Figure 3.4. Active Taxis and Speed

The figure 3.4 highlights something that could also be imaged before, the number of active taxis is not fixed during the day but it varies, having its lowest values during the night and the highest between 12:00 and 14:00. The average moving speed has an opposite trend, reaching its highest values during the night and dropping during the day.

As a a result of compressing the dataset in a single day, the number of active taxis increased from a maximum of 320 (on the assumption that all taxis are active together) to a mean value of active devices between 2000 and 3000, which is much more interesting than before for performing tests in a massive scenario.

3.6 Gateway Deployment

In the initial phase of this study, the primary focus was on the mobility patterns of the end devices, represented by taxi cabs with detailed location data. However, to fully simulate the network scenario and analyze the behavior of `Edge2Lora`, it was essential to incorporate the second critical component: the strategic placement of LoRaWAN gateways throughout the metropolitan area of Rome.

3.6.1 Gateway Deployment Strategy

The placement of gateways was informed by multiple factors, including the density of taxi movement, the geographic layout of the city, and the configurations provided by an existing LoRaWAN network provider. To simulate varying network conditions and scales, three different deployment scenarios were designed with varying numbers of gateways:

- **Small Deployment (3 Gateways):**
  - Focused on covering the core central areas of Rome.
  - Ideal for studying low-density gateway configurations and their impact on coverage and handover performance.
- **Medium Deployment (20 Gateways):**
  - Expanded coverage to include the broader central area of Rome.
  - Provides insights into the system's scalability in moderately dense configurations.
- **Large Deployment (50 Gateways):**
  - Aims to achieve comprehensive coverage across the entire metropolitan area.
  - Particularly valuable for analyzing `Edge2Lora`'s performance in managing device connections across a dense and geographically expansive network.

3.6.2 Gateway Placement Methodology

The deployment of gateways was not an exact replication of the provider's existing network. Instead, it followed the density and key areas identified by the provider as optimal, adapting these configurations to align with the mobility patterns of the taxi dataset.

Key considerations in the placement of gateways included:

- **Taxi Density Distribution:** Areas with higher concentrations of taxi movement, as revealed by the dataset analysis, were prioritized for gateway placement.
- **Geographic Relevance:** Important roads, highways, and urban hubs such as the airports (Fiumicino and Ciampino) and the port of Civitavecchia were specifically targeted in the largest deployment scenario.
- **Provider Insights:** The areas deemed critical by the provider were used as a reference, ensuring a realistic and practical deployment model.

3.6.3 Coverage Insights

The varying numbers of gateways led to distinct coverage patterns:

- **3 Gateways:**
  - Limited coverage concentrated within the central part of Rome.
  - Designed to represent minimal network infrastructure and assess its challenges.
- **20 Gateways:**
  - Expanded coverage around the central area but still left out the peripheral regions.
  - A practical balance between infrastructure cost and coverage.
- **50 Gateways:**
  - Achieved near-complete coverage of Rome's metropolitan area.
  - Allowed for a detailed evaluation of `Edge2Lora`'s behavior in a densely interconnected environment.

This tiered deployment strategy ensures that the study evaluates `Edge2Lora`'s adaptability and efficiency under diverse network configurations, ranging from minimal to extensive coverage.

In Figure 3.5 can be seen how the gateways have been positioned in the city of Rome, the presented picture contains the Gateways included in the scenario with 50 Gateways. Starting from the scenario with 2 gateways, that are positioned in the center of the city. As the number of gateway increases the distance from the center of Rome also is higher.

Figure 3.6 presents an estimation of the coverage radius of LoRaWAN gateways, based on a model that decreases the coverage radius as the distance from the center of Rome increases. The values for the reference coverage radius are derived from existing literature, particularly the study presented in [3]. In rural environments, the range of LoRa transmissions can extend over several kilometers, which is characteristic of the technology's ability to cover large areas with minimal infrastructure.

In addition, another study that investigates LoRa transmission power, presented in [5], suggests that in rural areas, LoRaWAN can achieve transmission ranges of up to approximately 19 kilometers. However, this is not reflective of urban environments such as Rome. In more urbanized or suburban areas, such as those surrounding the city, the LoRa transmission range tends to be significantly reduced. For these areas, the estimated coverage radius is fixed at around 6 kilometers. This difference underscores the impact that urban density, infrastructure, and environmental factors have on the performance of LoRaWAN networks.

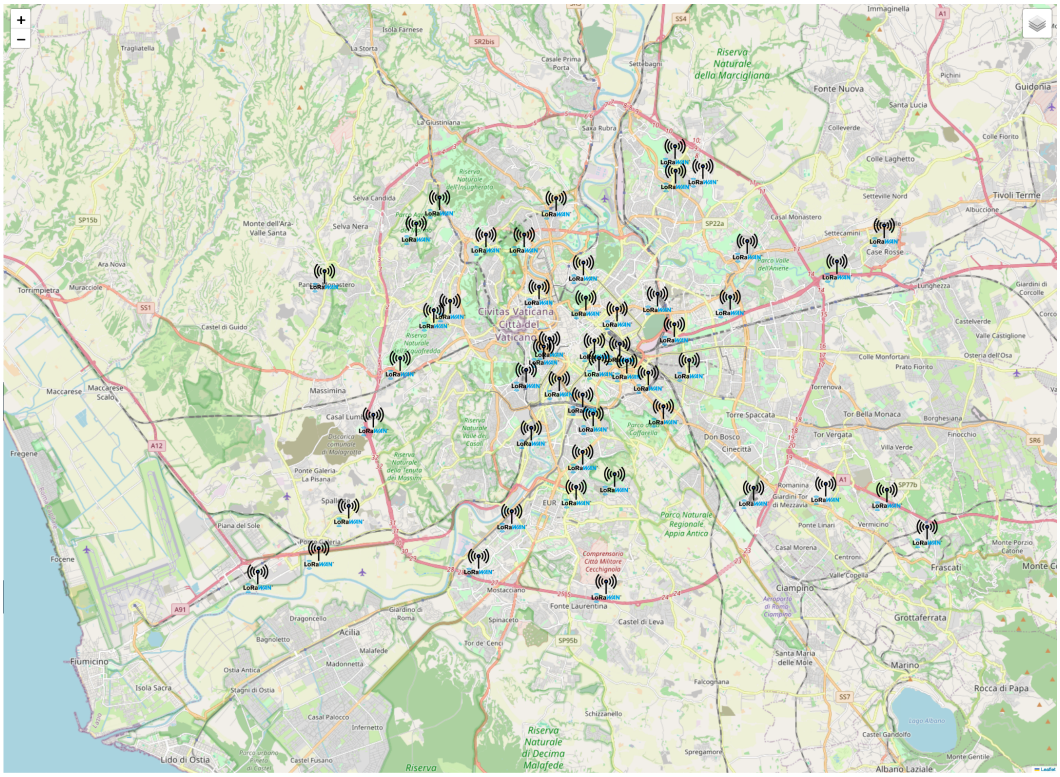

Figure 3.5. Gateways position

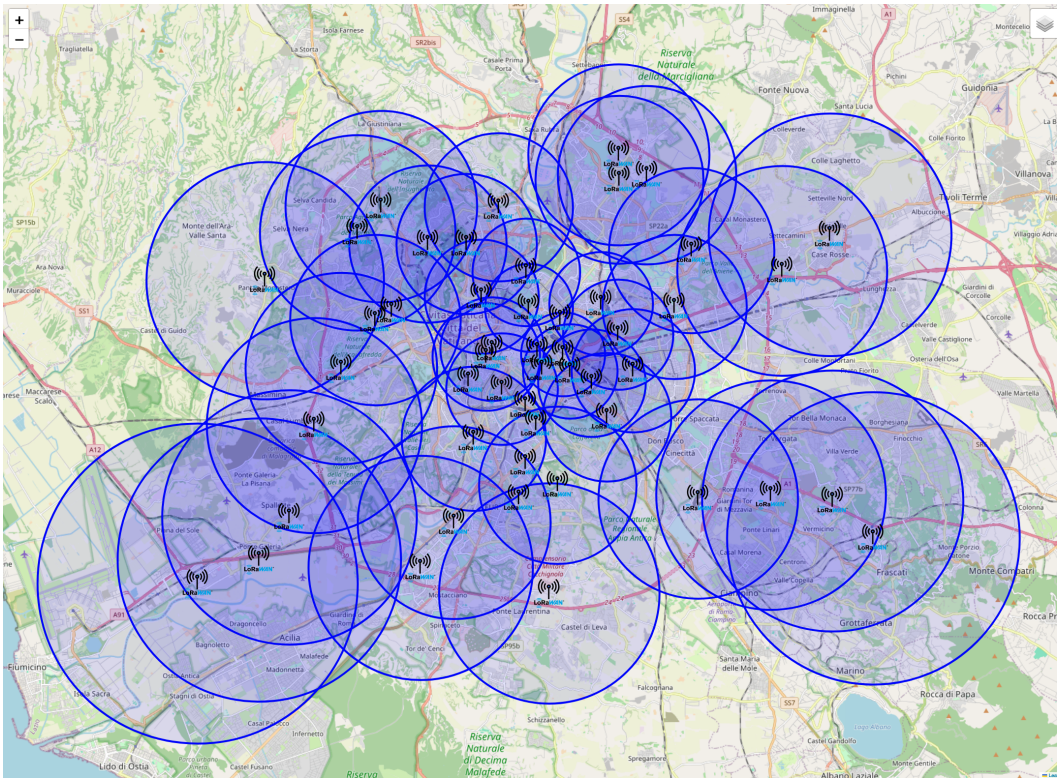

Figure 3.6. Gateways position

3.7 LoRaWAN Network Emulation

To complete the creation of the scenario required for this study, the evaluation of the network status is a crucial step. The dataset, which has already been divided into different snapshots, serves as the foundation for this process. Each snapshot can be seen as a "picture" of the entire system at a specific moment in time. By utilizing these snapshots, it is possible to compute the emulated network characteristics for any given time interval.

For this task, the NS-3 network simulator is employed. NS-3 is capable of taking the snapshots of the system and computing the emulated network characteristics. Specifically, NS-3 requires two input files: the locations of the LoRaWAN gateways and the positions of the end devices. By processing this data, NS-3 can simulate the communication network dynamics in detail, allowing for an in-depth analysis of system behavior under various conditions.

To enable NS-3 to process the dataset, it is necessary to convert all positional data from geographical coordinates (latitude and longitude) into Cartesian coordinates. This conversion ensures compatibility with the NS-3 simulation framework, as it simplifies the computation of distances and relationships between nodes in the network.

The two input files provided to NS-3 are structured as follows:

- **Gateway File:**
  - Gateway's ID: A progressive number that identifies the Gateway
  - X coordinate
  - Y coordinate
  - Z coordinate The assumption made is that all the gateways are positioned at 10 meters of altitude so Z coordinate is 10 for all gateways.
- **Device File:**
  - NODE-ID: an identifier for each node in the form: "TaxiID-Day"
  - X coordinate: extracted from latitude and longitude
  - Y coordinate: extracted from latitude and longitude
  - Z coordinate: is not present in the original dataset, in order to make possible the emulation in NS-3, the taxi altitude is assumed to be 1 meter.

By preparing the dataset in this format and feeding it into NS-3, the simulator can generate a detailed and accurate representation of the LoRaWAN network. This enables the analysis of various metrics, such as communication reliability, network coverage, and device handover events, which are essential for evaluating the performance of the `Edge2Lora` system.

The output of the Emulator of NS-3 is divided into three different folders, one for each different deployment of the scenarios, with 3, 20 and 50 gateways. Each of those folders are organized in the following structure:

- `Info`: A folder containing data regarding the experiment like the total number of packet sent and the list of Spreading Factors used from each device. However the files contained in this folder are not used in the next part of this work.
- `Pkt`: Also this is a folder that contains 3 types of files, each of those files is repeated 288 times. Files types are the following
  - `Snapshot_PKT_TX`: A file containing packet transmission events, each row has the following information for each transmission: `label, timestamp, x, y, z, device_address, spreading_factor,frame_counter`
  - `Snapshot_PKT_RX`: This kind of file contains the information regarding the reception events, including the fields: `label, timestamp, device_address, spreading_factor, frame_counter`
  - `Snapshot_RX_GW`: Also this kind of file stores the information regarding the reception of packets, adding information about the gateway that has received the packet. A single row in this files contains: `label, timestamp, device_address, spreading_factor, frame_counter, freq, RSSI and gw_mac_address`

In the set of produced files, the most informative and suitable for evaluating the capabilities of `Edge2Lora` in a massive scenario is the set of files `Snapshot_RX_GW`. These files include detailed information about the reception of messages at the gateways. Additionally, transmission data is also taken into account, enriching the dataset with valuable network insights.

The goal of creating this dataset extends beyond simply providing a collection of network information related to LoRaWAN networks. A critical aim is to enable the design and testing of various assignment strategies to allocate End Devices to Gateways within `Edge2Lora`.

Data related to the reception at the physical layer, such as Received Signal Strength Indicator (RSSI) and Spreading Factor (SF), are considered vital inputs for these assignment strategies. Consequently, such data is included in the final dataset to facilitate the development and evaluation of innovative and efficient assignment methods.

Starting from `Snapshot_RX_GW` and `Snapshot_PKT_TX` a new folder of files is produced called: "MERGED_TX_RX". Each transmission event is linked to all the receptions of this packet by all the gateways that receive the packet. Each file is organized as a CSV table in the following form:

Label	Timestamp	X	Y	Z	dev_addr	spreading_factor	frame counter	NODE ID	Reception

Table 3.1. Merged TX RX structure

- **Label** is a string identifying the type of the event, in this case is always a transmission event "TX".

- **Timestamp** is included by NS-3 and indicates the time of the transmission event
- **X, Y, Z** this three values are the Cartesian coordinates of the Node that sends the packet.
- **dev_addr** is the identifier of the node in the current snapshot, this is an identifier provided by NS-3, is a progressive number and depends on the number on the order of the devices position in the snapshot file.
- **Spreading Factor** indicates the Spreading Factor used in the transmission of the packet.
- **frame counter**: in NS-3 emulated scenario each taxi sends multiple packets, each packet has its own frame counter.
- **NODE_ID** is the identifier of the Taxi
- **Reception** is a list of reception event, their structure is provided below.

The reception field contains a list of reception events associated to transmission event described from all the other fields. A reception si an entity with the following structure:

Label	Timestamp	sender_addr	spreading_factor	frame counter	frequency	RSSI	Mac Address

Table 3.2. Reception object in TX RX structure

- **Label** is a string identifying the type of the event, in this case, is always a transmission event "RX_GW".
- **Timestamp** is included by NS-3 and indicates the time of the transmission event
- **Sender Address** is the identifier of the sender, not the identifier of the taxi
- **Spreading Factor** used in the transmission
- **Frame Counter** of the packet
- **Frequency** used to transmit the frame, and is always set to 868.1
- **RSSI**, Receive Signal Strenght Indicator is a parameter that describes the quality of received signal
- **Mac Address** is the Mac address of the receiver. Mac addresses are 48 bits identifiers of devices, they are assined in a progressive way starting from 00:00:00:00:00:02, increasing by 2 each time.

Chapter 4

Evaluation of Experimentation Scenarios

Once the network information was emulated and the dataset organized in the proposed structure from the previous section, an analysis of the data contained in this version of the dataset was performed. This analysis is crucial for understanding how devices interact with gateways based on the emulated data. Specifically, it examines changes in the packet loss percentage across the three scenarios involving 3, 20, or 50 gateways, as well as the impact of device mobility on the stability of the link between a device and a gateway.

The primary goal is to provide a scenario that allows for a deeper understanding of the relationship between devices and gateways. In particular, determining the ratio and the average time a device remains within the coverage area of a gateway is vital. Such insights can significantly inform the design of an efficient assignment strategy.

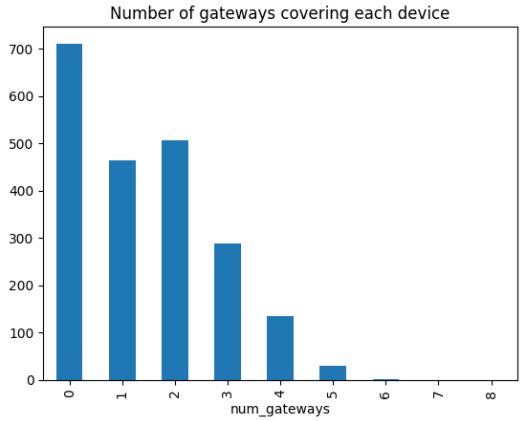

Figure 4.1. Number of gateways that covers a device

The figure 4.1 is produced from the scenario with 50 gateways to visualize the number of gateways covering a device in the network. This figure highlights that a significant number of devices are covered by at least two gateways, ensuring reliable communication. However, at the same time, there exists a substantial set of devices

that are not covered by any gateway.

While this visualization provides an overview of coverage, it does not easily convey the percentage of packets lost or offer a clear representation of link quality performance based on the number of gateways covering each device.

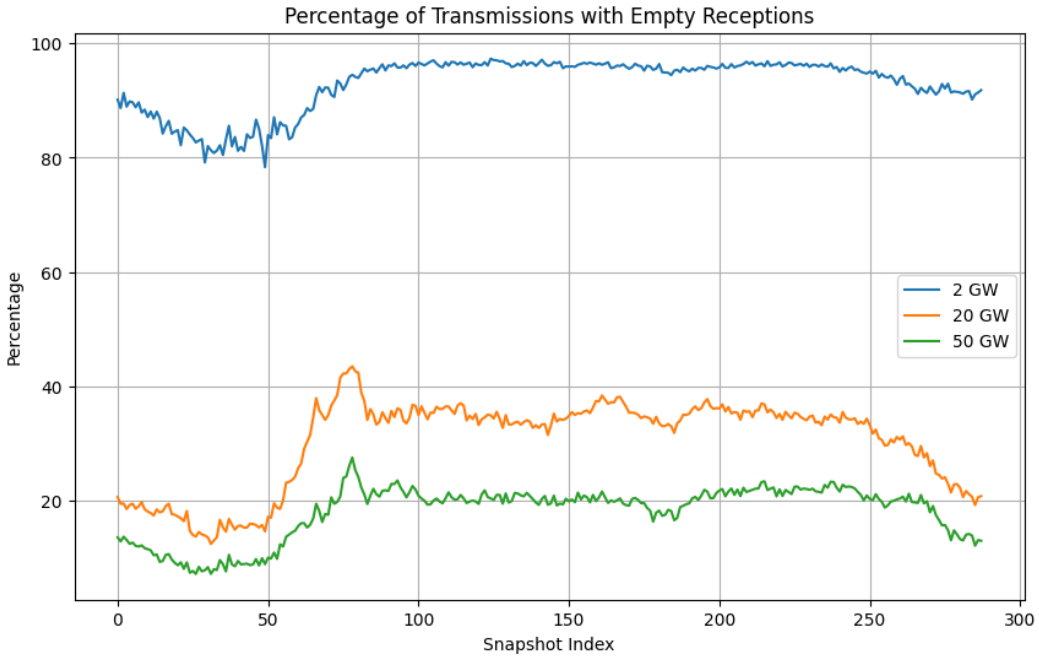

Figure 4.2. Packet Lost in different scenarios

In figure 4.2 there the percentage of packets lost during all the day is printed and can be analyzed. For the scenario with only 2 gateways we cannot expect an high percentage of delivered packets, the percentage of packets lost goes from 80% to 97%. Is not a bad result considering the wide spread of position of taxis, the peak of 20% of delivered packets at snapshot 50 is given by the low number of taxis active at that hour (snapshot 50 is around 4 A.M). At that hour anyone can expect a low number of taxis and most of them concentrated in the center of Rome.

Gateways are positioned always farther than previous from the center of Rome, the first two are positioned near "Roma Termini" train station. The group of 20 contains the previous two and other gateways covering also the center of Rome. The last scenario composed from 50 gateways is composed of all the gateways in the scenario with 20 gateways and other covering more peripheral zones.

In Figure 4.2 is possible to see how percentage of packets lost drops increasing the number of gateways. Adding 18 gateways the number of packets lost drops around 40%. That is interesting and follows in a precise way the distribution of the dataset, the gateways covers the center of Rome where there is the highest density of taxis.

Given the wideness of a City like Rome, also a scenario of 50 gateways is not sufficient to cover all the taxis, a 20% of packets loss remains also in the scenario with the highest number of taxis.

Is also true that an high number of devices is covered by 2 or more gateways, so

there is a wide overlapping of coverage areas between different gateways. This is given also by the fact that in a real scenario LoRaWAN gateways cannot reach high distances and they can cover a distance around 1 km. In NS-3 there is this limitation that does not include the presence of buildings that can generate interference of radio signals.

Another analysis done over the dataset regards the time that a device remains in the coverage area of a Gateway and the Average Spreading factor used from devices to reach a gateway.

4.1 Scenario with a Cluster of devices

The version of the dataset described in the previous section forms the foundation for evaluating the handover capabilities of `Edge2Lora` Gateways in a real-world-like scenario. This dataset was specifically designed to emulate the movement patterns of devices across a metropolitan area, providing a realistic testbed for studying how devices interact with multiple gateways over time.

Beyond the general scenario of dispersed devices, another intriguing scenario was developed to study the behavior of clusters of devices that move collectively, mimicking the movement of a single entity. This type of clustering provides valuable insights into scenarios where multiple devices, such as those in vehicles or groups, operate in close proximity and exhibit similar movement patterns.

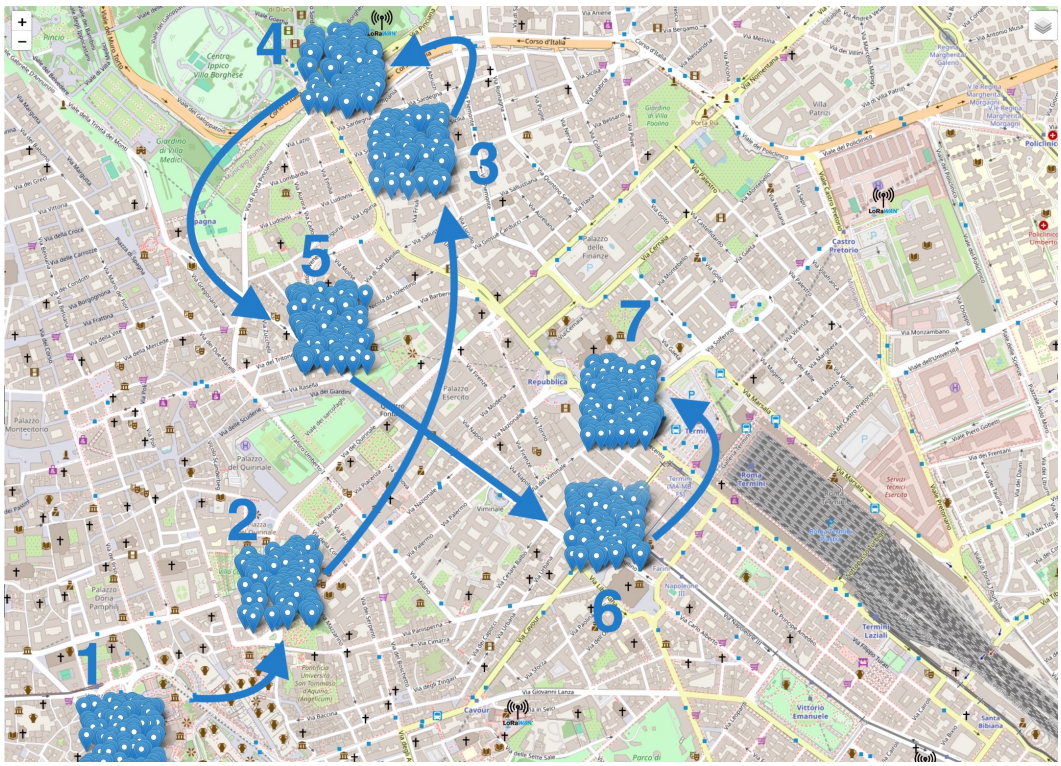

Figure 4.3. Movement of a cluster of devices

To create this scenario, the original dataset containing taxi paths and the

configuration with 50 gateways were utilized. A single taxi path was carefully extracted from this dataset to serve as the basis for the clustered movement. The selection of the taxi path was not arbitrary; rather, it was the result of a detailed analysis of various paths. Particular attention was paid to identifying a path that intersects with a significant number of gateways, ensuring meaningful handover opportunities. A taxi that travels to and from the Fiumicino Airport was chosen for this purpose. This specific path is notable for its interaction with multiple gateways, providing a rich environment to study the effectiveness of `Edge2Lora`'s handover mechanisms in scenarios involving concentrated clusters of devices.

The cluster was created starting from the position information of a single taxi and adding some random noise in a certain area to emulate the presence of multiple devices in that area.

In Figure 4.3 can be seen a sample of the movement of the cluster in the city.

4.2 Deployment with the Dataset

The dataset created in this work, along with a demo showing its use, was presented at the *EWSN'24 International Conference* held in Abu Dhabi in December. During this presentation, a dashboard was demonstrated to illustrate the effectiveness of different balancing techniques employed in the `Edge4Lora` framework. The demo highlighted the impact of these techniques on resource availability across gateways, providing a clear visualization of the results.

An hybrid deployment of the scenario was utilized for the demonstration, combining real gateways with emulated ones. This setup provided a flexible and scalable environment to analyze how various load-balancing strategies affect the performance and resource allocation of gateways. Both scenarios, the one including all taxis and the one including a cluster of devices, were used into the simulation of the `Edge4Lora` infrastructure.

As shown in Figure 4.4, the deployment includes both real and emulated gateways. Some gateways are physical devices, while others are emulated in the cloud. The dataset serves as the input for the *Device Emulator*, a key component responsible for simulating the packets sent by devices in the LoRaWAN network.

The dashboard is an external component that interacts with the `Edge4Lora` infrastructure. It is designed to communicate with other components using MQTT or RPC protocols, enabling real-time parameter tuning for experiments. This functionality provides a flexible platform for exploring the effects of different configurations and balancing techniques.

Within the `Edge4Lora Application Server`, the *Load Balancer* component plays a central role. It establishes the connection between devices and the gateways responsible for processing their data. The dashboard also offers the capability to dynamically switch between different load-balancing algorithms, allowing for comprehensive testing and comparison of various strategies.

This deployment approach demonstrates the adaptability and scalability of the `Edge4Lora` framework, highlighting its ability to handle diverse scenarios and enabling detailed performance analysis.

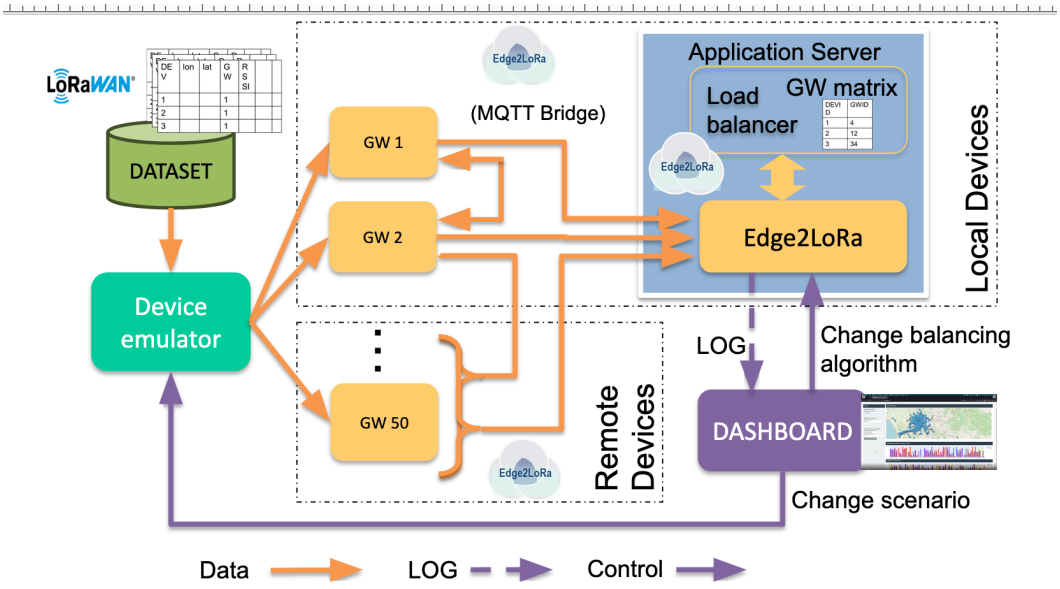

Figure 4.4. Architecture of Deployed Infrastructure

4.2.1 Design of the Dashboard

In experiments involving complex network scenarios such as those studied in this work, a well-designed dashboard is essential for effectively understanding and monitoring the state of the system. The dashboard plays a crucial role in visualizing relevant parameters and ensuring the seamless tracking of system performance. It provides a comprehensive overview of the state of the network by monitoring and displaying key metrics for each gateway.

One of the critical functionalities of the dashboard is its ability to display the current state of the devices in the "real world." This includes the real-time visualization of device positions as well as the positions of the gateways. Such visualizations offer invaluable insights into the spatial distribution of devices and their interactions with the gateways, enabling researchers to identify patterns and potential inefficiencies.

Beyond its role in data visualization, the dashboard also serves as an active component of the experimental framework. As highlighted in the previous section, the dashboard is not only a tool for observing data; it is an interface for interacting with the ongoing experiment. Researchers can use it to dynamically modify the experimental setup by adjusting critical parameters, including:

- Selecting the load-balancing function to be applied.
- Configuring the gateway selection rate.
- Adjusting other key settings that impact the performance of the network.

This dual functionality of the dashboard—combining real-time monitoring with interactive control—makes it a powerful tool for exploring the behavior of the system

under various configurations. By enabling dynamic adjustments and providing immediate feedback, the dashboard facilitates a deeper understanding of the network's performance and supports the optimization of the `Edge4Lora` framework.

Figure 4.5. Dashboard web interface

The final version of the dashboard in Figure 4.5 shows how it is organized. The dashboard can be diviede into 3 different parts:

- **Control Block:** is in the left side of the dashboard, and with this section is possible to edit: the kind of scenario between a cluder of devices or all the taxis, the assignment algorithm, the processing function.
- **Map:** In the upper part of the dashboard there is a map containnig all the gateways and all the devices. Devices updates dynamically and each gateway has its own designed color, whenever a gateway is processed by a certain gateway, it will be colored with the color of the gateway. This is an effective visualization of gateway device association.
- **Gateways metrics:** In the lower side of the dashboard there is a pair of barplots. The first barplot represents the current state of the LoRaWAN packets for that gateway, including: Received Frames, Processed Frames, Forwarded Frames (FWD), and Transmitted Frames after processing. The second barplot highlights the resources used for each gateway.

Chapter 5

Conclusions

This thesis explored a mobility scenario within the context of Low Power Wide Area Networks (LPWANs), focusing on the development of a dataset designed to evaluate the capabilities of the `Edge2LoRa` and `Edge4LoRa` frameworks in handling device mobility. The dataset provides valuable insights into network behavior under dynamic conditions and serves as a foundational tool for advancing edge computing in LPWANs. The obtained results contribute to the LPWAN domain and support ongoing research conducted at Sapienza University in enabling Edge Computing within these networks.

5.1 Key Takeaways from the Research

During the course of this work, I gained significant insights into the complexities of designing and evaluating IoT networks, particularly in mobility scenarios. Developing the dataset required an in-depth understanding of:

- The role of realistic mobility patterns in simulating device behaviors, and how these patterns influence network performance metrics such as packet loss, gateway load, and handover efficiency.
- The importance of balancing computational efficiency with data fidelity, as higher granularity in mobility data can improve simulation accuracy but also increases computational demands.
- The challenges of simulating radio frequency (RF) propagation in dynamic environments, including interference, multipath effects, and signal scattering, which are critical for improving real-world applicability.

I also deepened my understanding of edge computing principles, LoRaWAN architecture, and the importance of designing frameworks that address the inherent limitations of centralized systems. This experience emphasized the need for multidisciplinary approaches, combining insights from networking, data science, and machine learning to address emerging IoT challenges.

5.2 Dataset Limitations

While the proposed dataset is well-organized and provides a significant contribution to the field, several limitations remain, which present opportunities for improvement:

- **Morphological Area Consideration:** The current emulator does not account for the physical geography or morphological characteristics of the deployment area. For instance, the effects of buildings, terrain elevation, and natural obstacles on signal propagation are not simulated.
- **Signal Propagation Modeling:** The dataset does not fully consider RF-specific phenomena such as multipath interference, scattering, and attenuation. Incorporating advanced propagation models could significantly improve the quality and realism of the data.

Addressing these limitations would enhance the dataset's applicability for testing advanced algorithms and further improving LPWAN performance in real-world deployments.

5.3 Future Works

The work presented in this thesis lays the groundwork for future research aimed at optimizing load balancing and device-gateway assignment in Edge2LoRa. Several directions for further exploration include:

- **Optimal Load Balancing Strategies:** Future efforts should focus on developing and testing algorithms that optimize load distribution across gateways. These strategies could be based on real-time LoRaWAN transmission metrics, such as Received Signal Strength Indicator (RSSI) and Spreading Factor (SF), as well as device positions.
- **Integration of Machine Learning:** Machine Learning (ML) offers promising solutions for automating and optimizing device-to-gateway assignments. Training ML models using rich datasets like the one developed in this work could enable predictive decision-making, adapting network behavior to dynamic conditions. An approach that is now considered a efficient solution from our research team is to use a Q-Learning approach to assign devices and gateways.
- **Real-World Validation:** Deploying the enhanced `Edge4LoRa` framework in real-world environments would provide invaluable insights into its performance and reliability under practical conditions. Such deployments could also highlight areas for refinement and scalability improvements.
- **Dataset Enrichment:** The current dataset includes only transmission and reception events along with network data. As a next step, additional application-level information should be incorporated into the dataset to enhance its usability and enable broader publication.

5.4 Closing Remarks

This thesis demonstrates the potential of integrating edge computing with LoRaWAN networks to address critical challenges in mobility and scalability. By developing a realistic dataset and extending the `Edge2LoRa` framework, this work contributes to advancing IoT networking for dynamic, high-density environments. The insights gained here not only validate the feasibility of edge-enabled LPWANs but also open avenues for future research in intelligent resource allocation, robust data processing, and real-world deployments. The journey of innovation in this domain continues, with the goal of creating smarter, more adaptive networks for the IoT-driven future.

Bibliography

[1] Ling Qian et al. "Cloud Computing: An Overview". In: *Cloud Computing*. Ed. by Martin Gilje Jaatun, Gansen Zhao, and Chunming Rong. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 626–631. ISBN: 978-3-642-10665-1.

[2] Blesson Varghese et al. "Challenges and Opportunities in Edge Computing". In: *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. 2016, pp. 20–26. DOI: 10.1109/SmartCloud.2016.18.

[3] Rúben Oliveira, Lucas Guardalben, and Susana Sargento. "Long range communications in urban and rural environments". In: *2017 IEEE Symposium on Computers and Communications (ISCC)*. 2017, pp. 810–817. DOI: 10.1109/ISCC.2017.8024627.

[4] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. "Low Power Wide Area Networks: An Overview". In: *IEEE Communications Surveys Tutorials* 19.2 (2017), pp. 855–873. DOI: 10.1109/COMST.2017.2652320.

[5] Ramon Sanchez-Iborra et al. "Performance Evaluation of LoRa Considering Scenario Conditions". In: *Sensors* 18.3 (2018). ISSN: 1424-8220. DOI: 10.3390/s18030772. URL: https://www.mdpi.com/1424-8220/18/3/772.

[6] Lorenzo Bracciale et al. *CRAWDAD roma/taxi*. 2022. DOI: 10.15783/C7QC7M. URL: https://dx.doi.org/10.15783/C7QC7M.

[7] Ivan Fardin et al. "Enabling Edge Computing over LoRaWAN: A Device-Gateway Coordination Protocol". In: *Proceedings of the 12th ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*. DIVANet '22. Montreal, Quebec, Canada: Association for Computing Machinery, 2022, pp. 23–30. ISBN: 9781450394826. DOI: 10.1145/3551662.3560926. URL: https://doi.org/10.1145/3551662.3560926.

[8] LoRa Alliance. *LoRaWAN® Specification v1.1*. Available at https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1. Sept. 2023.

[9] Lorenzo Frangella Stefano Milani Domenico Garlisi Ioannis Chatzigiannakis. "Enhancing LoRaWAN Networks with Edge Computing: A Demonstration on a Large-Scale Scenario". In: (2024).

[10] Stefano Milani et al. "Edge2LoRa: Enabling edge computing on long-range wide-area Internet of Things". In: *Internet of Things* 27 (2024), p. 101266. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2024.101266. URL: https://www.sciencedirect.com/science/article/pii/S2542660524002075.

[11] URL: https://pandas.pydata.org/.

[12] URL: https://www.python.org/.

[13] URL: https://python-visualization.github.io/folium/latest/.

[14] URL: https://numpy.org/.

[15] URL: https://dash.plotly.com/.

[16] URL: https://matplotlib.org/.

[17] URL: https://www.ns3.it/.

[18] Carlo Carugno. "Design, Development and Evaluation of a LoRaWAN Packet Simulator for Testing Realistic Large-Scale Experimentation LoRaWAN Deployments". MA thesis. Sapienza University of Rome.

[19] Porto City. *Porto Taxis' data*. URL: https://paperswithcode.com/dataset/porto-taxi.

[20] *Edge2Lora Github page*. URL: https://github.com/Edge2LoRa.

[21] Ivan Fardin. "Design, Implementation and Evaluation of a Gateway-Device Coordination Protocol to enable Edge Computing over LoRaWAN". MA thesis. Sapienza University of Rome.

[22] Stefano Milani. "Edge2LoRa: A New Paradigm for Enabling Cloud Edge Computing Continuum over LoRaWAN". PhD thesis. Sapienza University of Rome.

[23] NYC. *TLC Trip Record Data*. URL: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[24] Christian Tedesco. "Analyzing Distributed Processing Applications in Edge Computing Environments: A Case Study with Realistic Implementation and Evaluation using a Lo- RaWAN Architecture". MA thesis. Sapienza University of Rome.